COUNTERFACTUAL SELF-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Unlike traditional supervised learning, in many settings only partial feedback is available. We may only observe outcomes for the chosen actions, but not the counterfactual outcomes associated with other alternatives. Such settings encompass a wide variety of applications including pricing, online marketing and precision medicine. A key challenge is that observational data are influenced by historical policies deployed in the system, yielding a biased data distribution. We approach this task as a domain adaptation problem and propose a self-training algorithm which imputes outcomes for the unseen actions in the observational data to simulate a randomized trial. We offer a theoretical motivation for this approach by providing an upper bound on the generalization error defined on a randomized trial under the self-training objective. We empirically demonstrate the effectiveness of the proposed algorithms on both synthetic and real datasets.

1 INTRODUCTION

Counterfactual inference (Pearl et al., 2000) attempts to address a question central to many applications - *What would be the outcome had an alternative action was chosen?* It may be selecting relevant ads to engage with users in online marketing (Li et al., 2010), determining prices that maximize profit in revenue management (Bertsimas & Kallus, 2016), or designing the most effective personalized treatment for a patient in precision medicine (Xu et al., 2016). With observational data, we have access to past actions, their outcomes, and possibly some context, but in many cases not the complete knowledge of the historical policy which gave rise to the action (Shalit et al., 2017). Consider a pricing setting in the form targeted promotion. We might record information of a customer (context), promotion offered (action) and whether an item was purchased (outcome), but we do not know why a particular promotion was selected.

Unlike traditional supervised learning, we only observe feedback for the chosen action in observational data, but not the outcomes associated with other alternatives (*i.e.*, , in the pricing example, we do not observe what would occur if a different promotion was offered). In contrast to the gold standard of a randomized controlled trial, observational data are influenced by historical policy deployed in the system which may over or under represent certain actions, yielding a biased data distribution. A naive but widely used approach is to learn a machine learning algorithm directly from observational data and use it for prediction. This is often referred to as direct method (DM) (Dudík et al., 2014). Failure to account for the bias introduced by historical policy often results in an algorithm which has high accuracy on the data it was trained on, but performs considerably worse under a different policy. For example in the pricing setting, if historically most customers who received high promotion offers bear a certain profile, then a model based on direct method may fail to produce reliable predictions on these customers when low offers are given.

To overcome the limitations of direct method, Shalit et al. (2017); Johansson et al. (2016); Lopez et al. (2020) cast counterfactual learning as a *domain adaptation* problem, where the source domain is observational data and the target domain is a *randomized trial* whose assignment of actions follows a uniform distribution for a given context. The key idea is to map contextual features to an embedding space and jointly learn a representation that encourages similarity between these two domains, leading to better counterfactual inference. The embedding is generally learned by a neural network and the estimation of the domain gap is usually slow to compute.

In this paper, while we also view counterfactual inference as a domain adaptation problem between observational data and an ideal randomized trial, we take a different approach - instead of estimating

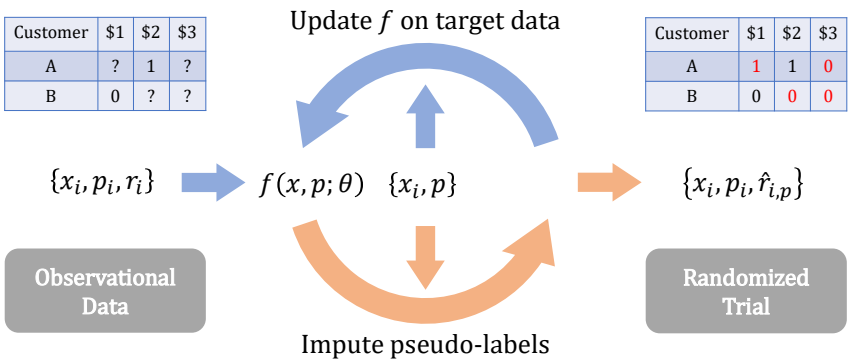

Figure 1: Illustration of the proposed Counterfactual Self-Training (CST) framework. There are two sales records (observational data) shown in the table, *i.e.*, , Customer A was offered $2 and bought an item; Customer B was offered $1 and did not buy. The question marks in the tables represent the counterfactual outcome which we do not observe. For all these unseen counterfactual outcomes, pseudo-labels which are colored in red in the tables are imputed by a model and are used to augment the observational data. The model is subsequently updated by training on both the imputed counterfactual data and the factual data. This iterative training procedure continues until it converges.

the domain gap between the two distributions via an embedding, we explicitly simulate a randomized trial by imputing pseudo-labels for the unobserved actions in the observational data. The optimization process is done by iteratively updating the pseudo-labels and a model that is trained on both the factual and the counterfactual data, as illustrated in Figure 1. As this method works in a self-supervised fashion (Zou et al., 2018; Amini & Gallinari, 2002), we refer to our proposed framework as Counterfactual Self-Training (CST).

The contribution of our paper is as follows. First, we propose a novel self-training algorithm for counterfactual inference. To the best of our knowledge, this is the first application of self-training algorithm for learning from observational data. Moreover, in contrast to the existing methods from domain adaption on counterfactual inference, CST is flexible and can work with a wide range of machine learning algorithms, not limited to neural networks. Second, we offer a theoretical motivation of our approach by providing an upper bound on the generalization error defined on a randomized trial under the self-training objective. In other words, we show that the counterfactual self-training algorithm helps minimizing the risk on the target domain. Our theoretical bounds suggest generating pseudo-labels with random imputation, which is a methodological departure from traditional self-training algorithms which impute hard labels. Third, we present comprehensive experiments on several synthetic datasets and three counterfactual learning datasets converted from multi-label classification tasks to evaluate our method against state-of-the-art baselines. In all experiments, CST shows competitive or superior performance against all the baselines. Moreover, our algorithm is easy to optimize with a much faster training time than other baselines.

2 Related Work

Counterfactual estimation has received a lot of attention in the machine learning community in the recent years (Swaminathan & Joachims, 2015a; Joachims et al., 2018; Shalit et al., 2017; Lopez et al., 2020; Kallus, 2019; Kallus & Zhou, 2018). Most of the proposed algorithms can be divided into two categories: counterfactual risk minimization (CRM) and direct method (DM). Both can be used together to construct doubly robust estimators (Dudík et al., 2014) to further improve efficiency. CRM, also known as off-policy learning or batch learning from bandit feedback, typically utilizes inverse propensity weighting (IPW) (Rosenbaum, 1987; Rosenbaum & Rubin, 1983) to account for the bias in the data. Swaminathan & Joachims (2015a) introduces the CRM principle with a variance regularization term derived from an empirical Bernstein bound (Maurer & Pontil, 2009) for finite samples. In order to reduce the variance of the IPW estimator, Swaminathan & Joachims (2015b) proposes a self-normalized estimator, while BanditNet (Joachims et al., 2018) utilizes the baseline technique (Greensmith et al., 2004) in deep nets. As pointed out by Lefortier et al. (2016), CRM-

based methods tend to struggle with medium to large action spaces in practice. Morever, CRM-based methods generally require a known and stochastic logging policy, along with full support on the action space. When either one of the requirements is violated, Sachdeva et al. (2020); Kang et al. (2007) observe the direct method often demonstrates a more robust performance. When the logging policy is not available, the counterfactual learning problem is often referred to as learning from observational data, which is the setting we focus on.

Learning from observational data is also closely related to estimating Individualized Treatment Effects (ITE) (Shpitser & Pearl, 2012), which is defined as the difference of expected outcomes between two actions, with respect to a given context. The main challenge of identifying ITE is that unlike an ideal randomized trial, observational data is biased and we do not have the access to the counterfactuals. Hill (2011) uses a bayesian nonparametric algorithm to address this issue. Yoon et al. (2018) proposes using generative adversarial nets to capture the uncertainty in the counterfactual distributions to facilitate ITE estimation. Johansson et al. (2016); Shalit et al. (2017) approach counterfactual inference with representation learning and domain adaptation. Their key idea is to learn a representation between observational data and a randomized trial that encourages better generalization on all possible actions. It is achieved by minimizing a weighted sum of the factual loss on the observational data (loss for direct method) plus an estimated domain gap measured by integral probability metrics. Lopez et al. (2020) further extends this framework to multiple treatments using Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2008) and achieves state-of-the-art performance. The HSIC proposed in Lopez et al. (2020) has a computation time of at least $\mathcal{O}(N^2)$, making its training process slow. While the aforementioned methods and our approach can be viewed as extensions to direct method, we tackle the domain adaptation problem differently by explicitly augmenting the observational data to create a simulated randomized trial via self-training.

Self-training algorithms have been widely studied in semi-supervised learning and domain adaptation problems (Nigam et al., 2000; Amini & Gallinari, 2002; Grandvalet & Bengio, 2005; Zou et al., 2019; Han et al., 2019). Grandvalet & Bengio (2005) proposes to use entropy regularization for semi-supervised learning as a class-overlapping measure to utilize unlabeled data. Nigam et al. (2000); Amini & Gallinari (2002); Zou et al. (2019) formulate the pseudo-label imputation as classification EM algorithm and show its convergence under proper initialization. Han et al. (2019) points out that pseudo-label imputation can be viewed as minimizing min-entropy as a type of Rényi entropy $\frac{1}{1-\alpha}\log(\sum_{i=1}^{n} p_i^{\alpha})$ when $\alpha \to \infty$, and Shannon entropy in Grandvalet & Bengio (2005) is the case when $\alpha \to 1$. Self-training is also shown to be effective in semi-supervised learning for many other machine learning models besides neural networks (Tanha et al., 2017; Li et al., 2008). It is worthy to mention that unlike traditional self-training where the target domain is given by the problem, we propose to construct a target domain by imputing pseudo-labels on all unseen actions to simulate a pseudo-randomized trial. Moreover, instead of hard labels used in traditional self-training, we propose to use random imputation to create pseudo-labels which have a theoretical motivation tailored for counterfactual inference and are shown to be more effective based on the experiments results.

3 Problem Statement

Following the notation in Lopez et al. (2020), we use $\mathcal{X}$ to represent an abstract space and $\mathbb{P}(x)$ is a probability distribution on $\mathcal{X}$. Each sample $x = x_1, \cdots, x_n \in \mathcal{X}^n$ is drawn independently from $\mathbb{P}(x)$. $\mathcal{P}$ is the discrete action space that a central agent can select for each sample, after which a reward r is revealed to the agent. In precision medicine, $\mathcal{X}$ may represent a patient cohort, $\mathcal{P}$ refers to feasible treatment for a disease, and r can be the indicator of whether a patient survives after the treatment. Similarly, $\mathcal{X}, \mathcal{P}, r$ can represent visitors, ads shown and click-through-rates in online marketing.

We focus on an example of pricing to illustrate our method. We use $x \in \mathcal{X}^n \sim \mathbb{P}(x)$ to denote a customer. Let $\mathcal{P}$ represent discrete price options a central agent can offer to customers. After offering price $p \in \mathcal{P}$, the agent observes the response from the customer $r \sim \mathbb{P}(r|x, p)$, *i.e.*, , either a 1 (buy) or a 0 (no-buy). As a direct method, the task is to learn a function $f(x, p)$ by minimizing the loss $\mathbb{E}_{x\sim\mathbb{P}(x), p\sim\pi_0(p|x)}\mathcal{L}(f(x, p), r)$ where $\pi_0(p|x)$ is a randomized assignment policy (Shalit et al., 2017; Lopez et al., 2020). The estimation task is often referred to as demand estimation (Wales & Woodland, 1983), which is critical for many downstream decisions such as inventory optimization and revenue management (Kök & Fisher, 2007; McGill & Van Ryzin, 1999). This is in contrast to

CRM-based methods which use the final reward as objective to learn a policy $\pi(p|x)$ that maximizes $\mathbb{E}_{x\sim\mathbb{P}(x),p\sim\pi(p|x)}\mathbb{E}[r|x,p]$ (Swaminathan & Joachims, 2015a).

With observational data, the individualized treatment effect is not always identifiable. We use Rubin's potential outcome framework and assume consistency and strong ignorability which is a sufficient condition for identifying ITE (Imbens & Wooldridge, 2009; Pearl, 2017). Here we formally present the ignorability assumption (Rubin, 2005; Shalit et al., 2017):

Assumption 3.1 (Ignorability)**.** *Let $\mathcal{P}$ be action set, x is context (feature), $r(p)|x$ is observed reward for action $p \in \mathcal{P}$ given context x, $r(p) \perp\!\!\!\perp p|x, \forall p \in \mathcal{P}$.*

In other words, we assume there is no unobserved confounders. This condition generally cannot be made purely based on data and requires some domain knowledge.

4 ALGORITHM

In this section, we introduce Counterfactual Self-Training (CST) algorithm, which can be viewed as an extension of the direct method via domain adaptation. Unlike existing methods using representation learning, we propose a novel self-training style algorithm to account the bias inherent in the observational data.

4.1 SELF-TRAINING

Self-training has recently been used in unsupervised domain adaptation (UDA) and semi-supervised learning (SSL) and achieved great success (Zou et al., 2019; Han et al., 2019; Zou et al., 2018; Amini & Gallinari, 2002; Nigam et al., 2000; Grandvalet & Bengio, 2005). The self-training algorithm works in an iterative fashion: First, after training a classifier $f(x,p)$ on a source dataset, pseudo-labels are created by the best guess of f. Next, the model is trained on a target dataset, and the trained model is used to generate new pseudo labels. This idea is illustrated in Figure 1.

To formulate the counterfactual learning problem as a domain adaptation problem, observational data is viewed as data sampled from a source distribution $\mathcal{D}_S = \mathbb{P}(x)\pi(p|x)$. The target domain is a randomized trial on the same feature distribution to ensure a uniformly good approximation on all actions. Our goal is to transfer observational data from the source domain to a simulated pseudo-randomized trial via self-training. To accomplish this, we first train an initial classifier $f_0(x,p)$ on observational data, then impute pseudo-labels on all unseen actions from the observation data with $\hat{r}_{i,p} \sim f(x_i,p)$. The model is then updated by training with the following objective:

$$\min_{\theta}\mathcal{L}_{ST} = \frac{1}{N|\mathcal{P}|}\Big(\underbrace{\sum_{i=1}^{N} l(f_\theta(x_i,p_i),r_i)}_{\mathcal{L}_{src}} + \sum_{i=1}^{N}\sum_{p\in\mathcal{P}\setminus p_i} l(f_\theta(x_i,p),\hat{r}_{i,p})\Big) \tag{1}$$

The first term $\mathcal{L}_{src}$ in Equation 1 corresponds to the loss used in direct method, defined over the factual data alone. Meanwhile, the second term refers to the loss defined over the imputed counterfactual data. In other words, in order to get a good model across all actions, we utilize the pseudo-population induced from imputation which represents a simulated randomized trial. We iteratively train the model and impute pseudo-labels until it converges. The algorithm is stated in Algorithm 1.

Note that a key difference between our CST algorithm and traditional self-training (ST) methods for unsupervised domain adaptation (such in Zou et al. (2018)): Pseudo-labels in traditional ST are generated from hard imputation while ours are sampled from a probability distribution as illustrated in Algorithm 1 line 4. Not only this randomized imputation has a theoretical motivation presented in Section 4.2, it also demonstrates superior performance over hard imputation in our experiments in Section 5.

4.2 THEORETICAL MOTIVATION

As our objective is to augment observational data to a randomized trial such that the learnt model is able to perform better on all feasible actions, we focus on bounding the generalization error defined

Algorithm 1 Counterfactual Self-Training

```
while NOT converged do                                          ▷ Main training loop
    for each i ∈ {1 ... N} do
        for each p ∈ P \ p_i do
            Impute pseudo-label r̂_{i,p} ~ P_θ(r|x_i, p).        ▷ Pseudo-label imputation
        end for
    end for
    Update θ by minimizing L_ST defined in Equation 1.          ▷ Self-training
end while
```

on a randomized trial. We use $\mathcal{D}$ to represent the distribution of a true randomized trial where the assignment policy is a uniform probability over $\mathcal{P}$ given context, and $\hat{\mathcal{D}}$ is the distribution of pseudo-label generated by the current model output $P_\theta(r|x,p)$. Define $\mathcal{R}_{\mathcal{D}}(f)$ as the risk of function f with respect to a loss function $l(\cdot,\cdot)$ as $\mathcal{R}_{\mathcal{D}}(f) = \mathbb{E}_{x,p\sim\mathcal{D}}[l(f(x,p),p)]$, and $\hat{\mathcal{R}}_{\hat{D}}(f)$ as empirical risk on $\hat{D}$. Assume our classifier outputs a probability estimation $P_\theta(r|x,p)$ for a feature and action pair (x,p), and we use a random imputation $\hat{r} \sim P_\theta(r|x,p)$ to generate outcomes for the unseen actions. We have the following theorem on the generalization bound:

Theorem 1. *Assume $\mathbb{P}_\theta(r|x,p) \geq \frac{1}{M_0+1}$, where $M_0 > 1$ is constant, let $M = \min\{\max(\frac{\mathbb{P}}{\mathbb{P}_\theta} - 1), M_0\}$, $f^\star = \operatorname{argmin}_{f\in\mathcal{F}} \mathcal{R}_{\mathcal{D}}(f)$, $\hat{\mathcal{D}}$ is the dataset generated by random imputation of current model output $\mathbb{P}_\theta$, and $\hat{f}$ minimizes the empirical risk on $\hat{D}$. For any loss function $l(\cdot,\cdot)$, we have:*

$$\mathcal{R}_{\mathcal{D}}(\hat{f}) - \mathcal{R}_{\mathcal{D}}(f^\star) \leq C(\sqrt{\frac{V}{n}} + \sqrt{\frac{\log(1/\delta)}{n}}) + (M+1)\hat{\mathcal{R}}_{\hat{\mathcal{D}}}(\hat{f}) - \mathcal{R}_{\mathcal{D}}(f^\star) \tag{2}$$

The proof is in Appendix A.1. By replacing M with M_0 and minimizing the right hand side of Equation 2 over θ, we recover Equation 1 which is the objective that we are optimizing in the training procedure. This complete optimization involves optimizing over θ and $\hat{r}$, and can be solved via classification EM (CEM) (Amini & Gallinari, 2002) and traditional self-training is an instance of CEM (Zou et al., 2019). These methods use a hard label as a classification step to impute the pseudo-labels but it is not clear how it relates to the risk that we are interested in. To establish Theorem 1, we require a random imputation of labels based on the probability output of the classifier to upper bound the risk under a true randomized trial using this objective. Therefore, we use a random sampling to generate pseudo-labels in our algorithm, and it is shown to be more robust than hard labels in our experiments. We would like to note this bound is relatively loose when $\mathbb{P}$ is very different from $\mathbb{P}_\theta$, thus we only use it as a motivation of our proposed algorithm. Since in the source domain, $\mathbb{P} = \mathbb{P}_\theta$, it is possible to get a tighter upper bound, which we leave for future work.

Our assumption $\mathbb{P}_\theta(r|x,p) \geq \frac{1}{M_0+1}$ is also proposed in Zou et al. (2019) in the form of entropy regularization to prevent the model converging too fast and getting too confident in the early stages of training. Since cross-validation is biased in the counterfactual learning due to the logging policy (Lopez et al., 2020), we avoid using hyperparameters and do not use this regularization in our experiments.

5 Experiments

We construct synthetic datasets for a pricing example and utilize three real datasets to demonstrate the efficacy of our proposed algorithm. Implementationwise, we use a three layer neural network with 128 nodes as our model and binary entropy loss as the loss function. We avoid using early stopping and train each method until convergence to ensure a fair comparison. The following baselines are considered in our experiments:

- Direct Method (DM): This baseline directly trains a model on observational data
- HSIC (Lopez et al., 2020): We use the last layer as an embedding and calculate HSIC between the embedding and the actions. The training objective is binary cross entropy loss + λ·HSIC, where λ is the hyperparameter which we choose from a grid search over $[0.01, 0.1, 1, 10, 100]$.

- BanditNet (Joachims et al., 2018): BanditNet is a CRM-based method developed for deep nets. For the baseline required in BanditNet, we normalize the reward as in Swaminathan & Joachims (2015a) and choose the hyperparameter using a grid search over $[0, 0.2, 0.4, 0.6, 0.8]$ and cross validation.

Since BanditNet is designed for reward maximization, evaluation of the accuracy (*i.e.*, hamming loss) is not appropriate under our problem. In each experiment, we only evaluate BanditNet in the reward comparison. We also experiment with two versions of CST, one with *random imputation* (CST-RI) and one with deterministic pseudo-labels commonly used in self-training literature by *argmax operation* $\hat{r} = \text{argmax}_r f(r|x, p)$, which we refer to as CST-AI. Unlike CST, HSIC and BanditNet require a hyperparameter as an input to their algorithms. Following Joachims et al. (2018); Lopez et al. (2020), we use a 5-fold cross-validation and grid search to select the hyperparameter for all experiments. All experiments are conducted using one NVidia GTX 1080-Ti GPU with five repetitions. Mean and standard error are reported for each metric.

5.1 Synthetic Datasets

In synthetic experiments, we use a pricing example similar to the experiment in Lopez et al. (2020). Let $U(\cdot, \cdot)$ be a uniform distribution. Assume customer features are a 50-dimensional vector X drawn from $U(0, 1)^{50}$ and there are 10 price options from \$1 to \$10. The logging policy is set as $\pi(p = i|x) = \frac{x_i}{\sum_{i=1}^{10} x_i}$. Five types of demand functions are simulated, and the complete data generation process is detailed in Appendix A.2.

We generate 1000 samples for each demand function and report hamming loss which relies on the hard labels generated by the algorithm in Table 1. In addition, as we are interested in probability estimation, we report the multi-label soft margin loss in Table 2. Lastly, as a pricing application, we also evaluate the revenue generated on the test set by solving the revenue maximization problem:

$$p_i = \underset{p}{\text{argmax}} \ \mathbb{P}(r = 1|x_i, p) \cdot p \tag{3}$$

For each dataset, the test set has 5000 samples from the corresponding demand distribution. The results are shown in Table 3.

Among all datasets, CST-RI has the best performance in terms of both hamming loss and soft margin loss. HSIC outperforms DM baseline by a significant margin and comes as a close second to CST-RI. In 4 out of 5 demand functions (with the exception of D1), CST-RI achieves a comparable or superior performance on reward as shown in Table 3. Hence, while CST-RI results in the best demand model in terms of the losses, it does not guarantee the highest revenue in all cases. This is because the downstream optimization task is independent from demand estimation (Elmachtoub & Grigas, 2017). Nevertheless, CST-RI significantly outperforms BanditNet which is designed for reward maximization due to unknown logging policy (Kang et al., 2007). We also want to point out that CST-AI performs worse than DM which is a naive baseline, demonstrating the importance of random imputation in our algorithm.

	D1	D2	D3	D4	D5
Direct Method	0.2228±0.0035	0.2361±0.0026	0.3422±0.0045	0.2778±0.0060	0.1913±0.0038
CST-RI	**0.1799**±0.0029	**0.1836**±0.0024	**0.2692**±0.0047	**0.2051**±0.0058	**0.1464**±0.0043
CST-AI	0.5525±0.0159	0.4460±0.0239	0.5094±0.0348	0.5296±0.0212	0.4964±0.0495
HSIC	0.2023±0.0058	0.2143±0.0000	0.3006±0.0052	0.2103±0.0060	0.1513±0.0044

Table 1: Hamming Loss on Synthetic Dataset.

	D1	D2	D3	D4	D5
Direct Method	2.0949±0.1391	2.0877±0.0410	3.86±0.1343	3.0224±0.2116	1.8476±0.1559
CST-RI	**0.3832**±0.0019	**0.4068**±0.0023	**0.5240**±0.0044	**0.4412**±0.0104	**0.3323**±0.0070
CST-AI	1.5415±0.0300	1.3053±0.0583	1.5049±0.0854	1.5074±0.0519	1.3814±0.1065
HSIC	0.4673±0.0104	0.4551±0.0017	0.5891±0.0069	0.4591±0.0109	0.3578±0.0098

Table 2: Multi-Label Soft-Margin Loss on Synthetic Datasets

	D1	D2	D3	D4	D5
Direct Method	**5506.8**±396.0	4279.6±109.3	**9241.0**±509.0	5082.4±270.2	3664.0±291.2
CST-RI	4816.8±305.05	**4656.6**±195.7	8917.0±488.1	**5368.0**±374.8	**3939.0**±317.15
CST-AI	4077.4±182.0	2212.4±138.4	8366.4±422.4	2532.6±251.3	1297.0±54.7
HSIC	4178.0±214.5	4649.8±323.4	8447.2±406.3	5017.0±323.2	3099.4±448.4
BanditNet	3929.8±354.04	4530.2±159.3	8385.8±395.4	5087.0±454.2	2838.8±463.3
Skyline	7267.6±363.2	5082.0±0	11357.4±640.9	6997.8±424.1	5084.0±315.8

Table 3: Total Reward on Synthetic Dataset. Skyline represents the best possible reward with perfect knowledge of the demand function.

5.2 Multi-Label Datasets

We use three multi-label datasets from LIBSVM repository (Elisseeff & Weston, 2002; Boutell et al., 2004; Chang & Lin, 2011), which are used for semantic scene, text and gene classification. We convert the supervised learning datasets to bandit feedback by creating a logging policy using 5% of the data following Swaminathan & Joachims (2015a); Lopez et al. (2020). More specifically, each feature x has a label $y \in \{0,1\}^p$ where p is the number of labels. After the logging policy selects a label (action) i, a reward y_i is revealed as bandit feedback (x, i, y_i), *i.e.*, , for each data point, if the policy selects one of the correct labels of that data point, it gets a reward of 1 and 0 otherwise. By doing so, we have the full knowledge of counterfactual outcomes for evaluation. Data statistics are summarized in Table 4.

Hamming loss, multi-label soft margin loss and reward are reported in Table 5, 6 and 7 respectively. CST-RI generally achieves comparable or superior performance against all baselines in all three datasets. Since we assume we do not know the logging policy, BanditNet performs poorly in dataset like Scene, which is consistent with Kang et al. (2007). HSIC has a comparable performance with CST-RI on TMC and Yeast, but performs poorly on Scene. We suspect this is due to the bias introduced in cross-validation which in turn results in a sub-optimal hyperparameter selection. Overall, CST-RI shows the most robust performance across all three metrics being studied.

5.3 Running Time Analysis

We compare the average running time for one repetition for each experiment under same number of epochs. The results are summarized in Table 8 and 9. Unsurprisingly, DM is the fastest algorithm.

	# features	# labels	train size	test size
Yeast	103	14	1500	917
Scene	294	6	1211	1196
TMC	30438	22	21519	7077

Table 4: Dataset Statistics

	TMC	Yeast	Scene
Direct Method	0.1257±0.0187	0.2992±0.0022	0.1774±0.0010
CST-RI	**0.0772**±0.0043	0.2629±0.0062	**0.1358**±0.0005
CST-AI	0.4593±0.0503	0.4516±0.0906	0.3977±0.0680
HSIC	0.0829±0.0029	**0.2506**±0.0067	0.2869±0.0611

Table 5: Hamming Loss on Real Datasets

	TMC	Yeast	Scene
Direct Method	2.0001±0.1124	7.1978±0.0204	3.8722±0.0966
CST-RI	0.2322±0.0131	**0.5565**±0.0011	**0.3802**±0.0085
CST-AI	2.9240±0.4491	1.7463±0.3652	1.7244±0.3293
HSIC	**0.2310**±0.0053	0.5739±0.0014	0.6024±0.0721

Table 6: Multi-Label Soft-Margin Loss on Real Datasets

	TMC	Yeast	Scene
Direct Method	3931.4±147.1	494.6±3.3	602.2±5.1
CST-RI	**5063**±119.6	**686.2**±0.7	**747.8**±4.7
CST-AI	468.4±92.6	349±167.9	239.2±36.4
HSIC	4719.0±115.1	661.4±15.6	310.2±38.1
BanditNet	4246.4±74.0	661.2±22.2	414.6±190.4

Table 7: Reward on Real Datasets

While our method is almost twice as slow as DM, it is still relatively fast compared to the other baselines. BanditNet is relatively slow due to the cross validation selection. Note that the time efficiency of HSIC is bottlenecked by its high computational complexity (Lopez et al., 2020). We thus observe HSIC is approximately 30 to 100 times slower than CST across all datasets. Since CST offers a competitive performance against HSIC with a much faster running time, it is potentially more suitable for large-scale applications.

	D1	D2	D3	D4	D5
Direct Method	21.64±1.46	21.34±1.81	21.70±1.71	21.44±1.66	21.67±1.18
CST-RI	38.28±1.23	35.30±0.02	36.65±0.27	37.19±0.33	37.71±0.80
CST-AI	38.00±1.14	35.90±0.04	36.99±0.26	38.10±0.54	37.89±0.33
HSIC	4038.35±4.65	4029.02±0.51	4087.17±1.36	4060.01±0.79	4055.30±1.11
BanditNet	657.82±10.50	614.91±1.99	632.93±2.77	639.83±1.53	659.10±3.11

Table 8: Running Time on Synthetic Datasets (measured in seconds)

	TMC	Yeast	Scene
Direct Method	570.63±0.87	33.43±2.95	26.88±0.29
CST-RI	1428.65±2.22	93.60±4.14	71.98±0.69
CST-AI	1441.3±3.00	95.11±4.54	72.74±0.59
HSIC	43806.3±216.05	3276.82±46.25	3299.99±18.90
BanditNet	10342.65±35.83	488.71±35.84	271.22±8.79

Table 9: Running Time on Real Datasets (measured in seconds)

6 Conclusion and Future Work

In this paper, we proposed a novel counterfactual self-training algorithm for learning from observational data. Comparing to existing approaches, our method is easy to compute and optimize. It also does not have the need for hyperparameter selection through cross validation, which is biased in nature for observational data. We provided a theoretical analysis showing self-training objective serves as an upper bound of the true risk of a randomized trial.

While this analysis is tailored for counterfactual learning, we hope it can shed light on a broader range of problems such as unsupervised domain adaptation and semi-supervised learning. It may also open doors for solving counterfactual learning with a model-based extrapolation for direct method. As shown in our pricing example, a good demand model may not necessarily lead to the highest revenue because of the downstream revenue maximization optimization (Elmachtoub & Grigas, 2017). A different formulation of target domain may help address this problem, which we leave for future work. Moreover, we believe our counterfactual self-training framework can be adapted to yield many specific algorithms for tasks such as learning from observational data with structured reward (Lopez et al., 2020; Kallus, 2019) and deficient historical logging policy (Sachdeva et al., 2020).

REFERENCES

Massih-Reza Amini and Patrick Gallinari. Semi-supervised logistic regression. 2002.

Dimitris Bertsimas and Nathan Kallus. The power and limits of predictive approaches to observational-data-driven optimization. *arXiv preprint arXiv:1605.02347*, 2016.

Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771, 2004.

Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

Miroslav Dudík, Dumitru Erhan, John Langford, Lihong Li, et al. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4):485–511, 2014.

André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*, pp. 681–687, 2002.

Adam N Elmachtoub and Paul Grigas. Smart" predict, then optimize". *arXiv preprint arXiv:1710.08005*, 2017.

Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pp. 529–536, 2005.

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530, 2004.

Arthur Gretton, Kenji Fukumizu, Choon H Teo, Le Song, Bernhard Schölkopf, and Alex J Smola. A kernel statistical test of independence. In *Advances in neural information processing systems*, pp. 585–592, 2008.

Ligong Han, Yang Zou, Ruijiang Gao, Lezi Wang, and Dimitris Metaxas. Unsupervised domain adaptation via calibrating uncertainties. In *CVPR Workshops*, volume 9, 2019.

Jennifer L Hill. Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics*, 20(1):217–240, 2011.

Guido W Imbens and Jeffrey M Wooldridge. Recent developments in the econometrics of program evaluation. *Journal of economic literature*, 47(1):5–86, 2009.

Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. Deep learning with logged bandit feedback. In *International Conference on Learning Representations*, 2018.

Fredrik Johansson, Uri Shalit, and David Sontag. Learning representations for counterfactual inference. In *International conference on machine learning*, pp. 3020–3029, 2016.

Nathan Kallus. Classifying treatment responders under causal effect monotonicity. *arXiv preprint arXiv:1902.05482*, 2019.

Nathan Kallus and Angela Zhou. Confounding-robust policy improvement. In *Advances in neural information processing systems*, pp. 9269–9279, 2018.

Joseph DY Kang, Joseph L Schafer, et al. Demystifying double robustness: A comparison of alternative strategies for estimating a population mean from incomplete data. *Statistical science*, 22 (4):523–539, 2007.

A Gürhan Kök and Marshall L Fisher. Demand estimation and assortment optimization under substitution: Methodology and application. *Operations Research*, 55(6):1001–1021, 2007.

Damien Lefortier, Adith Swaminathan, Xiaotao Gu, Thorsten Joachims, and Maarten de Rijke. Large-scale validation of counterfactual learning methods: A test-bed. *arXiv preprint arXiv:1612.00367*, 2016.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pp. 661–670, 2010.

Yuanqing Li, Cuntai Guan, Huiqi Li, and Zhengyang Chin. A self-training semi-supervised svm algorithm and its application in an eeg-based brain computer interface speller system. *Pattern Recognition Letters*, 29(9):1285–1294, 2008.

Romain Lopez, Chenchen Li, Xiang Yan, and Junwu Xiong. Cost-effective incentive allocation via structured counterfactual inference. 2020.

Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740*, 2009.

Jeffrey I McGill and Garrett J Van Ryzin. Revenue management: Research overview and prospects. *Transportation science*, 33(2):233–256, 1999.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.

Judea Pearl. Detecting latent heterogeneity. *Sociological Methods & Research*, 46(3):370–389, 2017.

Judea Pearl et al. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 2000.

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

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

Donald B Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100(469):322–331, 2005.

Noveen Sachdeva, Yi Su, and Thorsten Joachims. Off-policy bandits with deficient support. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 965–975, 2020.

Uri Shalit, Fredrik D Johansson, and David Sontag. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*, pp. 3076–3085, 2017.

Ilya Shpitser and Judea Pearl. Identification of conditional interventional distributions. *arXiv preprint arXiv:1206.6876*, 2012.

Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*, pp. 814–823, 2015a.

Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. In *advances in neural information processing systems*, pp. 3231–3239, 2015b.

Jafar Tanha, Maarten van Someren, and Hamideh Afsarmanesh. Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics*, 8(1):355–370, 2017.

Terence J Wales and Alan Donald Woodland. Estimation of consumer demand systems with binding non-negativity constraints. *Journal of Econometrics*, 21(3):263–285, 1983.

Yanbo Xu, Yanxun Xu, and Suchi Saria. A bayesian nonparametric approach for estimating individualized treatment-response curves. In *Machine Learning for Healthcare Conference*, pp. 282–300, 2016.

Jinsung Yoon, James Jordon, and Mihaela van der Schaar. Ganite: Estimation of individualized treatment effects using generative adversarial nets. In *International Conference on Learning Representations*, 2018.

Yang Zou, Zhiding Yu, BVK Kumar, and Jinsong Wang. Domain adaptation for semantic segmentation via class-balanced self-training. *arXiv preprint arXiv:1810.07911*, 2018.

Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5982–5991, 2019.

A Appendix

A.1 Proof of Theorem 1

Theorem 1. *Assume $\mathbb{P}_\theta(r|x,p) \geq \frac{1}{M_0+1}$, where $M_0 > 1$ is constant, let $M = \min\{\max(\frac{\mathbb{P}}{\mathbb{P}_\theta} - 1), M_0\}$, $f^\star = \operatorname{argmin}_{f\in\mathcal{F}} \mathcal{R}_\mathcal{D}(f)$, $\hat{\mathcal{D}}$ is the dataset generated by random imputation of current model output $\mathbb{P}_\theta$, and $\hat{f}$ minimizes the empirical risk on $\hat{D}$. For any loss function $l(\cdot,\cdot)$, we have:*

$$\mathcal{R}_\mathcal{D}(\hat{f}) - \mathcal{R}_\mathcal{D}(f^\star) \leq C(\sqrt{\frac{V}{n}} + \sqrt{\frac{\log(1/\delta)}{n}}) + (M+1)\hat{\mathcal{R}}_{\hat{\mathcal{D}}}(\hat{f}) - \mathcal{R}_\mathcal{D}(f^\star) \tag{2}$$

Proof.

$$\begin{aligned}
&\mathcal{R}_\mathcal{D}(\hat{f}) - \mathcal{R}_\mathcal{D}(f^\star) \\
=&\mathcal{R}_{\hat{\mathcal{D}}}(\hat{f}) - \mathcal{R}_{\hat{\mathcal{D}}}(f^\star) + \mathcal{R}_\mathcal{D}(\hat{f}) - \mathcal{R}_{\hat{\mathcal{D}}}(\hat{f}) - (\mathcal{R}_\mathcal{D}(f^\star) - \mathcal{R}_{\hat{\mathcal{D}}}(f^\star)) && (4)\\
=&\mathcal{R}_{\hat{\mathcal{D}}}(\hat{f}) - \mathcal{R}_{\hat{\mathcal{D}}}(f^\star) + \mathbb{E}_{\hat{\mathcal{D}}}(\frac{\mathbb{P}(r|x,p)}{\mathbb{P}_\theta(r|x,p)} - 1)(l(\hat{f}(x,p),r) - l(f^\star(x,p),r)) && (5)\\
\leq&\mathcal{R}_{\hat{\mathcal{D}}}(\hat{f}) - \mathcal{R}_{\hat{\mathcal{D}}}(f^\star) + M\mathbb{E}_{\hat{\mathcal{D}}} l(\hat{f}(x,p),r) + \mathbb{E}_{\hat{\mathcal{D}}} l(f^\star(x,p),r) - \mathcal{R}_\mathcal{D}(f^\star) && (6)\\
=&(M+1)(\mathcal{R}_{\hat{\mathcal{D}}}(\hat{f}) - \hat{\mathcal{R}}_{\hat{\mathcal{D}}}(\hat{f})) + (M+1)\hat{\mathcal{R}}_{\hat{\mathcal{D}}}(\hat{f}) - \mathcal{R}_\mathcal{D}(f^\star) && (7)\\
\leq&C(\sqrt{\frac{V}{n}} + \sqrt{\frac{\log(1/\delta)}{n}}) + (M+1)\hat{\mathcal{R}}_{\hat{\mathcal{D}}}(\hat{f}) - \mathcal{R}_\mathcal{D}(f^\star) && (8)
\end{aligned}$$

Equation 4 comes from adding and subtracting the risk defined on $\mathcal{D}$ and $\hat{\mathcal{D}}$. Since $\hat{\mathcal{D}}$ is imputed with probability $\mathbb{P}_\theta$, we can use importance sampling to get Equation 5. We get Equation 7 by adding and subtracting the empirical risk. Equation 8 is from the basic excess-risk bound. V is the VC dimension of hypothesis class $\mathcal{F}$, and C is a universal constant. □

A.2 Data Generation for Synthetic Dataset

In the synthetic experiments, we use a pricing example similar to the experiment in Lopez et al. (2020). Let $U(\cdot,\cdot)$ be a uniform distribution. Assume customer features are a 50-dimensional vector X drawn from $U(0,1)^{50}$ and there are 10 price options from \$1 to \$10. The logging policy is set as $\pi(p=i|x) = \frac{x_i}{\sum_{i=1}^{10} x_i}$. σ denotes sigmoid function. We simulated five types of demand functions, with $h(x) = \sum a_i \sum \exp(\sum b_j \|x_j - c_j\|)$, $a, b, c \sim U(0,1)^{50}$, $r \in \{0,1\}$:

- $r \sim \sigma(h(x) - 2x_0 \cdot p)$
- $r \sim \sigma(5 \cdot (x_0 - 0.5) - 0.4 \cdot p)$
- $r \sim \sigma(h(x) - \text{stepwise1}(x_0) \cdot p)$
- $r \sim \sigma(h(x) - \text{stepwise2}(x_0, x_1) \cdot p)$
- $r \sim \sigma(h(x) - (x_0 + x_1) \cdot p)$

where the stepwise function is defined as:

$$stepwise1(x) = \begin{cases} 0.7, & \text{if } x \leq 0.1 \\ 0.5, & \text{if } 0.1 < x \leq 0.3 \\ 0.3, & \text{if } 0.3 < x \leq 0.6 \\ 0.1, & \text{if } 0.6 < x \leq 1 \end{cases} \tag{9}$$

$$stepwise2(x, y) = \begin{cases} 0.65, & \text{if } x \leq 0.1 \text{ and } y > 0.5 \\ 0.45, & \text{if } x \leq 0.1 \text{ and } y \leq 0.5 \\ 0.55, & \text{if } 0.1 < x \leq 0.3 \text{ and } y > 0.5 \\ 0.35, & \text{if } 0.1 < x \leq 0.1 \text{ and } y \leq 0.5 \\ 0.45, & \text{if } 0.3 < x \leq 0.6 \text{ and } y > 0.5 \\ 0.25, & \text{if } 0.3 < x \leq 0.6 \text{ and } y \leq 0.5 \\ 0.35, & \text{if } 0.6 < x \leq 1 \text{ and } y > 0.5 \\ 0.15, & \text{if } 0.6 < x \leq 1 \text{ and } y \leq 0.5 \end{cases} \tag{10}$$