Dynamic Inter-treatment Information Sharing for Heterogeneous Treatment Effects Estimation

A Preprint

Vinod Kumar Chauhan, Jiandong Zhou, Soheila Molaei, Ghadeer Ghosheh and David A. Clifton
Institute of Biomedical Engineering, University of Oxford, UK
{vinod.kumar,jiandong.zhou,soheila.molaei,ghadeer.ghosheh,david.clifton}@eng.ox.ac.uk

May 26, 2023

Abstract

Existing heterogeneous treatment effects learners, also known as conditional average treatment effects (CATE) learners, lack a general mechanism for end-to-end inter-treatment information sharing, and data have to be split among potential outcome functions to train CATE learners which can lead to biased estimates with limited observational datasets. To address this issue, we propose a novel deep learning-based framework to train CATE learners that facilitates dynamic end-to-end information sharing among treatment groups. The framework is based on *soft weight sharing* of *hypernetworks*, which offers advantages such as parameter efficiency, faster training, and improved results. The proposed framework complements existing CATE learners and introduces a new class of uncertainty-aware CATE learners that we refer to as *HyperCATE*. We develop HyperCATE versions of commonly used CATE learners and evaluate them on IHDP, ACIC-2016, and Twins benchmarks. Our experimental results show that the proposed framework improves the CATE estimation error via counterfactual inference, with increasing effectiveness for smaller datasets.

Keywords conditional average treatment effects, deep learning, information sharing, hypernetworks, transfer learning.

1 Introduction

While randomized controlled trials (RCT) are the gold standard to ascertain the safety and effectiveness of interventions (like policy actions and medical treatments) [31], they suffer from being expensive, time-consuming, and having less representative samples etc. [3]. Real-world evidence generated from real-world data, i.e., observational data, can complement the existing knowledge as well as address the limitations of RCT, e.g., conditional average treatment effect (CATE) estimation for personalized treatment recommendations from electronic health records. CATE is a fundamental problem in *causal inference* that requires predicting counterfactual outcomes [34] for making personalized decisions across various domains. For instance, deciding on the optimal treatment for a patient requires estimating patient outcomes under all treatments [2], making CATE a crucial aspect of decision-making in healthcare.

Recently, CATE estimation from real-world data has seen great attention from the machine learning community, resulting in the development of a large number of estimation techniques for the personalized treatments [24, 8, 9, 6]. The CATE estimation problem is different and more complex from the standard supervised learning [34, 31, 41] due to *the fundamental problem of causal inference* of having only factual outcomes and missing counterfactual outcomes for each unit (such as a patient in healthcare) [16] that complicate shared learning among treatment groups as compared with the supervised learning [4]. The primary differences in design choices across the various proposed techniques consider modeling potential outcome functions and addressing the confounding (and resulting selection bias) that exists in the real-world data [9, 8, 23, 24, 37, 20, 44, 6, 41]. To mitigate the effects of selection bias, the existing literature predominantly uses several estimation techniques for the personalized treatment recommendations, including meta-learners (different from meta-learning referring to 'learning to learn') [24, 20, 30, 8] and representation learning-based networks [19, 8, 37, 6].

Nonetheless, for accurate CATE estimation, the training of CATE learners is dependent on the availability of large enough real-world datasets [4]. In domains like healthcare, it can be very hard to gather enough data for rare diseases as only a few patients are available per year [42]. Furthermore, at the onset of pandemics, such as COVID-19, scarcity of data limits the ability to understand the efficacy of interventions [32]. Not only that but as compared with the predictive machine learning setting, the available data have to be split into treatment groups (such as control and treatment in binary treatment setting) further reducing the available data for each treatment group to get personalized treatment recommendations. This is because the existing CATE learners lack a general mechanism for end-to-end shared training. For example, meta-learners train multiple models independently, such as T-Learner trains two models for each treatment group in the binary treatment setting [24], while the representation learning-based techniques learn shared representations which are used by treatment-specific layers. Therefore, it becomes essential to develop a general framework for inter-treatment information sharing to improve CATE estimation on small-scale datasets for reliable personalized treatment recommendations.

In this paper, we address the problem of the lack of a general training framework to share end-to-end information among treatment groups for reliable CATE estimation for personalized treatment recommendations with limited-size real-world datasets. We propose hypernetworks [12] based first deep learning framework for dynamic end-to-end information sharing among the treatment groups to train CATE learners. Hypernetworks, also called as hypernets, are a class of neural networks that generate weights/parameters of another neural network, known as the target network (CATE learner in our case). The training process learns weights of the hypernetwork for data belonging to any treatment group, which then generates weights of different treatment models of a CATE learner, resulting in soft weight sharing among treatment groups (unlike hard-weight sharing achieved through shared layers). The soft weight sharing enables end-to-end sharing of information among treatment groups during each weight update step of training, resulting in dynamic information-sharing among the treatment groups – helping them to improve CATE estimates (for details refer to Section 4). The training of CATE learners using hypernetworks can also be advantageous for faster training and parameter efficiency, allowing the target network to have a smaller number of learnable parameters. This can be particularly useful when working in resource-constrained settings or when dealing with high-dimensional data, however, the focus of current work is only on information sharing.

The proposed framework complements existing CATE learners and introduces a new class of CATE learners, called HyperCATE. This is most suitable to the meta-learner approaches, such as T-Learner [24] which train separate models for each treatment group and do not share information among treatment groups despite solving the same problem. Moreover, representation learning-based learners also benefit from HyperCATE. While representation learning-based learners learn a shared representation, they have separate heads for each treatment [8], limiting them from sharing end-to-end information. By training different treatment heads using hypernetworks, the treatment heads can share information leading to better estimates. Additionally, we propose to use dropout in the hypernetwork that helps to generate multiple sets of weights for a CATE learner making the proposed framework an uncertainty-aware HyperCATE that supports building trustworthy decision systems. Finally, we propose split-head-based hypernetworks to manage the complexity of hypernetworks.

Contributions: The contributions of the paper are as follows: (1) We propose the first general deep learning framework for dynamic end-to-end inter-treatment information sharing for reliable CATE estimation which enables personalized treatment recommendations with limited-size real-world datasets. (2) This work presents a novel application of hypernetworks to train CATE learners where soft weight sharing provides improved CATE estimation. (3) A new class of uncertainty-aware CATE learners, called *HyperCATE*, is proposed by training CATE learners using hypernetworks with dropout. (4) We propose split-head hypernets to manage the complexity of hypernets which could be useful for complex CATE learners. (5) Finally, HyperCATE versions of the commonly used CATE learners are developed and evaluated on IHDP, ACIC-2016 and Twins benchmarks that show improved estimates and better performance with smaller dataset sizes.

2 Problem Setting

The paper addresses the problem of CATE estimation for binary treatments from real-world data using the potential outcomes framework [34] for personalized treatment recommendations. Let $D = \{X_i, T_i, Y_i\}_{i=1}^N$ be a sample of N units, such as patients, taken *i.i.d.* from an unknown distribution $\mathbb{P}$. Here, $X_i \in \mathbb{R}^d$ is a d-dimensional covariate vector, $T_i \in \{0, 1\}$ is a binary treatment variable (with 1 indicating a patient receiving the treatment and 0 not receiving the treatment), and Y_i is the patient's outcome that can be binary or real, depending on the CATE estimation problem.

Using the Neyman-Rubin potential outcomes framework [34], we define $Y_i(1)$ and $Y_i(0)$ as the potential outcomes (PO) when patient i receives treatment ($T_i = 1$) and when they do not ($T_i = 0$), respectively. However, due to *the fundamental problem of causal inference* [16], we only observe the factual outcome for the selected treatment, not the counterfactual outcome for the non-selected treatment. The CATE is defined as:

$$\tau(x) = \mathbb{E}_{\mathbb{P}}\left[Y(1) - Y(0)|X = x\right], \tag{1}$$

where $\mathbb{E}_{\mathbb{P}}$ denotes the expectation with respect to the unknown distribution $\mathbb{P}$.

To estimate CATE from the observational data for personalized treatment recommendations, we rely on standard assumptions of treatment effect estimation [17]. First, we assume that the *Stable Unit Treatment Value Assumption* (SUTVA) holds, which states that the distribution of a patient's outcome depends only on the treatment they received, and not on the treatment assignment of other patients. Second, we assume *ignorability*, i.e., the treatment assignment policy is ignorable given the patient's covariate information X, i.e.,

$$\{Y(0), Y(1)\} \perp T|X, \tag{2}$$

where $\perp$ denotes independence. This assumption is also known as unconfoundedness because it holds if there are no hidden confounders and if additional conditions are satisfied [33]. Finally, we assume that the treatment assignment policy is stochastic and that each patient has a certain probability of receiving treatment, i.e.,

$$0 < \pi(x) < 1, \quad \forall x \in X, \tag{3}$$

where $\pi(x)$ is the probability of a patient with covariate $X = x$ receiving treatment $T = 1$, called as propensity score. Assumptions 2 and 3 together are called *strong ignorability* assumptions [18]. Under these assumptions, the expected potential outcomes and the CATE are identifiable. Specifically, we have

$$\mu_t(x) = \mathbb{E}_{\mathbb{P}}\left[Y|X = x, T = t\right], \tag{4}$$

$$\tau(x) = \mu_1(x) - \mu_0(x), \tag{5}$$

where $\mu_t(x)$ (or also written as $\mu(w_t, x)$ where w_t are parameters) is PO function of a patient with covariate $X = x$ receiving treatment $T = t$.

3 Related Works

Our work is at the intersection of CATE estimation, information sharing and hypernetworks. So, we briefly discuss these topics as follows.

CATE Learners: CATE estimation has received great attention from the machine learning field due to its ability to handle high dimensional data and complex interactions among the features, as well as the availability of large observational/real-world datasets, such as electronic health records. This has resulted in the development of myriad CATE learners [19, 36, 37, 24, 9, 13, 6]. Our discussion will primarily focus on 'meta-learners' [24], (which are distinct from meta-learning referring to 'learning to learn') and representation learning-based CATE learners [19]. This is due to model-agnostic approach, good theoretical properties and performance of meta-learners, and flexibility, expressiveness and popularity of neural networks [8, 7, 4].

Meta-learners, originally proposed by [24], are model-agnostic CATE learners which can work with any machine learning model, and provide a general recipe for estimation. These can be categorized as one-step *plugin learners*, also called as *indirect learners* and two-step *direct learners*. The indirect learners train models for PO functions $\mu_t(x)$ and estimate treatment effect as $\tau(x) = \mu_1(x) - \mu_0(x)$. This category includes S(single)- and T(two)-Learner from [24]. S-Learner augments features with the treatment variable and trains a single model, while T-Learner trains two separate models for each of the PO head. On the other hand, two-step direct learners first train models to estimate nuisance parameters $\eta = \{\mu_0, \mu_1, \pi\}$, followed by training models on the pseudo-outcome Y_η calculated from nuisance parameters η, to directly estimate treatment effect τ. Different direct learners need different nuisance parameters and have different ways of calculating Y_η to estimate τ. The well-known direct learners are X-Learner [24], DR(doubly robust)-Learner [20], PW(propensity weighting)-learner and RA(regression adjustment)-learner [8], and R-Learner [30].

Representation learning based learners use neural networks and multitasking type architecture and, similar to indirect learners, they estimate treatment effect indirectly as $\tau(x) = \mu_1(x) - \mu_0(x)$ by learning $\mu_t(x)$. These learners have shared layers between PO functions $\mu_t(x)$ followed by treatment-specific layers for each $\mu_t(x)$ to get CATE estimates. The joint training of PO functions helps in partial information sharing through the shared layers which learn from the entire data but lack information sharing in PO-specific layers that learn from data corresponding to that treatment group. The most standard architecture and the pioneering work using representation learning is TARNet [19], which is extended in different ways to balance the covariates to reduce selection bias, such as disentangled representations to permit various types of information sharing between propensity score $\pi(x)$ and PO functions $\mu_t(x)$ [14, 8]. The disentangled representations are based on using orthogonality to learn three or five latent factors for instrument, adjustment and confounder variables, e.g., disentangled representations for counterfactual regression (DRCFR) [14] and SNet+ [6].

Information Sharing: To the best of our knowledge, there is no general mechanism for end-to-end inter-treatment information sharing. Meta-learners [24] train models separately for each of the nuisance parameters η, including PO functions μ_t that can access data of corresponding treatment only, and lack any mechanism to share information, except for S-Learner. S-Learner trains a single model and shares information between PO functions μ_t, however, it lacks the flexibility to capture complex treatment effects and is known to perform poorly in high dimensional settings. The representation learning-based learners have partial information sharing as they employ shared layers, however, the PO functions have treatment-specific layers which could access data related to the corresponding treatment only [19, 8]. Transfer learning-based learners allow sharing of information, such as across shared feature spaces [25] and heterogeneous feature spaces [4], however, by definition they consider two related problems or two patient populations for sharing information and do not apply to single problem/population setting. One learner that specifically addresses information sharing is the flexible treatment effect network (FlexTENet) [9] which is based on the inductive biases of having shared structure between PO functions, and have private and shared layers between PO functions.

Hypernetworks: *Hypernetworks*, or hypernets in short, also referred to as hypermodels, meta-networks, or naturally meta-models, is the term coined by [12], although, similar ideas were discussed earlier, e.g., *fast weights* in [35]. These are a class of neural networks which generate the parameters/weights of a target neural network. They are composed of two networks: a primary network (hypernet) and a target network (CATE learner in our case). The hypernet takes an identity/embedding of PO functions of a CATE learner and learns the features of the data, while the CATE learner learns the relationships between those features. Both the networks are trained end-to-end, however, only the hypernet has trainable weights. Training target networks using the hypernets offers many advantages, such as soft weight sharing, parameter efficiency, faster training / transfer learning and development of dynamic neural architectures etc. Hypernetworks have recently emerged as a promising approach for improving the expressiveness and flexibility of deep learning models, and are successful in providing state-of-the-art results across different problem settings, e.g., multitasking [39], continual learning [40], ensemble learning [22], neural architecture search [43], weight pruning [27], hyperparameter optimization [28], Bayesian neural networks [10], generative models [10], and adversarial defence [38] etc. However, to the best of our knowledge, hypernets are not used in treatment effect estimation.

Thus, based on our brief review of the related work, we conclude that we are the first to propose a general framework for end-to-end inter-treatment information sharing for CATE estimation for personalized treatment recommendations. We are also the first to propose an application of hypernets for treatment effects estimation.

4 Dynamic Inter-treatment Information Sharing

In this section, we propose a general framework for dynamic end-to-end information sharing across the potential outcome (PO) functions $(\mu_0(x), \mu_1(x))$. The proposed framework is based on the soft weight sharing[1], achieved with training of CATE learners using hypernets. Suppose, $\mathcal{H}$ is a hypernet with weights ψ that generates weights θ of a CATE learner f_{cate}, i.e., $\theta = \mathcal{H}(\psi, e)$, where e is an embedding vector for PO functions. Since only hypernet weights ψ and not CATE learner weights θ are trainable so an end-to-end training of a CATE learner with a hypernet involves the following optimization problem.

$$\min_{\psi} \quad f_{cate}(\theta) = f_{cate}(\mathcal{H}(\psi, e)). \tag{6}$$

The proposed framework complements the existing CATE learners and provides a general mechanism to share information across the PO functions, and introduces a new class of CATE learners referred to as *HyperCATE* learners. For HyperCATE learners, data belonging to all the treatment groups leads to gradients flow to update the hypernet weights in each training step – resulting in dynamic inter-treatment information sharing during the training of a CATE learner. We develop HyperCATE versions of two popular categories of CATE learners, which have either no information sharing or partial information sharing, and enable them to share end-to-end information between PO functions: meta-learners [24, 8] and representation learning-based learners [36, 6], as given below.

4.1 Dynamic Inter-treatment Information Sharing for Meta-learners

The CATE meta-learners are classified as direct (two-step) and indirect (also called plug-in – one step) learners [8]. Consider neural network (NN) based implementations of nuisance parameters $\eta = (\mu_0(x), \mu_1(x), \pi(x))$ for the meta-learners. The plug-in learners estimate PO functions $\mu_0(x), \mu_1(x)$, and calculate CATE as $\tau(x) = \mu_1(x) - \mu_0(x)$.

[1]This is soft weight sharing because unlike the hard-weight sharing where the PO functions directly share weights, here weights are shared indirectly to generate weights for the PO functions.

S- and T-Learner are two popular indirect learners [24]. S-Learner augments treatment variable t to the context x as (x, t) and trains a single nuisance parameter $\mu_t(x, t)$, resulting in an end-to-end information sharing between $\mu_0(x)$ and $\mu_1(x)$. However, S-Learner lacks flexibility to model complex treatment effects and high dimensional context, and does not scale well. On the other hand, T-Learner trains $\mu_0(x)$ and $\mu_1(x)$ independently, each of which takes data corresponding to that treatment group, and does not share any information – making it the best candidate for the proposed framework. So, next we develop HyperTLearner from T-Learner to elaborate on the proposed framework.

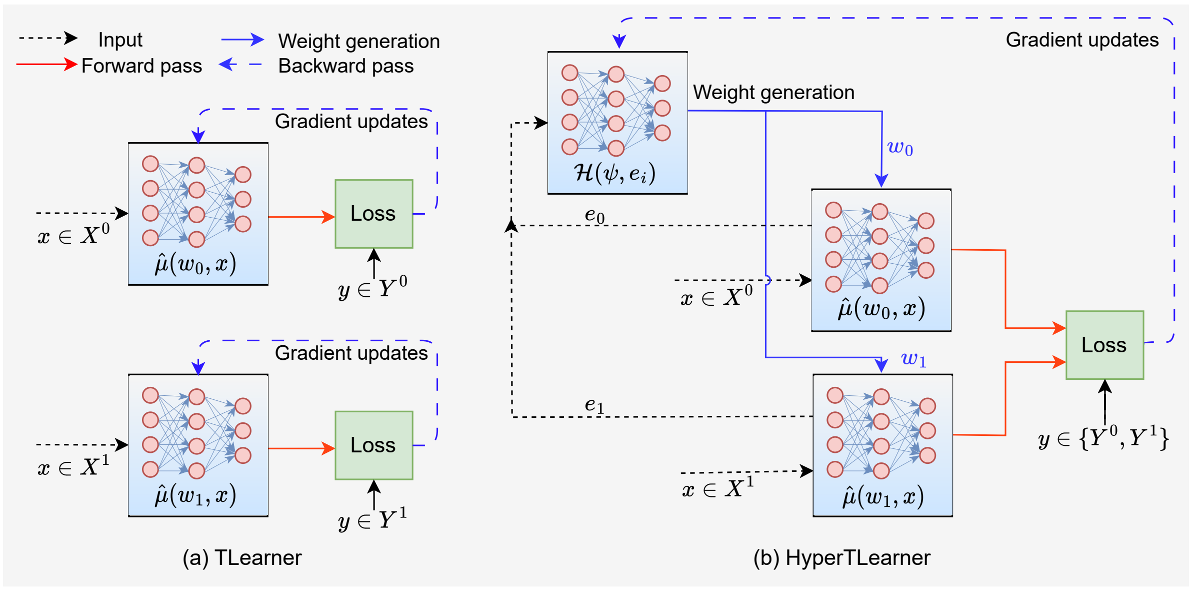


Figure 1: An overview of the architectures and gradient flows for T-Learner and HyperTLearner, where $\hat{\mu}(w_0, x)$ and $\hat{\mu}(w_1, x)$ have exactly the same architecture in both but different training process.

Fig. 1 presents a comparative view of T-Learner and HyperTLearner architectures and flow of gradients. T-Learner splits the dataset $\{x_i^0, y_i^0, x_j^1, y_j^1\}$, having $i = 1, 2, .., m$ and $j = 1, 2, .., n$ factual outcomes in two treatment groups as $\{x_i^0, y_i^0\}$ and $\{x_j^1, y_j^1\}$ for estimating $\mu_0(x)$, and $\mu_1(x)$, respectively. As shown in Fig. 1(a), each PO function has its own gradient flow and independent parameter w_t updates, lacking any information sharing, and making estimation of treatment effects difficult from limited real-world datasets. On the other hand, in HyperTLearner hypernet $\mathcal{H}$ takes an embedding e_t and generates weights w_t for the corresponding PO *functional* $\mu(w_t, x)$, i.e., $w_t = \mathcal{H}(\psi, e_t)$. Next, a forward pass through any of $\mu(w_0, x)$ or $\mu(w_1, x)$ results in gradients flowing back to $\mathcal{H}$ to update hypernet parameters ψ in each step. Thus, soft weight sharing between $\mu_0(x)$ and $\mu_1(x)$ through hypernet parameters ψ facilitates continuous inter-treatment information sharing as ψ are updated using data belonging to both treatment groups, unlike the T-Learner where $\mu_0(x)$ and $\mu_1(x)$ could use data corresponding to that treatment group only. This helps HyperTLearner to estimate CATE from limited datasets.

Direct learners work in two steps. They estimate nuisance parameters $\eta = (\mu_0(x), \mu_1(x), \pi(x))$ from the data in the first step which are then used to calculate pseudo-outcomes Y_η. Y_η is then regressed on the context X to directly estimate $\tau(x)$. There exist several approaches, including propensity weighting (PW), regression adjustment (RA), and doubly robust (DR) meta-learners [8], to calculate pseudo-outcomes Y_η that yield unbiased estimates of CATE as $\mathbb{E}[Y_\eta | X = x]$ when Y_η is known. For example, DR-Learner calculates Y_η using η, as given below, and regresses Y_η on X in second stage to get τ.

$$Y_{DR,\eta} = \left(\frac{T}{\pi(x)} - \frac{1-T}{1-\pi(x)}\right) Y + \left[\left(1 - \frac{T}{\pi(x)}\right)\mu_1 - \left(1 - \frac{1-T}{1-\pi(x)}\right)\mu_0\right]. \tag{7}$$

Similar to HyperTLearner, we can develop HyperDRLearner where one hypernet generates weights for three models $(\mu_0(x), \mu_1(x), \pi(x))$ and facilitate information sharing among them, however, the second stage does not need a hypernet as it trains a single model so the second stage in HyperDRLearner is similar to DRLearner. Similarly, we can develop hypernet variants for the rest of the meta-learners.

4.2 Dynamic Inter-treatment Information Sharing for Representation Learning-based Learners

Shalit et al. [36] pioneered the idea of representation learning to address the selection bias in treatment effect estimation where they presented a simple architecture having shared layers between PO functions, to learn shared representation $\phi(x) : \mathbb{R}^d \rightarrow \mathbb{R}^{d_r}$ from d-dimensional input to d_r-dimensional representation, but treatment-specific layers for $\mu_0(\phi(x)), \mu_1(\phi(x))$, resulting in TARNet architecture. TARNet being a popular and simple representation learning-based learner, we present HyperTARNet in Fig. 2 to illustrate application of our dynamic information sharing framework to representation learning based learners that enables them to share end-to-end information, even in the treatment specific layers.

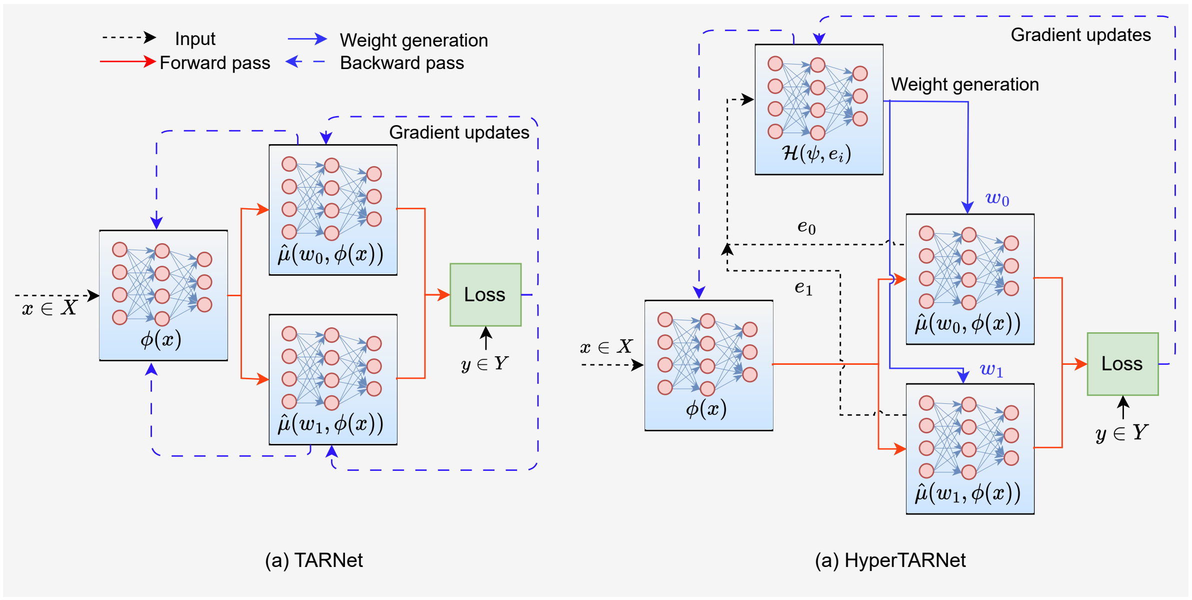


Figure 2: Architectures and gradient flows for TARNet and HyperTARNet, where $\hat{\mu}(w_0, x)$ and $\hat{\mu}(w_1, x)$ have exactly the same architecture in both but different training process. HyperTARNet is employed to train and share information between PO heads, while $\phi(x)$ is learnt similarly to TARNet.

In TARNet, during training, each input x after passing through shared layers $\phi(x)$ passes through treatment specific layer to get either $\mu_0(\phi(x))$ or $\mu_1(\phi(x))$ which along with factual outcome y is used for calculating loss. During the backward pass, gradients flow to $\phi(x)$ through the treatment specific layer $\mu_0(\phi(x))$ or $\mu_1(\phi(x))$, and so weights are updated for $\phi(x)$ and $\mu_0(\phi(x))$ or $\mu_1(\phi(x))$ depending on $x \in X^0$ or $x \in X^1$. On the other hand, in HyperTARNet, during the forward pass, hypernet $\mathcal{H}$ takes an embedding e_t of a PO function and generates weights w_t. Moreover, similar to TARNet, in HyperTARNet context x passes through shared layers to learn a representation $\phi(x)$ which is then passed through PO *functional* depending on the treatment group, i.e., $\mu_0(\phi(x))$ or $\mu_1(\phi(x))$. $\mu_t(\phi(x))$ for treatment t and factual outcome y are used to calculate loss which then backpropagates gradients to hypernet $\mathcal{H}$ for updating the weights ψ. Data corresponding to both treatments groups results in updating of ψ in each training step, and hence dynamically sharing end-to-end information between $\mu_0(x)$, and $\mu_1(x)$, unlike TARNet which has partial information sharing.

There are several extensions of TARNet for handling the selection-bias, such as most recently SNet+ [6]. The proposed HyperCATE framework for training CATE learners can be easily extended to train these representation learning based learners. Thus, the proposed framework complements the existing CATE learners and introduces a new class of HyperCATE learners which facilitate dynamic end-to-end information sharing between treatment groups, helping them to estimate CATE for personalized recommendations from the limited observational data.

Uncertainty-aware HyperCATE: We further propose to use dropout in the hypernetwork which helps to generate multiple sets of weights for a CATE learner, resulting in uncertainty-aware HyperCATE, that helps to build trustworthy decision systems – vital in safety-critical applications, such as healthcare.

Split-head Hypernets: One of the challenge with hypernets is the complexity of the output layer which becomes very large for a large target network and can affect the convergence. Layerwise, weight generation and generation of weights in small chunks [40] are the approaches to reduce the complexity. Here, we propose a split-head approach of weight generation that complements the existing approaches and helps to reduce the complexity and also shows good performance (refer to Appendix A for details of split-head hypernets, and Appendix B.2 for a comparative study of different types of hypernets.)

5 Evaluation

In this section, we present experimental settings and results to prove the efficacy of the proposed framework of dynamic end-to-end information sharing between $\mu_0(x),$ and $\mu_1(x)$.

5.1 Experimental Settings

Benchmarks: We consider two semi-synthetic and one real-world benchmarks: Infant Health and Development Program (*IHDP*) dataset, 2016 Atlantic Causal Inference Conference (*ACIC-2016*) Competition dataset and *Twins* dataset, respectively. IHDP dataset [15, 36] for causal inference is a binary treatment problem which has real covariates, however, outcomes are simulated. It is a small dataset with 139 treated and 608 untreated units (a total of 747 units) and 25 covariates. The child care/specialist home visits are regarded as treatments, the future cognitive test scores of the children are outcomes, while the covariates are features measured from both mother and child. On the other hand, ACIC-2016 dataset [11], initially made available as part of the competition and obtained from the Collaborative Perinatal Project, comprises data for 2200 patients. This dataset contains a total of 55 covariates, including both continuous and categorical variables. We utilized the preprocessing pipeline from the *CATENets* package[2], which results in 4000 and 802 points in train and test sets with 55 features. Twins [1] is a real-world dataset which collects 11,400 twin births in the US between 1989 to 1991. It consists of 39 covariates related to the parents, pregnancy, and birth. Being heavier at birth is treatment and one-year mortality is considered the outcome, i.e., this is a binary treatment problem with binary outcomes. We followed *CATENets* preprocessing and split the data as 50:50 for train and test sets. Moreover, we sample the heavier twin with a probability of 0.1 to have imbalanced data because CATE learners have less variability in the balanced cases.

Baselines: Following [4], we have used DR-Learner, S-Learner, T-Learner and TARNet as our baselines and have compared them with their HyperCATE variants which enable baselines to have dynamic end-to-end inter-treatment information sharing. We have also considered FlexTENet as a baseline because it proposes a flexible architecture for information sharing and related meta-learners such as RA-Learner and X-Learner and one of the recent representation learning-based learners as SNet+. For a fair comparison between CATE and HyperCATE, both have exactly the same architecture and training loop. Moreover, we follow [7] to set basic hyperparameters. All learners have two hidden layers of 100 neurons each, and for the sake of simplicity, hypernets also have two hidden layers of 100 neurons each (for details please refer to Appendix C.). We utilize Precision in the Estimation of Heterogeneous Effects (PEHE) [15] $\sqrt{N^{-1}\sum_{i=1}^{N}\left(\left(\hat{\mu}_1^i-\hat{\mu}_0^i\right)-\left(\mu_1^i-\mu_0^i\right)\right)^2}$ as a metric to evaluate the performance of the CATE learners, as semi-synthetic and real datasets contain both factual and counterfactual outcomes. All the experiments are implemented in Python using PyTorch as a deep learning framework. The experiments are executed on an Ubuntu machine (64GB RAM, one NVIDIA GeForce GPU 8GB) and each experiment is averaged using 10 seeds from 1 to 10 and reports one standard error. Final code will be released on acceptance.

5.2 Results

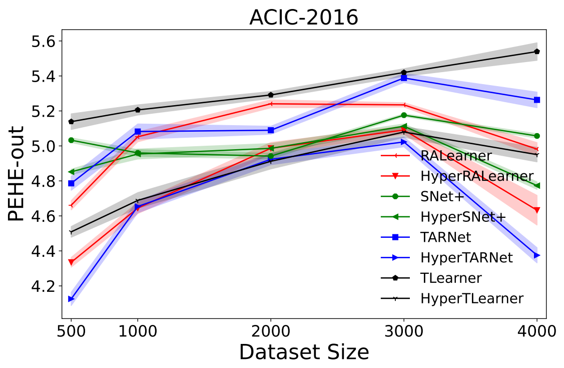
Table 1 presents a comparative study of HyperCATE against the baselines on IHDP, ACIC-2016 and Twins benchmarks using PEHE as a performance metric for CATE estimation. PEHE-in refers to PEHE (in distribution) on train data and PEHE-out (out distribution) refers to PEHE on a held-out test data. On IHDP dataset, all the HyperCATE perform better than the corresponding baselines as well as FlexTENet for both PEHE-in and -out metrics. For ACIC-2016 and Twins benchmarks, HyperCATE performs better than the baselines, except for HyperDRLearner and HyperXLearner, where HyperDRLearner performs better on the out-distribution of data but not on in-distribution. One potential reason for the poor performance of HyperDRLearner and HyperXLearner, which are two-step learners, is the regularization of the second-step models by the errors in the first step, especially by the propensity score $\hat{\pi}$ whose extreme probabilities can even destabilize the training. HyperSLearner is the best performer on IHDP and ACIC-2016, and close to the best performance on Twins dataset, however, the performance improvement is very small. This is because S-Learner already has an end-to-end information sharing for different treatment groups. The performance of CATE learners on PEHE-in and -out show similar patterns. In general, HyperTLearner and HyperRALearner show the highest improvements over the corresponding CATE learner across all the datasets. This is because the corresponding baselines do not have any

[2]https://github.com/AliciaCurth/CATENets

Table 1: Comparative study of CATE learners against HyperCATE learners on IHDP, ACIC-2016 and Twins benchmarks using PEHE (lower is better) as a performance metric. Each experiment is averaged over 10 seeds and the values in parenthesis refer to one standard error.

Learner	**IHDP**		**ACIC-2016**		**Twins**	
	PEHE-in	**PEHE-out**	**PEHE-in**	**PEHE-out**	**PEHE-in**	**PEHE-out**
SLearner	1.02 (0.007)	0.97 (0.013)	3.76 (0.005)	4.00 (0.010)	0.318 (0.001)	0.331 (0.001)
HyperSLearner	**0.99 (0.005)**	**0.95 (0.004)**	**3.75 (0.002)**	**3.96 (0.005)**	**0.309 (0.001)**	**0.324 (0.001)**
TLearner	1.44 (0.012)	1.29 (0.018)	5.37 (0.019)	5.14 (0.045)	0.398 (0.001)	0.410 (0.001)
HyperTLearner	**1.21 (0.022)**	**1.11 (0.015)**	**4.40 (0.053)**	**4.51 (0.031)**	**0.332 (0.003)**	**0.345 (0.003)**
DRLearner	1.20 (0.016)	1.14 (0.013)	**4.28 (0.061)**	4.81 (0.041)	**0.307 (0.001)**	**0.323 (0.001)**
HyperDRLearner	**1.17 (0.017)**	**1.10 (0.017)**	4.42 (0.048)	**4.48 (0.071)**	0.308 (0.001)	**0.323 (0.001)**
RALearner	1.42 (0.014)	1.29 (0.013)	5.22 (0.025)	4.66 (0.026)	0.373 (0.001)	0.384 (0.001)
HyperRALearner	**1.15 (0.015)**	**1.05 (0.012)**	**4.45 (0.054)**	**4.33 (0.028)**	**0.314 (0.001)**	**0.329 (0.001)**
XLearner	1.31 (0.012)	1.20 (0.009)	**4.13 (0.063)**	**4.20 (0.050)**	**0.310 (0.001)**	**0.326 (0.001)**
HyperXLearner	**1.18 (0.027)**	**1.06 (0.023)**	4.52 (0.039)	4.40 (0.042)	0.327 (0.004)	0.341 (0.003)
TARNet	1.37 (0.012)	1.25 (0.014)	4.97 (0.045)	4.79 (0.040)	0.355 (0.001)	0.366 (0.001)
HyperTARNet	**1.16 (0.024)**	**1.09 (0.021)**	**4.06 (0.040)**	**4.13 (0.038)**	**0.324 (0.003)**	**0.338 (0.002)**
SNet+	1.35 (0.010)	1.21 (0.009)	5.48 (0.013)	5.03 (0.013)	0.339 (0.003)	0.353 (0.003)
HyperSNet+	**1.29 (0.013)**	**1.15 (0.007)**	**5.09 (0.020)**	**4.85 (0.015)**	**0.310 (0.001)**	**0.325 (0.001)**
FlexTENet	1.31 (0.004)	1.19 (0.007)	4.86 (0.042)	4.70 (0.043)	0.313 (0.000)	0.328 (0.000)

mechanism to share information between $\mu_0(x)$ and $\mu_1(x)$ while HyperCATE variants provide end-to-end information sharing. On comparing representation learning and meta-learner based HyperCATE, we observe that representation learning-based HyperCATE shows improvements, however, for meta-learners there is not any clear pattern because one side HyperTLearner (direct learner) and HyperRALearner (in-direct learner) show impressive improvements but rest show either small improvements or no improvement.



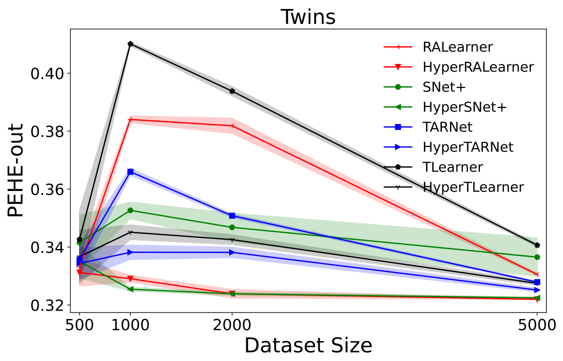


Figure 3: Effect of dataset size on the performance of selected learners using ACIC-2016 and Twins datasets (shaded region shows one standard error).

Effect of dataset size: In Table 1, we studied the performance of HyperCATE with one thousand data points of ACIC-2016 and Twins datasets, to study the effectiveness of HyperCATE for limited real-world data settings. In Fig. 3, we present the effect of dataset size on HyperCATE performance using ACIC-2016 and Twins datasets. For the sake of keeping figures tidy, we keep only those HyperCATEs which showed good results on three datasets in Table 1, and we study dataset size effect on those to find if they show consistent performance (please refer to Appendix B. for the rest of the learners). Moreover, the performance curves are almost similar for PEHE-in and -out so we present results using PEHE-out (please refer to Appendix B. for the rest of the results). From the figure for both the datasets, we find that all HyperCATEs show consistent performance over the corresponding baselines. Moreover, overall the performance curves of learners diverge towards the left side. More specifically, for ACIC-2016, for lower than three thousand data points and for Twins, until one thousand data points, i.e., with decreasing dataset size the performance gap between HyperCATE and baselines increases. This proves effectiveness of HyperCATEs for limited real-world data setting for estimation of CATE for personalized treatment recommendations.

6 Conclusion and Discussion

We proposed *HyperCATE*, a new class of uncertainty-aware CATE learners which complements the existing CATE learners and provides a general framework to train neural network based CATE learners, such as meta-learners and representation learning-based learners, that provides dynamic end-to-end information sharing across treatment groups, unlike the existing learners which lack such a mechanism. HyperCATEs utilize *soft weight sharing* provided by *hypernetworks* for dynamic information sharing across treatment groups. We add a novel application of hypernetworks to the literature and propose the first general mechanism for information across treatment groups for CATE estimation which is also applicable to average treatment effects. The proposed framework enables accurate CATE estimation from limited observational data and is helpful in constrained resource settings for personalized treatment recommendations. We validated the proposed framework with IHDP, ACIC-2016 and Twins benchmarks, and empirically proved their effectiveness for small datasets.

One limitation of HyperCATE is the convergence issue of hypernetworks [5] for generating weights of potential outcome (PO) functions. Similar to multi-tasking [26], hypernets can suffer from conflicting gradients among the treatment groups, unlike standard CATE learners, such as T-Learner which trains PO functions separately and do not face such issues. A proper mechanism for handling the convergence of hypernets is still an open issue, however, adaptive optimizers such as Adam can address the issue to some extent [5]. Another limitation of the proposed mechanism is that it adds extra hyperparameters which have to be tuned for each dataset, although these can be tuned along with other machine-learning hyperparameters on the validation dataset. In addition, although we have experimentally shown the advantages of our HyperCATE learners, it is essential for future research to offer theoretical assurances regarding their performance for meta-learners as well as the representation of learning-based learners. Finally, we acknowledge that causal assumptions may not always hold in real-world scenarios, which can lead to biased results so causal inference models in healthcare can have severe consequences, emphasizing the need for rigorous testing and using them for decision support only in collaboration with clinicians.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

This work was supported in part by the National Institute for Health Research (NIHR) Oxford Biomedical Research Centre (BRC) and in part by InnoHK Project Programme 3.2: Human Intelligence and AI Integration (HIAI) for the Prediction and Intervention of CVDs: Warning System at Hong Kong Centre for Cerebro-cardiovascular Health Engineering (COCHE). DAC was supported by an NIHR Research Professorship, an RAEng Research Chair, the InnoHK Hong Kong Centre for Cerebro-cardiovascular Health Engineering (COCHE), and the Pandemic Sciences Institute at the University of Oxford. The views expressed are those of the authors and not necessarily those of the NHS, the NIHR, the Department of Health, the InnoHK – ITC, or the University of Oxford.

References

[1] Douglas Almond, Kenneth Y Chay, and David S Lee. The costs of low birth weight. *The Quarterly Journal of Economics*, 120(3):1031–1083, 2005.

[2] Susan Athey and Guido Imbens. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27):7353–7360, 2016.

[3] Ioana Bica, Ahmed M Alaa, Craig Lambert, and Mihaela Van Der Schaar. From real-world patient data to individualized treatment effects using machine learning: current and future methods to address underlying challenges. *Clinical Pharmacology & Therapeutics*, 109(1):87–100, 2021.

[4] Ioana Bica and Mihaela van der Schaar. Transfer learning on heterogeneous feature spaces for treatment effects estimation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[5] Oscar Chang, Lampros Flokas, and Hod Lipson. Principled weight initialization for hypernetworks. In *International Conference on Learning Representations*, 2020.

[6] Vinod Kumar Chauhan, Soheila Molaei, Marzia Hoque Tania, Anshul Thakur, Tingting Zhu, and David A. Clifton. Adversarial de-confounding in individualised treatment effects estimation. In *Proceedings of The 26th*

International Conference on Artificial Intelligence and Statistics, volume 206, pages 837–849. PMLR, 25–27 Apr 2023.

[7] Jonathan Crabbé, Alicia Curth, Ioana Bica, and Mihaela van der Schaar. Benchmarking heterogeneous treatment effect models through the lens of interpretability. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[8] Alicia Curth and Mihaela van der Schaar. Nonparametric estimation of heterogeneous treatment effects: From theory to learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, pages 1810–1818. PMLR, 2021.

[9] Alicia Curth and Mihaela van der Schaar. On inductive biases for heterogeneous treatment effect estimation. *Advances in Neural Information Processing Systems*, 34:15883–15894, 2021.

[10] Lior Deutsch, Erik Nijkamp, and Yu Yang. A generative model for sampling high-performance and diverse weights for neural networks. *arXiv preprint arXiv:1905.02898*, 2019.

[11] Vincent Dorie, Jennifer Hill, Uri Shalit, Marc Scott, and Dan Cervone. Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition. *Statistical Science*, 34(1):43–68, 2019.

[12] David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017.

[13] Negar Hassanpour and Russell Greiner. Counterfactual regression with importance sampling weights. In *IJCAI*, pages 5880–5887, 2019.

[14] Negar Hassanpour and Russell Greiner. Learning disentangled representations for counterfactual regression. In *International Conference on Learning Representations*, 2019.

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

[16] Paul W Holland. Statistics and causal inference. *Journal of the American statistical Association*, 81(396):945–960, 1986.

[17] Guido W Imbens and Donald B Rubin. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.

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

[19] Fredrik Johansson, Uri Shalit, and David Sontag. Learning representations for counterfactual inference. In *International Conference on Machine Learning*, pages 3020–3029. PMLR, 2016.

[20] Edward H. Kennedy. Towards optimal doubly robust estimation of heterogeneous causal effects, 2022.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] Agustinus Kristiadi, Sina Däubener, and Asja Fischer. Predictive uncertainty quantification with compound density networks. *arXiv preprint arXiv:1902.01080*, 2019.

[23] Kun Kuang, Peng Cui, Bo Li, Meng Jiang, Yashen Wang, Fei Wu, and Shiqiang Yang. Treatment effect estimation via differentiated confounder balancing and regression. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(1):1–25, 2019.

[24] Sören R Künzel, Jasjeet S Sekhon, Peter J Bickel, and Bin Yu. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the national academy of sciences*, 116(10):4156–4165, 2019.

[25] Sören R Künzel, Bradly C Stadie, Nikita Vemuri, Varsha Ramakrishnan, Jasjeet S Sekhon, and Pieter Abbeel. Transfer learning for estimating causal effects using neural networks. *arXiv preprint arXiv:1808.07804*, 2018.

[26] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021.

[27] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3296–3305, 2019.

[28] Jonathan Lorraine and David Duvenaud. Stochastic hyperparameter optimization through hypernetworks. *arXiv preprint arXiv:1802.09419*, 2018.

[29] Damian Machlanski, Spyridon Samothrakis, and Paul Clarke. Hyperparameter tuning and model evaluation in causal effect estimation. *arXiv preprint arXiv:2303.01412*, 2023.

[30] Xinkun Nie and Stefan Wager. Quasi-oracle estimation of heterogeneous treatment effects. *Biometrika*, 108(2):299–319, 2021.

[31] Judea Pearl. *Causality*. Cambridge university press, 2009.

[32] Zhaozhi Qian, Ahmed M Alaa, Mihaela van der Schaar, and Ari Ercole. Between-centre differences for covid-19 icu mortality from early data in england. *Intensive Care Medicine*, 46:1779–1780, 2020.

[33] 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.

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

[35] Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

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

[37] Claudia Shi, David Blei, and Victor Veitch. Adapting neural networks for the estimation of treatment effects. *Advances in neural information processing systems*, 32, 2019.

[38] Zhun Sun, Mete Ozay, and Takayuki Okatani. Hypernetworks with statistical filtering for defending adversarial examples. *arXiv preprint arXiv:1711.01791*, 2017.

[39] Yi Tay, Zhe Zhao, Dara Bahri, Donald Metzler, and Da-Cheng Juan. Hypergrid transformers: Towards a single model for multiple tasks. In *International Conference on Learning Representations*, 2021.

[40] Johannes von Oswald, Christian Henning, Benjamin F. Grewe, and João Sacramento. Continual learning with hypernetworks. In *International Conference on Learning Representations*, 2020.

[41] Stefan Wager and Susan Athey. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228–1242, 2018.

[42] Jenna Wiens, John Guttag, and Eric Horvitz. A study in transfer learning: leveraging data from multiple hospitals to enhance hospital-specific predictions. *Journal of the American Medical Informatics Association*, 21(4):699–706, 2014.

[43] Chris Zhang, Mengye Ren, and Raquel Urtasun. Graph hypernetworks for neural architecture search. In *International Conference on Learning Representations*, 2019.

[44] Weijia Zhang, Lin Liu, and Jiuyong Li. Treatment effect estimation with disentangled latent factors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10923–10930, 2021.

Appendix

A Split-head Hypernets

Here, we present a novel way of weight generation in hypernets, called Hypernet-Split-head, which splits the head of a standard hypernet (that generates all weights at once and is denoted as *Hypernet-All*). Each of the split heads generates part of the weights of a CATE learner. This helps reduce the complexity of the hypernet as the number of parameters required in the last layer of hypernet, which is the most expensive layer, is reduced by the number of splits. In Fig. 4, we illustrate Hypernet-Split-head applied to Hypernet-All. For the sake of simplicity, we split the head into two parts.

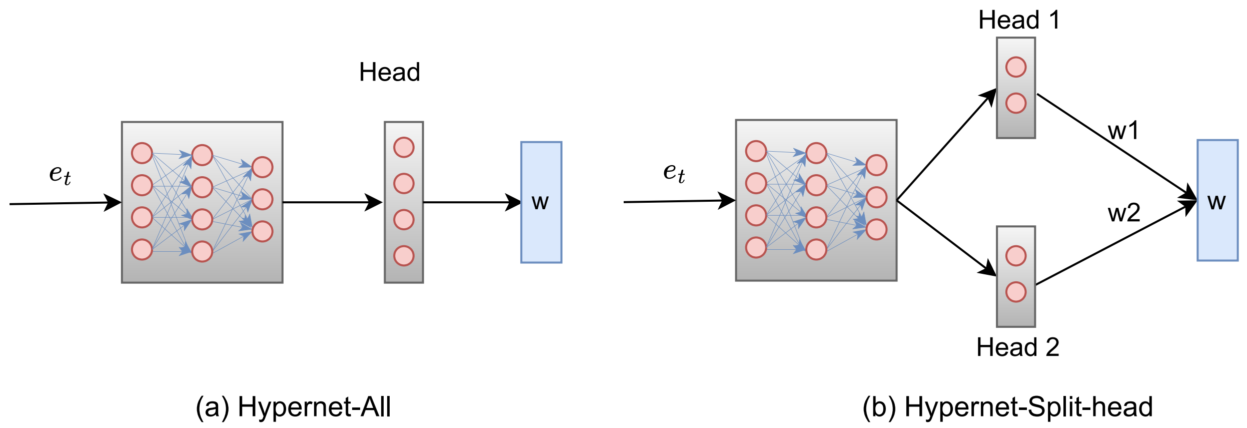


Figure 4: An example of Hypernet-Split-head: (a) Hypernet-All, (b) Hypernet-Split-head as applied to Hypernet-All.

Suppose, the CATE learner has n weights, accordingly, Hypernet-All will have an output layer of n neurons. Furthermore, assume that the immediate previous hidden layer of output has 100 neurons then the output layer would need $n \times 100$ weights (ignoring bias terms). Hypernet-Split-head would split that layer into two parallel layers having 50 neurons, each of which would generate $n/2$ weights of the CATE learner, and overall the output layer would need $n \times 50$. Thus, reducing the number of parameters required in the last layer of the hypernet by two times. This approach of weight generation can be extended to *layerwise* and *chunking* strategies [40]. For a comparative study of the performance of *split-head* hypernet, refer to Appendix B.2.

B Additional Results

Here, we provide some additional results related to effect of embedding size, type of hypernet, and dataset size on the performance of HyperCATE, as discussed below.

B.1 Effect of Embedding Size

An embedding for Potential Outcome (PO) function μ_0 or μ_1, acts as an input to the hypernet which then maps the embedding to weights for the corresponding PO function. Fig. 5 presents the effect of embedding size on the performance of the hypernetworks. We have selected one CATE learner from each category, i.e., TARNet from the representation learning and TLearner from meta-learners based HyperCATE, and considered embedding sizes of 8, 16 and 32, on IHDP, ACIC-2016 and Twins datasets. From the figure, we observe that HyperTARNet has similar performances on all datasets for both PEHE-in and -out, where embedding of size 8 outperforms the rest. For HyperTLearner also the embedding of size 8 outperforms the rest on both datasets, however, the performance is almost similar for embedding sizes 16 and 32.

B.2 Effect of Hypernets Type

Hypernets are the neural networks that generate weights for CATE learners. Depending on the architecture of the CATE learner, the size of the output layer of the hypernet can become very large and affect the convergence of the learning

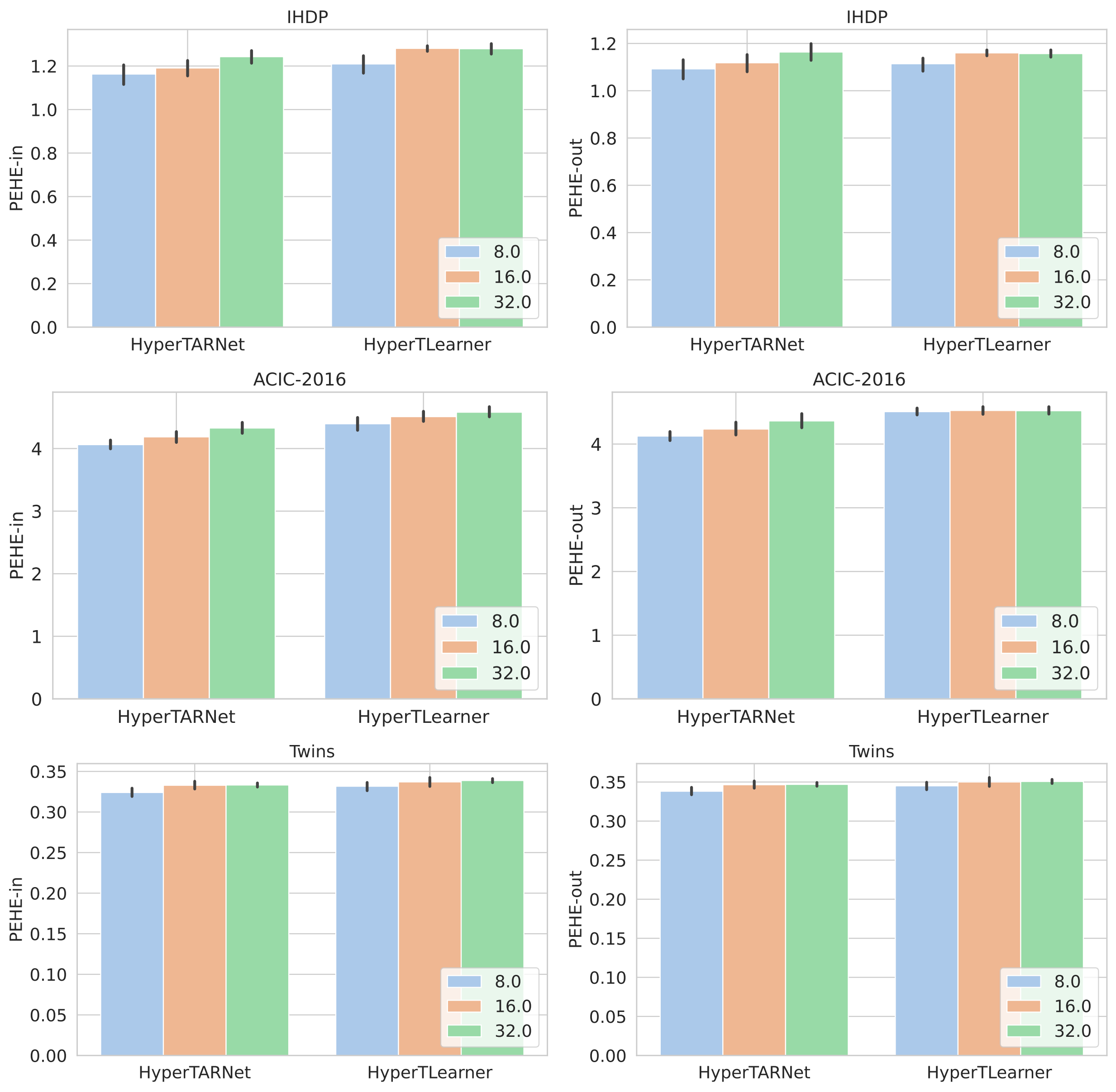


Figure 5: Effect of embedding size on the performance of HyperTARNet and HyperTLearner using PEHE-in and -out performance metrics on IHDP, ACIC-2016 and Twins datasets.

process. To manage the complexity of hypernets, which is one of the challenges, instead of generating *all* weights together (called Hypernet-All), *chunking* (called Hypernet-Chunking) and *layerwise* (called Hypernet-Layerwise) weight generation approaches [40] can be used, i.e., the weights for the target network (CATE learner in our case) are generated in small chunks or weights are generated for one layer at a time. However, each chunk or layer will need an embedding so that hypernet could distinguish among them for weight generation. These three approaches of weight generation have their own advantages and disadvantages, e.g., for Hypernet-All, the complexity of the output layer is highest but its input embedding space is simple and all weights are used and each layer's weight is generated together. On the other hand, Hypernet-Chunking simplifies output space but makes the input space bit complex, and not all weights are used as weights are generated in fixed chunk sizes. Moreover, if the chunk size is smaller than the layer size, then the weights of a layer will not be generated together. Hypernet-Layerwise simplifies output space at the cost of complicating input space of the hypernets. This approach also ends up not using all the generated weights because weights are generated according to the size of the largest layer of a CATE learner, however, weights of each layer are generated altogether. We proposed a split-head approach (called Hypernet-Split-head) for weight generation, which has the best of all approaches as it simplifies the output space without affecting the input space. Moreover, it does not lead to wastage of weights, similar to Hypernet-All. However, one limitation of the split-head approach is that we can't have a large number of splits, otherwise, that will make each head very small.

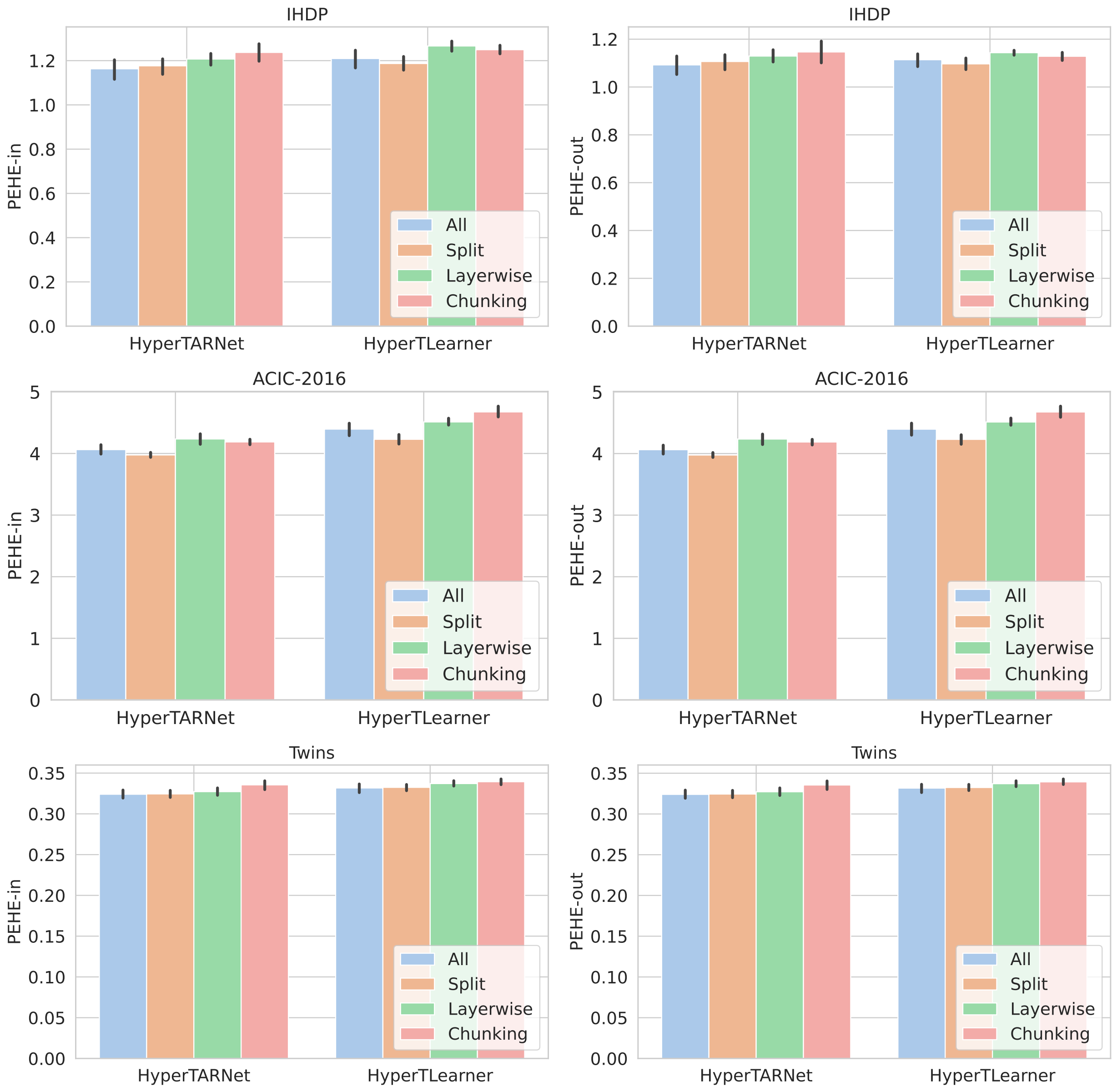


Figure 6: Effect of hypernets type on the performance of HyperTARNet and HyperTLearner using PEHE-in and -out performance metrics on IHDP, ACIC-2016 and Twins datasets.

Fig. 6 studies the effect of the type of hypernets on the performance of HyperCATE on IHDP, ACIC-2016 and Twins datasets with HyperTARNet and HyperTLearner. For Hypernet-Chunking, we have used 10 chunks so Hypernet-Chunking reduces the output layer size by 10 times while increasing the number of embeddings by the same number. However, the effect on output space is huge as compared with input space so we now focus only on output space. Hypernet-Layerwise reduces output space only by 80% and does not have a large drop as in Hypernet-Chunking. This is because of having only two hidden layers in our CATE learners. Split-head approach is applied to Hypernet-All weight generation strategy with two splits so it helps to reduce the number of parameters in the output layer of the hypernet by two times. In terms of performance, from the figure, we observe that Hypernet-All and Hypernet-Split-head outperform Hypernet-Layerwise and Hypernet-Chunking across all the datasets as well for both HyperCATEs. However, there is no clear pattern between Hypernet-Chunking and Hypernet-Layerwise approach of target weight generation in hypernets for the problem under study. Moreover, there is also not a clear winner between Hypernet-Split-head and Hypernet-All as Hypernet-Split-head performs best on ACIC-2016 dataset while Hypernet-All performs best, with a tiny margin against Hypernet-Split-head, on the Twins dataset. While on IHDP dataset, Hypernet-Split-head performs best for HyperTLearner and Hypernet-All performs best for HyperTARNet.

B.3 Effect of Dataset Size

Here, we discuss the effect of dataset size on the performance of the HyperCATE against the baselines for PEHE-in as well as for the learners which were not covered in the main part of the paper and were also not showing consistent performance. Moreover, for the sake of completeness, we re-include the results presented earlier so that all the learners can be studied in one place. We present results in Fig. 7. From the figure, we observe the similar performance of learners on PEHE-in and -out performance metrics, and as discussed earlier, results show that HyperCATE shows improvements over the baselines and results become better with decreasing dataset size. For HyperSLearner on both the datasets, there is not any clear pattern, however, it shows some improvements with smaller sizes. HyperDRLearner does not show any clear pattern, while, as observed earlier, HyperXLearner does not show improvements over X-Learner. Moreover, FlexTENet also does not show consistent results over the two datasets and with a decrease in dataset size the performance drops for Twins and shows unclear performance with ACIC-2016.

C Implementation Details

We follow [7] to set the basic hyperparameters. All learners have two hidden layers of 100 neurons each, and for the sake of simplicity, the hypernet also has two hidden layers of 100 neurons each. Neither can one CATE learner perform well for all the problems [29] nor do we intend to compare them. Moreover, it is difficult to set similar architecture for all the learners. But this does not create any bias in our results because our comparison is between CATE and HyperCATE, and for both cases the learner always has the exact same architecture and training loop but with a different training approach. So, S-Learner and T-Learner have two layers with 100 neurons each, TARNet has one common layer of 100 neurons and each treatment specific head has one hidden layer with 100 neurons. Similarly, for all direct learners, each model in the first stage and stage have two layers with 100 neurons each. For SNet+, we use a hidden layer with 50 neurons in each of the five multi-layer perceptrons (MLPs) to learn latent factors for confounders, instrument and adjustment variables. The treatment-specific layers have one hidden layer with 100 neurons each. For FlexTENet, private and shared layers have two hidden layers with 50 neurons each. All CATE learners use ReLU as an activation function in the hidden layers of CATE learners as well as hypernet. We have used Adam as optimizer [21] with a learning rate of 0.0001, weight-decay of 0.0001, early stopping with the patience of 50 and mini-batch size of 1024. We have used a validation set as 30% of the train set. For uncertainty-aware HyperCATE, we fix dropout of 0.05 in the hypernetwork – a small value is used to avoid disruptive changes in the output weights. Moreover, we use the spectral-norm layer in the hypernet network to stabilize the training. All experiments use Hypernet-All that generates all the weights of a CATE learner at once. The gradient reversal parameter in SNet+ is gradually increased during training by setting γ to 300. For FlexTENet, we set λ_1, λ_2 and orthogonality regularisation factor to 0.0001, 0.01 and 0.1, respectively. We use embeddings of size 1, 8, 8, 8, 8, 16 and 16 for HyperSLearner, HyperTLearner, HyperTARNet, HyperSNet+, HyperRALearner, HyperDRLearner and HyperXLearner, respectively, and are implemented inside the hypernet, i.e., hypernet takes the identity of the PO function and learns an embedding. It is an implementation choice as the embeddings can be put outside the hypernet. In that case, it would take an embedding as an input. Moreover, we use five-fold cross validation for direct learners to have the same number of data points in the second phase and to ensure first models are evaluated on a held-out dataset.

All the experiments are implemented in Python using PyTorch as a deep learning framework. The experiments are executed on an Ubuntu machine (64GB RAM, one NVIDIA GeForce GPU 8GB) and each experiment is averaged using 10 seeds from 1 to 10 and reports one standard error. Final code will be released on acceptance.

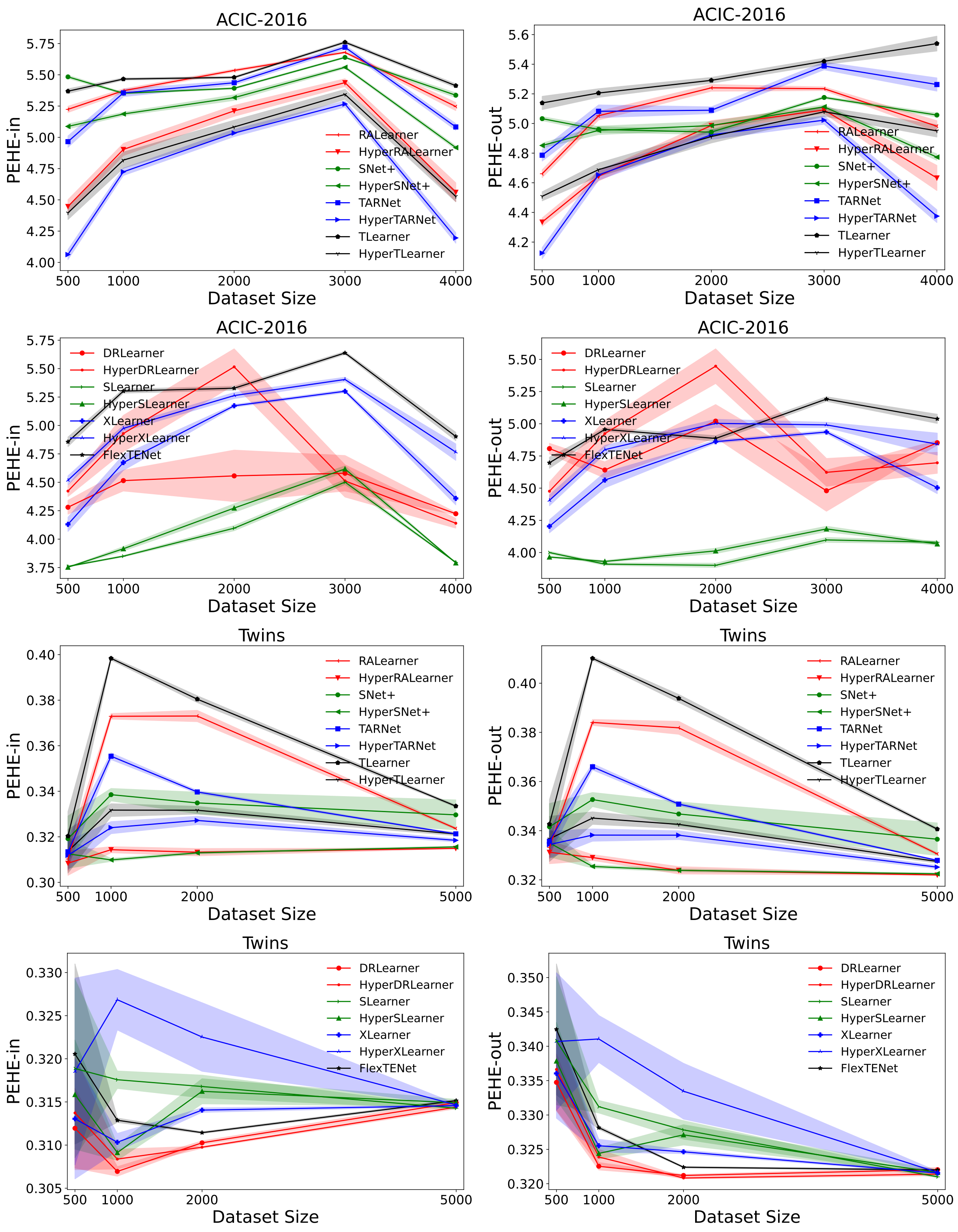


Figure 7: Effect of dataset size on the performance of learners using ACIC-2016 and Twins datasets (shaded region shows one standard error over 10 seeds).