Dynamic Inter-treatment Information Sharing for Individualized Treatment Effects Estimation

Vinod Kumar Chauhan[1] **Jiandong Zhou**[1] **Ghadeer Ghosheh**[1]
Soheila Molaei[1] **David A. Clifton**[1,2]
[1]Institute of Biomedical Engineering, University of Oxford, OX3 7DQ, UK
[2]Oxford-Suzhou Institute of Advanced Research (OSCAR), Suzhou, China
Correspondence: vinod.kumar@eng.ox.ac.uk

Abstract

Estimation of individualized treatment effects (ITE) from observational studies is a fundamental problem in causal inference and holds significant importance across domains, including healthcare. However, limited observational datasets pose challenges in reliable ITE estimation as data have to be split among treatment groups to train an ITE learner. While information sharing among treatment groups can partially alleviate the problem, there is currently no general framework for end-to-end information sharing in ITE estimation. To tackle this problem, we propose a deep learning framework based on '*soft weight sharing*' to train ITE learners, enabling *dynamic end-to-end* information sharing among treatment groups. The proposed framework complements existing ITE learners, and introduces a new class of ITE learners, referred to as *HyperITE*. We extend state-of-the-art ITE learners with *HyperITE* versions and evaluate them on IHDP, ACIC-2016, and Twins benchmarks. Our experimental results show that the proposed framework improves ITE estimation error, with increasing effectiveness for smaller datasets.

1 INTRODUCTION

While randomized controlled trials (RCT) are the gold standard to ascertain safety and effectiveness of interventions (like policy actions and medical treatments)

Pearl (2009), they suffer from several limitations, including being expensive, time-consuming, and ethical challenges Bica et al. (2021). Real-world evidence generated from real-world data, i.e., observational data, can complement existing knowledge as well as address the limitations of RCT, e.g., individualized treatment effects (ITE) estimation for treatment recommendations from electronic health records. ITE is a fundamental problem in *causal inference* that requires predicting counterfactual outcomes Rubin (2005) for making personalized decisions across various domains. For instance, deciding on the optimal treatment for a patient requires estimating patient outcomes under all treatments Athey and Imbens (2016), making ITE a crucial aspect of decision-making in healthcare.

Recently, ITE estimation from real-world data has seen great attention from machine learning community, resulting in development of a large number of estimation techniques for personalized treatments Curth and van der Schaar (2021b,a, 2023a,b); Chauhan et al. (2023a); Xue et al. (2023); Berrevoets et al. (2023); Guo et al. (2023); Wen et al. (2023). ITE estimation problem is different and more complex from standard supervised learning Rubin (2005); Pearl (2009); Wager and Athey (2018). This is 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) Holland (1986). This complicates shared learning among treatment groups as compared with supervised learning Bica and van der Schaar (2022). The primary differences in design choices across the various proposed techniques consider modelling potential outcome functions and addressing the confounding (and resulting selection bias) that exists in the real-world data Curth and van der Schaar (2021a,b); Kuang et al. (2019); Künzel et al. (2019); Shi et al. (2019); Kennedy (2022); Zhang et al. (2021); Chauhan et al. (2023a); Wager and Athey (2018); Nilforoshan et al. (2023). To mitigate the effects of selection bias, the existing litera-

ture predominantly uses several estimation techniques for the personalized treatment recommendations, including meta-learners (different from meta-learning referring to 'learning to learn') Künzel et al. (2019); Kennedy (2022); Nie and Wager (2021); Curth and van der Schaar (2021b) and representation learning-based learners Johansson et al. (2016); Curth and van der Schaar (2021b); Shi et al. (2019); Chauhan et al. (2023a); Guo et al. (2023).

Nonetheless, for accurate ITE estimation, the training of ITE learners is dependent on the availability of large enough real-world datasets Bica and van der Schaar (2022). 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 Wiens et al. (2014). Furthermore, at the onset of pandemics, such as COVID-19, scarcity of data limits the ability to understand the efficacy of interventions Qian et al. (2020). 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 ITE learners have no general mechanism for end-to-end shared training. For example, meta-learners train multiple models independently, such as T-Learner trains two independent models for each treatment group in the binary treatment setting Künzel et al. (2019), while the representation learning-based techniques learn partly shared representations which are used by treatment-specific layers Shi et al. (2019). Since shared learning among related tasks is known to improve performance, including learning to learn setting Aloui et al. (2023); Maurer et al. (2016) so it becomes essential to develop a general framework for inter-treatment information sharing to improve ITE estimation on limited datasets for reliable personalized treatment recommendations. There is some work on transfer learning with homogeneous feature spaces Künzel et al. (2018) and heterogeneous feature spaces Bica and van der Schaar (2022), where information is shared from source domain to target domain in a static way, however, despite some concrete ITE learners, such as Curth and van der Schaar (2021a), there is no general solution for information sharing among treatment groups.

In this paper, we resolve the problem of lack of a general training framework to share end-to-end information among treatment groups, which is crucial for reliable ITE estimation with limited real-world datasets. We employ soft weight sharing of hypernetworks Ha et al. (2017) to propose first deep learning framework for dynamic end-to-end information sharing among the

treatment groups to train ITE 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 (ITE 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 an ITE 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 ITE estimates (for details refer to Section 4). The proposed framework complements existing ITE learners as it provides an alternate way to train, and introduces a new class of ITE learners, called as HyperITE. We extend state-of-the-art ITE learners from meta-learners as well as representation learning based learners to develop HyperITE versions and evaluate on IHDP, ACIC-2016 and Twins benchmarks that show improved estimates and better performance with smaller dataset sizes.

2 BACKGROUND

The paper addresses the problem of ITE estimation for binary treatments from real-world data using potential outcomes framework Rubin (2005) 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 patient's outcome that can be binary or real, depending on the ITE estimation problem.

Using the Neyman-Rubin potential outcomes framework Rubin (2005), we define $Y_i(1)$ and $Y_i(0)$ as potential outcomes (PO) when patient i receives treatment ($T_i = 1$) and when patient does not ($T_i = 0$), respectively. However, due to *the fundamental problem of causal inference* Holland (1986), we only observe the factual outcome for the selected treatment, not the counterfactual outcome for the non-selected treatment, i.e., $Y_i = T_i \times Y_i(1) + (1 - T_i) \times Y_i(0)$. The ITE is defined as follow, which is also referred to as conditional average treatment effects or heterogeneous treatment effects estimation:

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

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

To estimate ITE from the observational data, we rely on standard assumptions of treatment effect estimation Imbens and Rubin (2015), which are required by the baselines. First, we assume that the *Stable Unit Treatment Value Assumption* (SUTVA) holds, which states that each unit's potential outcome is independent of other units' treatment assignments (no interference) and that all units would receive the same treatment if assigned the same treatment assignment (no hidden variations of treatments). 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 Rosenbaum and Rubin (1983). 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 Imbens and Wooldridge (2009). Under these assumptions, the expected potential outcomes and the ITE 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 $\mu(x; w_t)$ where w_t are weights) is a PO function of a patient with covariate $X = x$ belonging to treatment group $T = t$.

3 RELATED WORKS

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

ITE Learners: ITE estimation has received great attention from machine learning field due to its ability to handle high dimensional data and complex interactions among the features, as well as due to availability of large observational/real-world datasets, such as electronic health records. This has resulted in the development of myriad ITE learners Johansson et al. (2016); Shalit et al. (2017); Shi et al. (2019); Künzel et al. (2019); Curth and van der Schaar (2021a); Hassanpour and Greiner (2019a); Tesei et al. (2023); Xue

et al. (2023); Yao et al. (2021). Our discussion will primarily focus on 'meta-learners' Künzel et al. (2019), (which are distinct from meta-learning referring to 'learning to learn') and representation learning-based ITE learners Johansson et al. (2016). This is due to model-agnostic approach, good theoretical properties and performance of meta-learners, and flexibility, expressiveness and popularity of neural networks Curth and van der Schaar (2021b); Chauhan et al. (2022, 2023b, 2024).

Meta-learners, originally proposed by Künzel et al. (2019), are model-agnostic ITE 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 Künzel et al. (2019). S-Learner augments features with the treatment variable and trains a single model, while T-Learner trains two separate models for each of PO function. 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 DR(doubly robust)-Learner Kennedy (2022), RA(regression adjustment)-learner Curth and van der Schaar (2021b), X-Learner Künzel et al. (2019), PW(propensity weighting)-learner, and R-Learner Nie and Wager (2021).

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 ITE 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 Johansson et al. (2016), 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 $\mu_t(x)$ Hassanpour and Greiner (2019b); Curth and van der Schaar (2021b).

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) Hassanpour and Greiner (2019b) and SNet+ Chauhan et al. (2023a).

Information Sharing: To the best of our knowledge, there is no general mechanism for end-to-end inter-treatment information sharing. Meta-learners Künzel et al. (2019) 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, PO functions have treatment-specific layers which could access data related to the corresponding treatment only Johansson et al. (2016); Curth and van der Schaar (2021b). Transfer learning-based learners allow sharing of information, such as across shared feature spaces Künzel et al. (2018) and heterogeneous feature spaces Bica and van der Schaar (2022), 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) Curth and van der Schaar (2021a) which is based on the inductive biases of having shared structure between PO functions, and have private and shared layers between PO functions. Additionally, disentangled variational auto-encoder (tVAE) Xue et al. (2023) is an encoder-decoder based architecture that learns a latent variable where first three elements corresponds to PO functions and propensity score so by architecture have end-to-end information sharing.

Soft Weight Sharing: It is one of the characteristics of hypernetworks arising when there are multiple target networks, like PO functions in our case. *Hypernetworks*, or hypernets in short, also referred to as hypermodels, meta-networks, or naturally meta-models, is the term coined by Ha et al. Ha et al. (2017), although, similar ideas were discussed earlier, e.g., *fast weights* in Schmidhuber (1992). These are a class of neural networks which generate weights of a target neural network. They are composed of two networks: a primary network (hypernet) and a target network (ITE learner in our case). The hypernet takes an identity/embedding of PO functions of an ITE learner and learns to generate weights of ITE learner. Both the networks are trained end-to-end, however, only the hypernet has trainable weights. Since hypernet generates the weights for all potential outcome functions so act as soft weight sharing (unlike hard weight sharing where layers are shared) among PO functions and facilitates information sharing among them. 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, including in multitasking Tay et al. (2021), federated learning Litany et al. (2022), continual learning Oswald et al. (2020), ensemble learning Kristiadi et al. (2019), neural architecture search Peng et al. (2020), camera pose localization Ferens and Keller (2023), weight pruning Liu et al. (2019), hyperparameter optimization Lorraine and Duvenaud (2018), Bayesian neural networks Deutsch et al. (2019), generative models Deutsch et al. (2019), knowledge distillation Wu et al. (2023), neural style transfer Ruta et al. (2023), quantum computing Carrasquilla et al. (2023), density estimation Höfer et al. (2023), the Pareto-front learning Tran et al. (2023), and adversarial defence Sun et al. (2017). However, to the best of our knowledge, soft weight sharing and hypernets are not used in treatment effect estimation.

Thus, based on our brief review of the related work, we conclude that information sharing among treatment groups for ITE estimation remains an open challenge. While there are studies on transfer learning in both heterogeneous and homogeneous feature spaces, the field lacks a general framework for end-to-end information sharing. We propose the first general framework for dynamic end-to-end inter-treatment information sharing for ITE estimation, which similar to transfer learning is crucial for limited observational datasets.

4 DYNAMIC INTER-TREATMENT INFORMATION SHARING

Motivated by the proven benefits of information sharing among related tasks, such as transfer learning Aloui et al. (2023) and learning to learn settings Maurer et al. (2016), and the lack of a general framework for information sharing among treatment groups in treatment effects estimation, we introduce a deep learning framework based on soft weight sharing[1]. This framework enables dynamic information sharing and enhances the reliability of ITE estimation, especially in scenarios with limited observational data. The

---

[1]This is soft weight sharing because unlike the hard-weight sharing where PO functions directly share weights, here weights are shared indirectly to generate weights for PO functions.
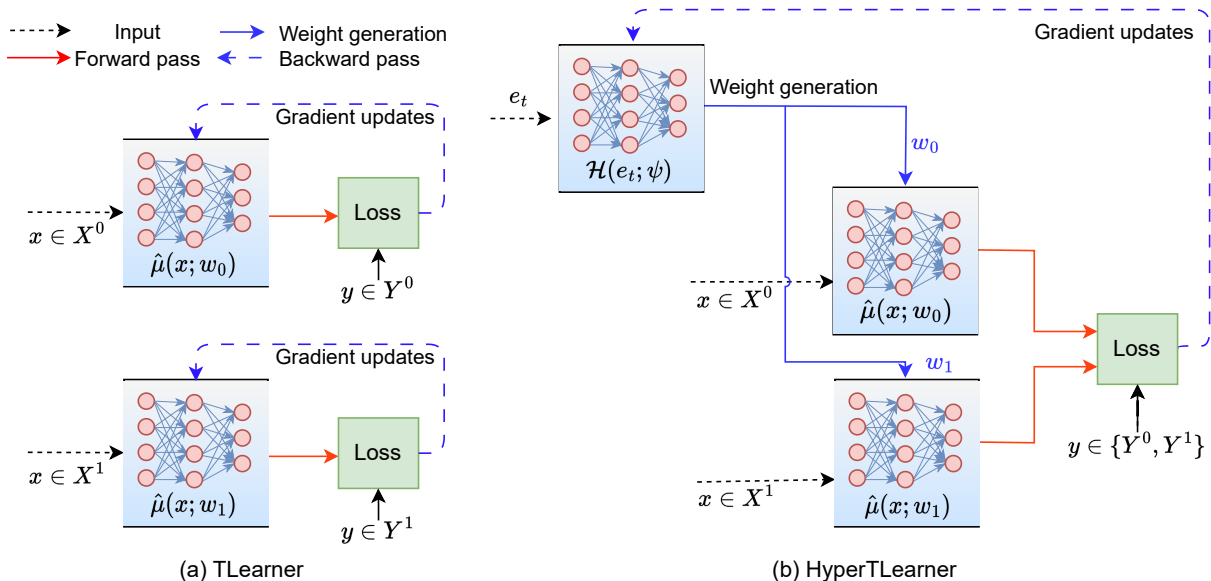
Figure 1: An overview of the architectures and gradient flows for T-Learner and HyperTLearner, where $\hat{\mu}(x; w_0)$ and $\hat{\mu}(x; w_1)$ have exactly the same architecture in both but different training process.

proposed framework provides an alternative way of training existing ITE learners, resulting in a new class of learners called as HyperITE.

Suppose $f_{ITE}(\theta)$ is an ITE learner with weights θ and $(\mu_0(x), \mu_1(x))$ are potential outcome (PO) functions corresponding to treatment $T = 0$ and $T = 1$, respectively. $\mathcal{H}$ is a hypernet with weights ψ that generates weights θ of f_{ITE}, i.e., $\theta = \mathcal{H}(\cdot; \psi)$ by taking an embedding vector e as input corresponding to PO functions. In standard training of an ITE learner, the training process optimizes its weights θ directly from data, while in HyperITE, i.e., an ITE learner trained using hypernets, only hypernet weights ψ and not ITE learner weights θ are trainable. So, optimization problems for ITE and HyperITE are as given below.

$$\begin{aligned}
\text{ITE:} & \quad \min_\theta \quad f_{ITE}(\theta), \\
\text{HyperITE:} & \quad \min_\psi \quad f_{ITE}(\theta = \mathcal{H}(\cdot; \psi)).
\end{aligned} \quad (6)$$

For HyperITE learners, data belonging to all treatment groups lead to gradients flow to update the hypernet weights in each training step – resulting in dynamic inter-treatment information sharing during the training of an ITE learner, as explained in following subsections. We illustrate HyperITE versions of following two categories of ITE learners: meta-learners Künzel et al. (2019); Curth and van der Schaar (2021b) which have no information sharing, and representation learning-based learners Shalit et al. (2017); Chauhan et al. (2023a) that have partial information sharing, and the framework enables them to share end-to-end information between PO functions.

4.1 HyperITE for Meta-learners

To illustrate application of the proposed framework to train existing meta-learners, we take an example of T-Learner due to ease of application and its lack of any information sharing as it trains potential outcome functions $\mu_0(x)$ and $\mu_1(x)$ independently. Fig. 1 presents a comparative view of T-Learner and HyperTLearner, i.e., T-Learner trained with the proposed framework, with architectures and flow of gradients. As shown in Fig. 1(a), T-Learner splits dataset $\{X_i, T_i, Y_i\}$ according to treatment into two groups as $\{X_i^0, T_i = 0, Y_i^0\}$ and $\{X_i^1, T_i = 1, Y_i^1\}$ for estimating $\mu_0(x)$, and $\mu_1(x)$, respectively. Each PO function has its own gradient flow and independent weights w_t updates, lacking any information sharing. On the other hand, in HyperTLearner in Fig. 1(b), the exact same T-Learner is trained using soft weight sharing. Hypernet $\mathcal{H}$ takes an embedding e_t and generates weights w_t for the corresponding PO *functional* $\mu(x; w_t)$, i.e., $w_t = \mathcal{H}(e_t; \psi)$. Next, a forward pass through any of $\mu(x; w_0)$ or $\mu(x; w_1)$ results in gradients flowing back to hypernet $\mathcal{H}$ to update weights ψ in each step. Thus, soft weight sharing between $\mu_0(x)$ and $\mu_1(x)$ through hypernet weights 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. Thus, the training process optimizes hypernet weights ψ so that it generates T-Learner's weights correctly to reduce prediction errors, and resulting
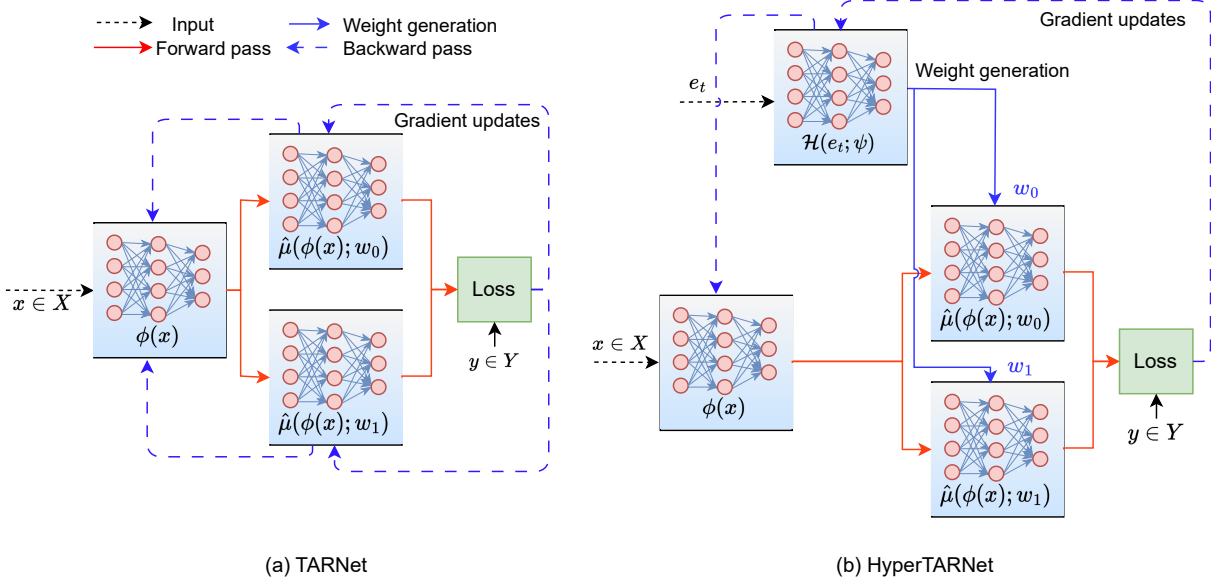
Figure 2: Architectures and gradient flows for TARNet and HyperTARNet, where $\hat{\mu}(x; w_0)$ and $\hat{\mu}(x; w_1)$ have exactly the same architecture in both but different training process. HyperTARNet is employed to train and share information between PO heads, while $\phi(x)$ is learnt similarly to TARNet.

into dynamic end-to-end information sharing between treatment groups.

Using similar approach, we can extend the proposed framework to other meta-learners. For example, DR-Learner Kennedy (2022) is a two-step learner where it estimates nuisance parameters $\eta = (\mu_0(x), \mu_1(x), \pi(x))$ in first step which are then used to calculate pseudo-outcomes Y_η. In second step, Y_η is regressed on the context X to directly estimate treatment effect $\tau(x)$. To develop HyperDRLearner, we use hypernet in the first step to generate three networks corresponding to $(\mu_0(x), \mu_1(x), \pi(x))$, instead of two in HyperTLearner, while the second step remains similar to DR-Learner as it trains a single network.

4.2 HyperITE for Representation Learning-based Learners

Shalit et al. Shalit et al. (2017) pioneered the idea of representation learning to address the selection bias in treatment effect estimation where they presented a simple architecture having shared layers (hard weight sharing) between PO functions, to learn shared representation $\phi(x): \mathbb{R}^d \to \mathbb{R}^{d_r}$ 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. We extend the proposed framework to TARNet due to its popularity and having simple architecture among representation learning based learners, and illustrate differences between TARNet and Hyper-

TARNet in Fig. 2.

As shown in the figure, during forward pass, TARNet learns a shared representation $\phi(x)$ for each input x, which then passes through treatment specific layer to get either $\mu_0(\phi(x))$ or $\mu_1(\phi(x))$. This outcome 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 weights are updated for $\phi(x)$ and $\mu_0(\phi(x))$ or $\mu_1(\phi(x))$ depending on $x \in X^0$ or $x \in X^1$. So, TARNet has partial information sharing due to hard weight sharing through shared layers. On the other hand, in HyperTARNet, during the forward pass, hypernet $\mathcal{H}$ takes an embedding e_t of treatment specific head and generates weights w_t. Moreover, similar to TARNet, HyperTARNet takes an input x to learn a representation $\phi(x)$ which is then passed through treatment specific head, 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+ Chauhan et al. (2023a). Similar to TARNet, the proposed HyperITE framework can be easily extended

to train these representation learning based learners. Please refer to supplementary material for one more example of application of the framework to SNet+ which is relatively a complex learner, as well as details of remaining HyperITE versions.

5 EVALUATION

In this section, we present experimental settings and results to prove 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 Hill (2011); Shalit et al. (2017) 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 Dorie et al. (2019), 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 Almond et al. (2005) 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 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 ITE learners have less variability in the balanced cases.

Baselines: Following Bica and van der Schaar (2022), we have used DR-Learner Kennedy (2022), S-Learner Künzel et al. (2019), T-Learner Künzel et al. (2019) and TARNet Johansson et al. (2016) as our baselines. Additionally, we have also considered Flex-

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

TENet Curth and van der Schaar (2021a) as a baseline because it proposes a flexible architecture for information sharing, RA-Learner Curth and van der Schaar (2021b) due to its suitability to the framework and importance Nilforoshan et al. (2023), and recent representation learning-based learners as SNet+ Chauhan et al. (2023a) and MitNet Guo et al. (2023). We present more results with tVAE Xue et al. (2023) in supplementary material. We have compared the baselines with their HyperITE variants which enable baselines to have dynamic end-to-end inter-treatment information sharing. For a fair comparison between ITE and HyperITE, both have exactly the same architecture and training loop. Moreover, we follow Crabbé et al. (2022) 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 supplementary material Section 2. Implementation Details). We utilize Precision in the Estimation of Heterogeneous Effects (PEHE) Hill (2011) $\sqrt{N^{-1} \sum_{i=1}^{N} \left(\left(\hat{\mu}_1^i - \hat{\mu}_0^i \right) - \left(\mu_1^i - \mu_0^i \right) \right)^2}$ as a metric to evaluate the performance of the ITE learners. This is because semi-synthetic and real datasets contain both factual and counterfactual outcomes, and treatment effects can be calculated. 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 ten runs, except for IHDP where we averaged over 100 runs due to smaller size, and reports one standard error.

5.2 Results

Table 1 presents a comparative study of HyperITE against the baselines on IHDP, ACIC-2016 and Twins benchmarks using PEHE as a performance metric for ITE estimation. Here, FlexTENet does not have a HyperITE version because it has a flexible architecture that provides end-to-end information sharing. 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 HyperITE perform better than the corresponding baselines as well as FlexTENet for both PEHE-in and -out metrics. For ACIC-2016 and Twins benchmarks, HyperITE performs better than the baselines, except for HyperDR-Learner, where HyperDRLearner performs better on the out-distribution of data but not on in-distribution. One potential reason for the poor performance of HyperDRLearner, which is a two-step learner, 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 desta-

Table 1: Comparative study of ITE learners against HyperITE learners on IHDP, ACIC-2016 and Twins benchmarks using PEHE (lower is better) as a performance metric. Each experiment is averaged over multiple runs and 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 (0.001)	0.97 (0.001)	3.76 (0.005)	4.00 (0.010)	0.318 (0.001)	0.331 (0.001)
HyperSLearner	**0.99 (0.001)**	**0.95 (0.001)**	**3.75 (0.002)**	**3.96 (0.005)**	**0.309 (0.001)**	**0.324 (0.001)**
TLearner	1.46 (0.002)	1.32 (0.003)	5.37 (0.019)	5.14 (0.045)	0.398 (0.001)	0.410 (0.001)
HyperTLearner	**1.21 (0.002)**	**1.12 (0.001)**	**4.40 (0.053)**	**4.51 (0.031)**	**0.332 (0.003)**	**0.345 (0.003)**
DRLearner	1.18 (0.002)	1.12 (0.002)	**4.28 (0.061)**	4.81 (0.041)	**0.307 (0.001)**	**0.323 (0.001)**
HyperDRLearner	**1.16 (0.002)**	**1.09 (0.002)**	4.42 (0.048)	**4.48 (0.071)**	0.308 (0.001)	**0.323 (0.001)**
RALearner	1.41 (0.001)	1.25 (0.002)	5.22 (0.025)	4.66 (0.026)	0.373 (0.001)	0.384 (0.001)
HyperRALearner	**1.17 (0.002)**	**1.06 (0.002)**	**4.45 (0.054)**	**4.33 (0.028)**	**0.314 (0.001)**	**0.329 (0.001)**
TARNet	1.36 (0.001)	1.24 (0.002)	4.97 (0.045)	4.79 (0.040)	0.355 (0.001)	0.366 (0.001)
HyperTARNet	**1.15 (0.002)**	**1.08 (0.002)**	**4.06 (0.040)**	**4.13 (0.038)**	**0.324 (0.003)**	**0.338 (0.002)**
MitNet	1.36 (0.004)	1.25 (0.004)	4.93 (0.043)	4.76 (0.043)	0.309 (0.012)	0.324 (0.012)
HyperMitNet	**1.18 (0.009)**	**1.12 (0.005)**	**4.03 (0.041)**	**4.09 (0.041)**	**0.308 (0.011)**	**0.323 (0.012)**
SNet+	1.35 (0.001)	1.22 (0.001)	5.48 (0.013)	5.03 (0.013)	0.339 (0.003)	0.353 (0.003)
HyperSNet+	**1.29 (0.001)**	**1.17 (0.001)**	**5.09 (0.020)**	**4.85 (0.015)**	**0.310 (0.001)**	**0.325 (0.001)**
FlexTENet	1.32 (0.011)	1.19 (0.011)	4.86 (0.042)	4.70 (0.043)	0.313 (0.000)	0.328 (0.000)

bilize 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 trains a single learner. DR-Learner performs the best on Twins dataset and closely followed by HyperDRLearner and HyperMitNet. The performance of ITE learners on PEHE-in and -out show similar patterns. In general, HyperTLearner and HyperRALearner show the highest improvements over the corresponding ITE learner across all the datasets. This is because the corresponding baselines do not have any mechanism to share information between $\mu_0(x)$ and $\mu_1(x)$ while HyperITE variants provide end-to-end information sharing. On comparing representation learning and meta-learner based HyperITE learners, we observe that representation learning-based HyperITE show clear improvements over corresponding baselines, however, for meta-learners one side there are large gains for HyperTLearner (direct learner) and HyperRALearner (in-direct learner) but on the other side HyperDRLearner does not show clear pattern.

Effect of Limited Observational Datasets: In Table 1, we studied the performance of HyperITE with one thousand data points of ACIC-2016 and Twins datasets, to study effectiveness of HyperITE for limited real-world data settings. Fig. 3 depicts effect of dataset size on HyperITE performance using ACIC-2016 and Twins datasets (we can't do similar experiments for IHDP due to its small size). For the sake of keeping figures tidy, we keep only those HyperITEs 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 supplementary material 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 supplementary material for results with PEHE-in).

From Fig. 3 for both the datasets, we find that all HyperITEs show consistent performance over the corresponding baselines. Moreover, overall the performance curves of learners diverge towards the left side (except for Twins at 500 data points), i.e., with decreasing dataset size the performance gap between HyperITE and baselines increases. This proves effectiveness of HyperITEs for limited real-world data setting for estimation of ITE for personalized treatment recommendations. Please refer to supplementary material for additional experiments on effect of embedding size on performance of HyperITE, effect of different weight generation strategies in HyperITE, and convergence of ITE against HyperITE.

6 CONCLUSION

In this paper, we solved an important issue of information sharing among treatment groups in ITE estimation, which is crucial while studying interventional effects on limited observational datasets. We proposed *HyperITE*, a new class of ITE learners, which complements the existing ITE learners and provides a general framework to train neural network based ITE learners, such as meta-learners and representation learning-based learners. The proposed framework provides dynamic end-to-end information sharing across treatment groups, unlike the existing learners which lack such a mechanism. HyperITE utilizes *soft weight*
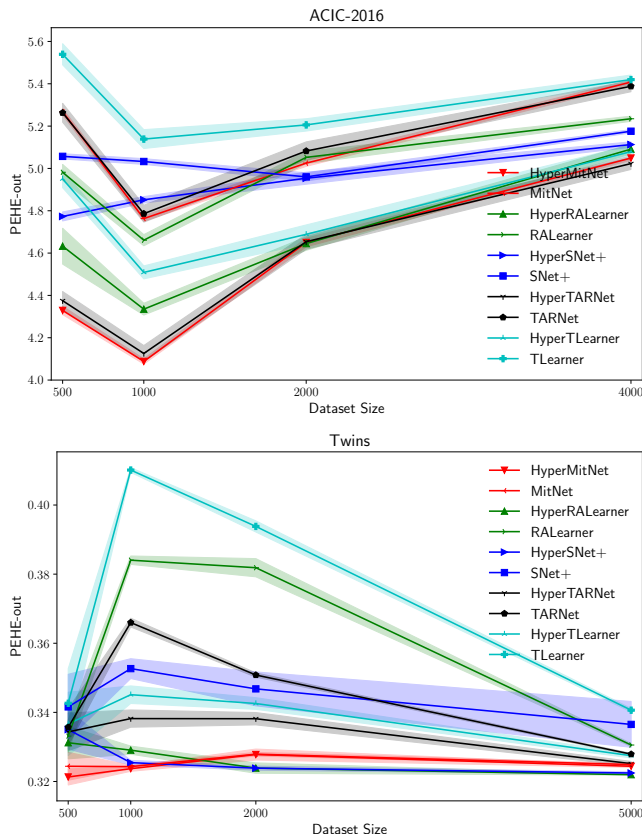
Figure 3: Effect of dataset size on the performance of selected learners using ACIC-2016 and Twins datasets (shaded region shows one standard error).

sharing for dynamic information sharing across treatment groups. We validated the proposed framework with IHDP, ACIC-2016 and Twins benchmarks, and empirically proved their effectiveness, especially for small datasets.

One limitation of HyperITE is the convergence issue of hypernetworks Chang et al. (2020) for generating weights of potential outcome (PO) functions. Similar to multi-tasking Liu et al. (2021), hypernets can suffer from conflicting gradients among the treatment groups, unlike standard ITE 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 Chang et al. (2020). 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. Finally, we acknowledge that causal assumptions may not always hold in real-world scenarios, potentially leading to biased results. There-

fore, it is crucial to validate causal inference models rigorously. These models should be used for decision support only in collaboration with clinicians due to the possible severe consequences in healthcare settings.

Acknowledgments

Code Availability

The code is is available at *https://github.com/ jmdvinodjmd/HyperITE*.

Contributions

V.K.C.: Conceptualization, Methodology, Software and Writing- Original draft preparation. **J.Z.**: Writing - Review & Editing. **G.G.**: Writing - Review & Editing. **S.M.**: Writing - Review & Editing. **D.A.C**: Writing - Review & Editing, Supervision and Funding acquisition.

References

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.

Ahmed Aloui, Juncheng Dong, Cat P Le, and Vahid Tarokh. Transfer learning for individual treatment effect estimation. In *Uncertainty in Artificial Intelligence*, pages 56–66. PMLR, 2023.

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

Jeroen Berrevoets, Fergus Imrie, Trent Kyono, James Jordon, and Mihaela van der Schaar. To impute or not to impute? missing data in treatment effect

estimation. In *International Conference on Artificial Intelligence and Statistics*, pages 3568–3590. PMLR, 2023.

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. URL https://openreview.net/forum?id=nRcyGtY2kBC.

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.

Juan Carrasquilla, Mohamed Hibat-Allah, Estelle Inack, Alireza Makhzani, Kirill Neklyudov, Graham W Taylor, and Giacomo Torlai. Quantum hypernetworks: Training binary neural networks in quantum superposition. *arXiv preprint arXiv:2301.08292*, 2023.

Oscar Chang, Lampros Flokas, and Hod Lipson. Principled weight initialization for hypernetworks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1lma24tPB.

Vinod Kumar Chauhan, Anshul Thakur, Odhran O'Donoghue, Omid Rohanian, and David A Clifton. Continuous patient state attention models. *medRxiv*, pages 2022–12, 2022.

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 2023a.

Vinod Kumar Chauhan, Jiandong Zhou, Ping Lu, Soheila Molaei, and David A Clifton. A brief review of hypernetworks in deep learning. *arXiv preprint arXiv:2306.06955*, 2023b.

Vinod Kumar Chauhan, Sukhdeep Singh, and Anuj Sharma. Hcr-net: A deep learning based script independent handwritten character recognition network. *Multimedia Tools and Applications*, pages 1–35, 2024.

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 Bench-marks Track*, 2022. URL https://openreview.net/forum?id=ddPXQt-gM--.

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

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, 2021b.

Alicia Curth and Mihaela van der Schaar. In search of insights, not magic bullets: Towards demystification of the model selection dilemma in heterogeneous treatment effect estimation. *ICML*, 2023a.

Alicia Curth and Mihaela van der Schaar. Understanding the impact of competing events on heterogeneous treatment effect estimation from time-to-event data. In *International Conference on Artificial Intelligence and Statistics*, pages 7961–7980. PMLR, 2023b.

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.

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.

Ron Ferens and Yosi Keller. Hyperpose: Camera pose localization using attention hypernetworks. *arXiv preprint arXiv:2303.02610*, 2023.

Xingzhuo Guo, Yuchen Zhang, Jianmin Wang, and Mingsheng Long. Estimating heterogeneous treatment effects: Mutual information bounds and learning algorithms. In *ICML*, 2023.

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkpACe1lx.

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

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

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

Timon Höfer, Benjamin Kiefer, Martin Messmer, and Andreas Zell. Hyperposepdf - hypernetworks predicting the probability distribution on so(3). In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2369–2379, January 2023.

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

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

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

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

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

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

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

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.

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.

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.

Or Litany, Haggai Maron, David Acuna, Jan Kautz, Gal Chechik, and Sanja Fidler. Federated learning with heterogeneous architectures using graph hypernetworks. *arXiv preprint arXiv:2201.08459*, 2022.

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.

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.

Jonathan Lorraine and David Duvenaud. Stochastic hyperparameter optimization through hypernetworks, 2018. URL https://openreview.net/forum?id=SJIA6ZWC-.

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

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

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

Hamed Nilforoshan, Michael Moor, Yusuf Roohani, Yining Chen, Anja Šurina, Michihiro Yasunaga, Sara Oblak, and Jure Leskovec. Zero-shot causal learning. *arXiv preprint arXiv:2301.12292*, 2023.

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

Judea Pearl. *Causality.* Cambridge university press, 2009.

Houwen Peng, Hao Du, Hongyuan Yu, Qi Li, Jing Liao, and Jianlong Fu. Cream of the crop: Distilling prioritized paths for one-shot neural architecture search. *Advances in Neural Information Processing Systems*, 33:17955–17964, 2020.

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.

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.

Dan Ruta, Andrew Gilbert, Saeid Motiian, Baldo Faieta, Zhe Lin, and John Collomosse. Hypernst: Hyper-networks for neural style transfer. In Leonid Karlinsky, Tomer Michaeli, and Ko Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 201–217, Cham, 2023. Springer Nature Switzerland.

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

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.

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

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

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. URL https://openreview.net/forum?id=hiq1rHO8pNT.

Gino Tesei, Stefanos Giampanis, Jingpu Shi, and Beau Norgeot. Learning end-to-end patient representations through self-supervised covariate balancing for causal treatment effect estimation. *Journal of Biomedical Informatics*, 140:104339, 2023.

Tuan Anh Tran, Long Phi Hoang, Dung Duy Le, and Thang Ngoc Tran. A framework for controllable pareto front learning with completed scalarization functions and its applications. *arXiv preprint arXiv:2302.12487*, 2023.

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.

Hechuan Wen, Tong Chen, Li Kheng Chai, Shazia Sadiq, Junbin Gao, and Hongzhi Yin. Variational counterfactual prediction under runtime domain corruption. *arXiv preprint arXiv:2306.13271*, 2023.

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.

Qi Wu, David Bauer, Yuyang Chen, and Kwan-Liu Ma. Hyperinr: A fast and predictive hypernetwork for implicit neural representations via knowledge distillation. *arXiv preprint arXiv:2304.04188*, 2023.

Bing Xue, Ahmed Sameh Said, Ziqi Xu, Hanyang Liu, Neel Shah, Hanqing Yang, Philip Payne, and Chenyang Lu. Assisting clinical decisions for scarcely available treatment via disentangled latent representation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5360–5371, 2023.

Liuyi Yao, Zhixuan Chu, Sheng Li, Yaliang Li, Jing Gao, and Aidong Zhang. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5):1–46, 2021.

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.

Checklist

1. For all models and algorithms presented, check if you include:

   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes

   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Not Applicable

   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. No. To be released on acceptance.

2. For any theoretical claim, check if you include:

   (a) Statements of the full set of assumptions of all theoretical results. Not Applicable

   (b) Complete proofs of all theoretical results. Not Applicable

   (c) Clear explanations of any assumptions. Not Applicable

3. For all figures and tables that present empirical results, check if you include:

   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). Yes

   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. Yes

   (b) The license information of the assets, if applicable. Not Applicable

   (c) New assets either in the supplemental material or as a URL, if applicable. Not Applicable

   (d) Information about consent from data providers/curators. Not Applicable

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. Not Applicable

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable

A AN ADDITIONAL EXAMPLE of ITE vs HYPERITE

Fig. 4 provides another example of application of the proposed framework to SNet+ Chauhan et al. (2023a), and compares the architecture and gradient flows of SNet+ against HyperSNet+. SNet+ is presented in subfigure (a), where weights w_0 and w_1 of potential outcome (PO) function estimators $\hat{\mu}_0$ and $\hat{\mu}_1$ and w_p for propensity score estimator $\hat{\pi}$ are learnable, and are trained through data points corresponding to the treatment group, i.e., $\hat{\mu}_0$ learns only from control group $T = 0$ and $\hat{\mu}_1$ learns only from treatment group $T = 1$. SNet+ has partial information sharing through latent factors, however, the PO functions have different gradient flows so SNet+ lacks end-to-end information sharing. On the other hand, for HyperSNet+ in subfigure (b), weights w_0, w_1 and w_p are not learnable as they are generated by hypernet. The data corresponding to any treatment group leads to updating weights of hypernet parameters during each learning step and this soft-weight sharing between PO functions enables dynamic end-to-end information sharing between treatment groups.
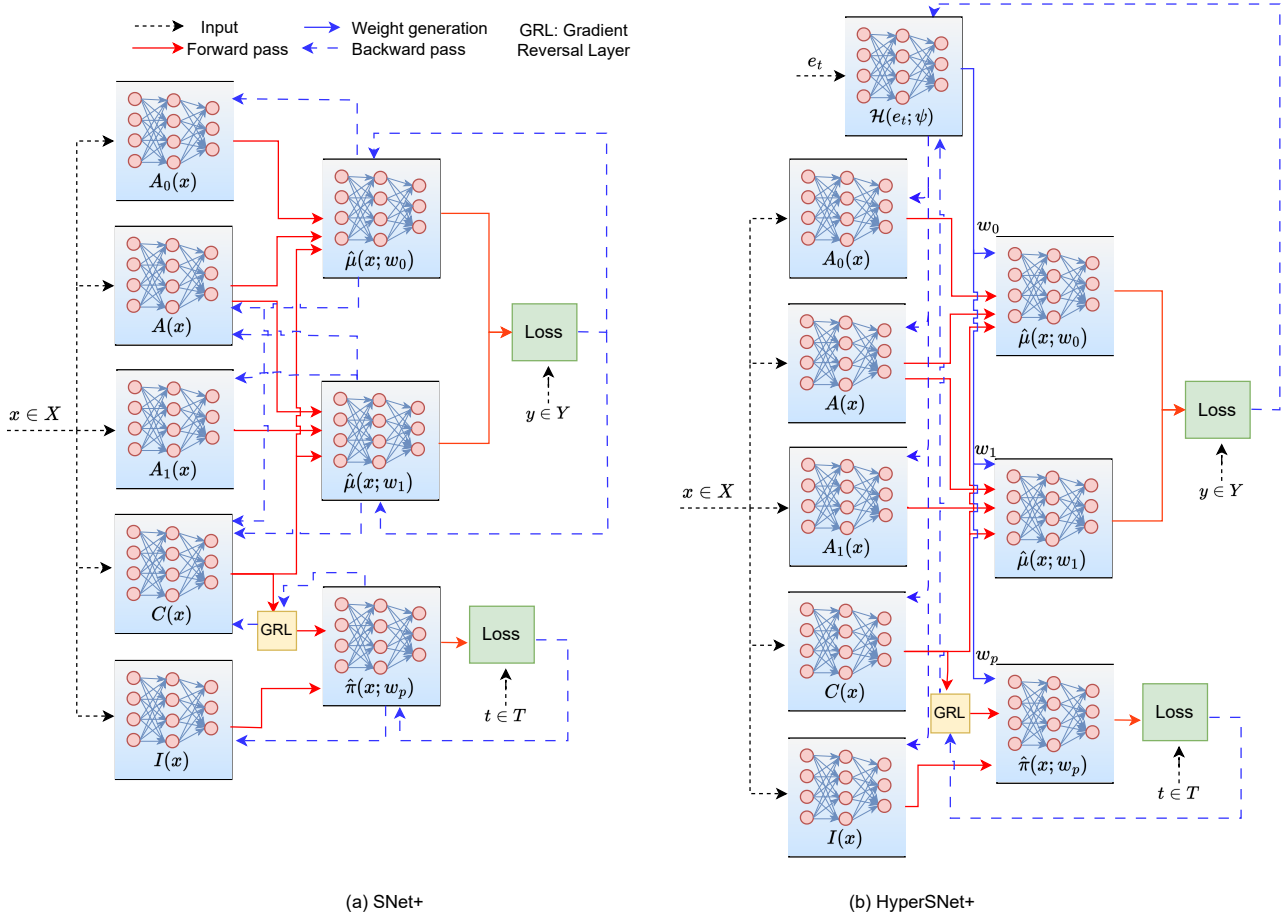

Figure 4: An overview of the architectures and gradient flows for SNet+ and HyperSNet+, where $\hat{\mu}(x; w_0), \hat{\mu}(x; w_1)$ and $\hat{\pi}(x; w_p)$ have exactly the same architecture in both but different training process.

B IMPLEMENTATION DETAILS

We follow Crabbé et al. (2022) 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 ITE learner perform well for all the problems Machlanski et al. (2023) 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 ITE and HyperITE, 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. MitNet has architecture similar to TARNet, and for discrete treatments, it uses adversarial training. For tVAE, we use official implementation[3] with two hidden layers of 100 neurons each for encoder as well as decoder. Similarly, for all direct learners, each model in the first and second 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 ITE learners use ReLU as an activation function in the hidden layers of ITE learners as well as hypernet. We have used Adam as an optimizer Kingma and Ba (2014) 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 HyperITE, 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 Miyato et al. (2018) in the hypernet network to stabilize the training. All experiments use generate once hypernet that generates all the weights of an ITE learner at once. The gradient reversal parameter in adversarially trained ITE learners, like 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 and 16 for HyperSLearner, HyperTLearner, HyperTARNet, HyperSNet+, HyperRALearner, and HyperDRLearner, 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 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 ten seeds from one to ten and reports one standard error, except for IHDP dataset where results are averaged over 100 runs due to smaller size of the dataset. Final code is available at *https://github.com/jmdvinodjmd/HyperITE*.

HyperITE versions of ITE Learners: All HyperITE learners have exact same architecture as that of the corresponding baselines, however they are trained through soft weight sharing of hypernet. For HyperTARNet and HyperMitNet, only weights of PO functions are generated and common representation is learned directly from the data, as shows in the figure in main part of the paper. We can generate the entire learner also but that will increase complexity of hypernet and common layers already share information. Similarly, HyperSNet+ generates weights for PO functions and propensity score estimator and latent factors are directly learned. For indirect learners, there is only one step so PO functions are directly generated by hypernet. HyperSLearner and HyperTLearner need to generate one and two networks, respectively. However, direct learners need multiple stages and have multiple models. For HyperDRLearner, in step one, one hypernet is used to generate weights for three networks corresponding to two PO functions and one propensity score estimator, however, second step is similar to DRLearner as there is direct learning of treatment effect from pseudo-outcomes. HyperRALearner needs to generate weights only for PO functions in step one, while step two is similar to RA-Learner as treatment effect is directly regressed on pseudo-outcomes. Number of embedding vectors needed in an HyperITE depends on number of networks whose weights have to be generated, as hypernet need to identify the network, e.g., HyperTLearner needs two embedding vectors corresponding to two PO functions.

C   ADDITIONAL EXPERIMENTS

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

C.1   Effect of Hypernets Type / Weight Generation Strategy

Hypernets are neural networks that generate weights for ITE learners. Depending on architecture of ITE learner, size of output layer of hypernet can become large and can affect convergence of learning process. To manage complexity of hypernets, which is one of the challenges in hypernets, instead of generating weights once/together, weights can be generated in chunk-wise or component-wise (component refers to a layer or a channel) Oswald

---

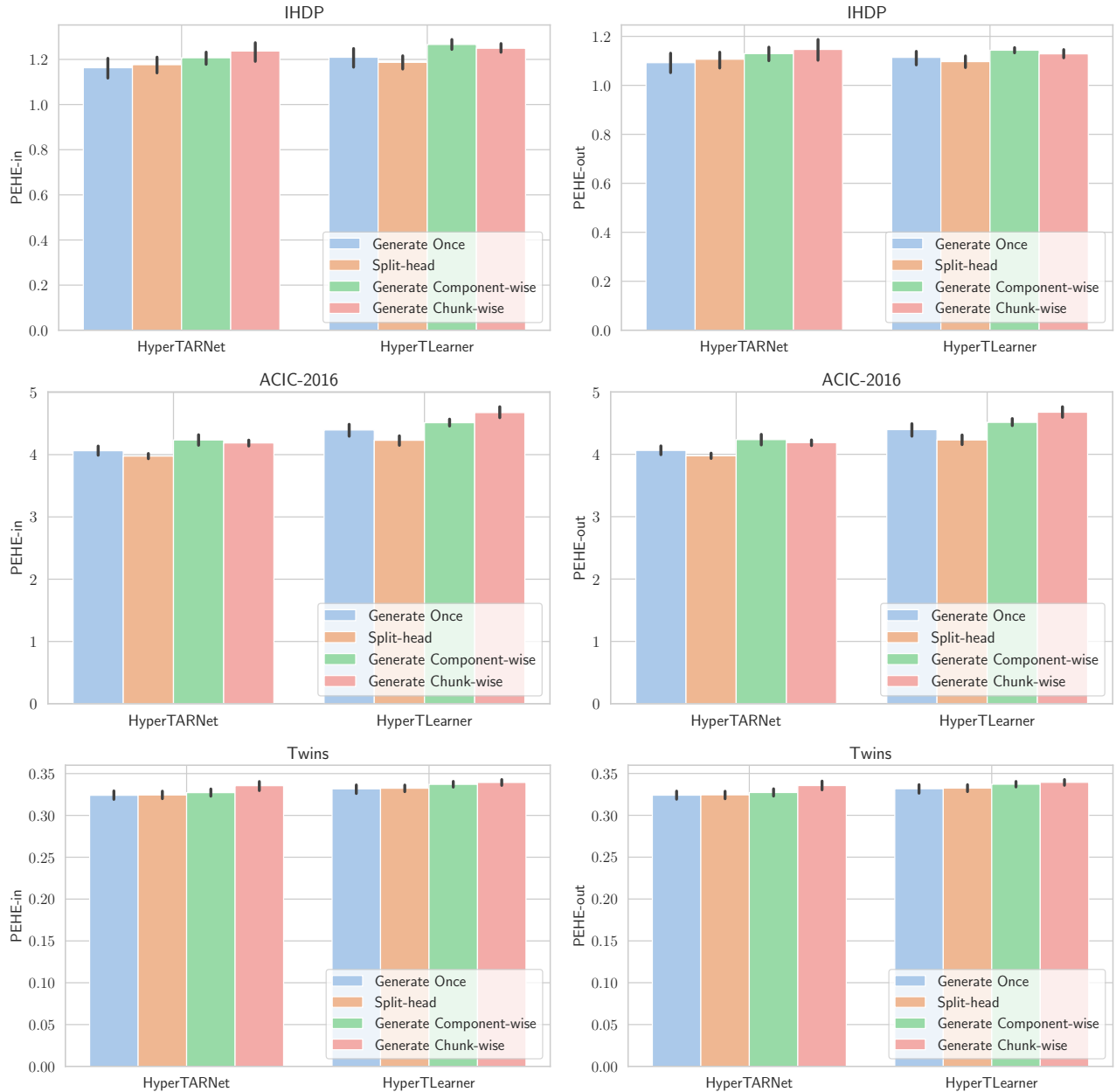[3]https://github.com/xuebing1234/tvae

Figure 5: Effect of hypernet's type on performance of HyperTARNet and HyperTLearner using PEHE-in (left) and PEHE-out (right) performance metrics on IHDP, ACIC-2016 and Twins datasets.

et al. (2020). However, each chunk or component will need an additional embedding so that hypernet could distinguish among them for weight generation. Additionally, split-head approach uses multiple heads for weights generation and can work in combination with other approaches. These target weight generation strategies have their own advantages and disadvantages, e.g., for generate once strategy, size of 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, chunk-wise generation 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 layer size, then the weights of a layer will not be generated together. Similarly, component-wise weight generation simplifies output space at 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 ITE learner, however, weights of each layer are generated altogether. The split-head weight generation has the best

of all approaches as it simplifies the output space without affecting the input space. However, one limitation of the split-head approach is that we can't have a large number of heads for weight generation, otherwise, that will make each head very small.

Fig. 5 studies the effect of type of hypernet on performance of HyperITE on IHDP, ACIC-2016 and Twins datasets with HyperTARNet and HyperTLearner. For chunk-wise generation, we have used ten chunks that reduces output layer size by ten 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. Component-wise generation (here referring to layer-wise generation) reduces output space only by 80% and does not have a large drop as in chunk-wise generation. This is because of having only two hidden layers in our ITE learners. Split-head approach is applied to generate once weight strategy with two splits so it helps to reduce number of parameters in output layer of the hypernet by two times. In terms of performance, from the figure, we observe that generate once and split-head weight generation strategies outperform the rest across all the datasets as well for both HyperITEs. However, there is no clear pattern between chunk-wise and layer-wise approach of weight generation in hypernets for the problem under study. Moreover, there is also not a clear winner between split-head and generate once strategy as split-head performs best on ACIC-2016 dataset while generate once performs best, with a tiny margin against split-head, on Twins dataset. While on IHDP dataset, split-head performs best for HyperTLearner and generate once performs best for HyperTARNet. It is difficult to generalize the choice of hypernet type as it depends on dataset as well as target network.

C.2 Effect of Embedding Size

Hypernet takes an embedding as an input and maps the embedding to weights for corresponding target network whose weights have to be generated. Fig. 6 presents the effect of embedding size on the performance of the hypernetworks. We have selected one ITE learner from each category, i.e., HyperTARNet from the representation learning and HyperTLearner from meta-learners, 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.

C.3 Comparative Study with Additional ITE Learners

Here, we discuss tVAE Xue et al. (2023) as an additional baseline, which is disentangled representation based variational autoencoder and learns latent representation where first three element corresponds to two PO functions and one to propensity score estimator. So, by architecture, tVAE is a concrete ITE learner that provides end-to-end information sharing. We have used official implementation of tVAE. We present the comparative study here as the results were unstable with ACIC dataset as it leads to NaN values. Results with rest of the datasets are presented in Table 2, and from the table, we find that it performs worse than FlexTENet Curth and van der Schaar (2021a) which is another concrete ITE learner with flexible architecture for information sharing. Overall, tVAE performs the worst on IHDP, however, it performs better on T-Learner and RA-Learner on Twins dataset.

C.4 Effect of Limited Observational Datasets

Here, we discuss the effect of dataset size on the performance of the HyperITE 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 similar performance of learners on PEHE-in and -out performance metrics, and as discussed earlier, results show that HyperITEs show 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. This is potentially due to having single learner and hence end-to-end information sharing. While, as observed earlier, HyperDRLearner does not show any clear pattern, and this is potentially due to regularization of second step by errors in the first step. 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
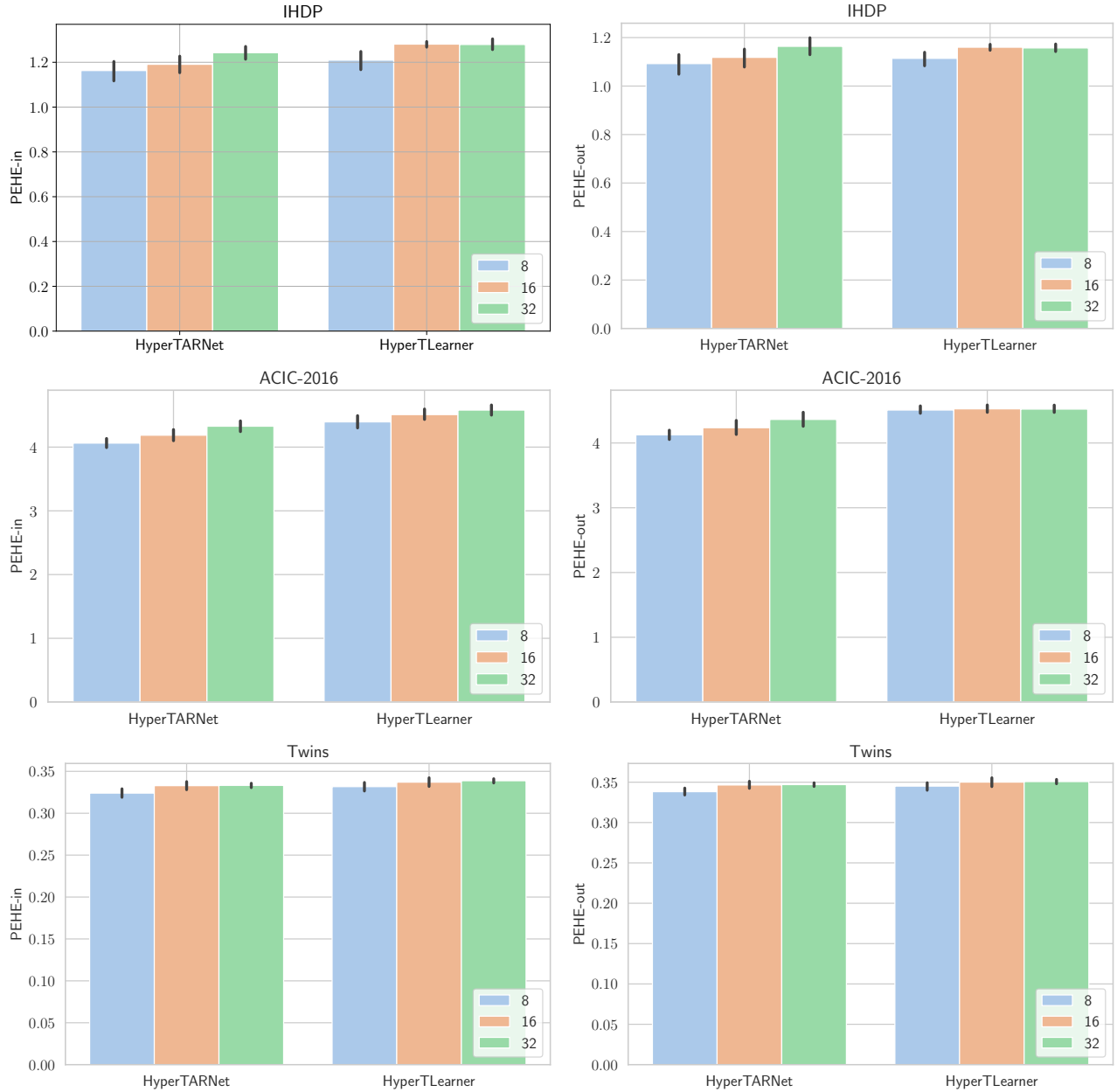
Figure 6: Effect of embedding size on the performance of HyperTARNet and HyperTLearner using PEHE-in and -out performance metrics on IHDP, ACIC-2016 and Twins datasets.

unclear performance with ACIC-2016.

C.5 Convergence of ITE and HyperITE Learners

Here, we discuss convergence of HyperITE against ITE learners on three benchmark datasets. However, it is to be noted that we can't compare their convergence for all learners because some learners have multiple training stages where each stage may have multiple models trained independently. But HyperITE can train all models of each stage together through soft weight sharing, i.e., through one model. For example, meta-learners like DR-Learner has two stages where first stage trains three independent models, and T-Learner trains two independent models, whereas HyperDRLearner and HyperTLearner train one model for first stage. Fig. 8 presents convergence for validation loss for MitNet and TARNet against HyperMitNet and HyperTARNet learners. From the figure, it is clear that HyperITEs converge faster than corresponding baselines across all the benchmarks because of dynamic

Table 2: Comparative study of ITE learners against HyperITE learners on IHDP and Twins benchmarks using PEHE-out (lower is better) as a performance metric. Each experiment is averaged over multiple runs and values in parenthesis refer to one standard error.

Learner	IHDP PEHE-out	Twins PEHE-out
FlexTENet	1.19 (0.011)	0.328 (0.000)
tVAE	1.38 (0.003)	0.366 (0.001)
SLearner	0.97 (0.001)	0.331 (0.001)
HyperSLearner	**0.95 (0.001)**	**0.324 (0.001)**
TLearner	1.32 (0.003)	0.410 (0.001)
HyperTLearner	**1.12 (0.001)**	**0.345 (0.003)**
DRLearner	1.12 (0.002)	**0.323 (0.001)**
HyperDRLearner	**1.09 (0.002)**	**0.323 (0.001)**
RALearner	1.25 (0.002)	0.384 (0.001)
HyperRALearner	**1.06 (0.002)**	**0.329 (0.001)**
TARNet	1.24 (0.002)	0.366 (0.001)
HyperTARNet	**1.08 (0.002)**	**0.338 (0.002)**
MitNet	1.25 (0.004)	0.324 (0.012)
HyperMitNet	**1.12 (0.005)**	**0.323 (0.012)**
SNet+	1.22 (0.001)	0.353 (0.003)
HyperSNet+	**1.17 (0.001)**	**0.325 (0.001)**

end-to-end information sharing between potential outcome functions. This helps HyperITE versions to perform better from the first epoch resulting in better estimates.
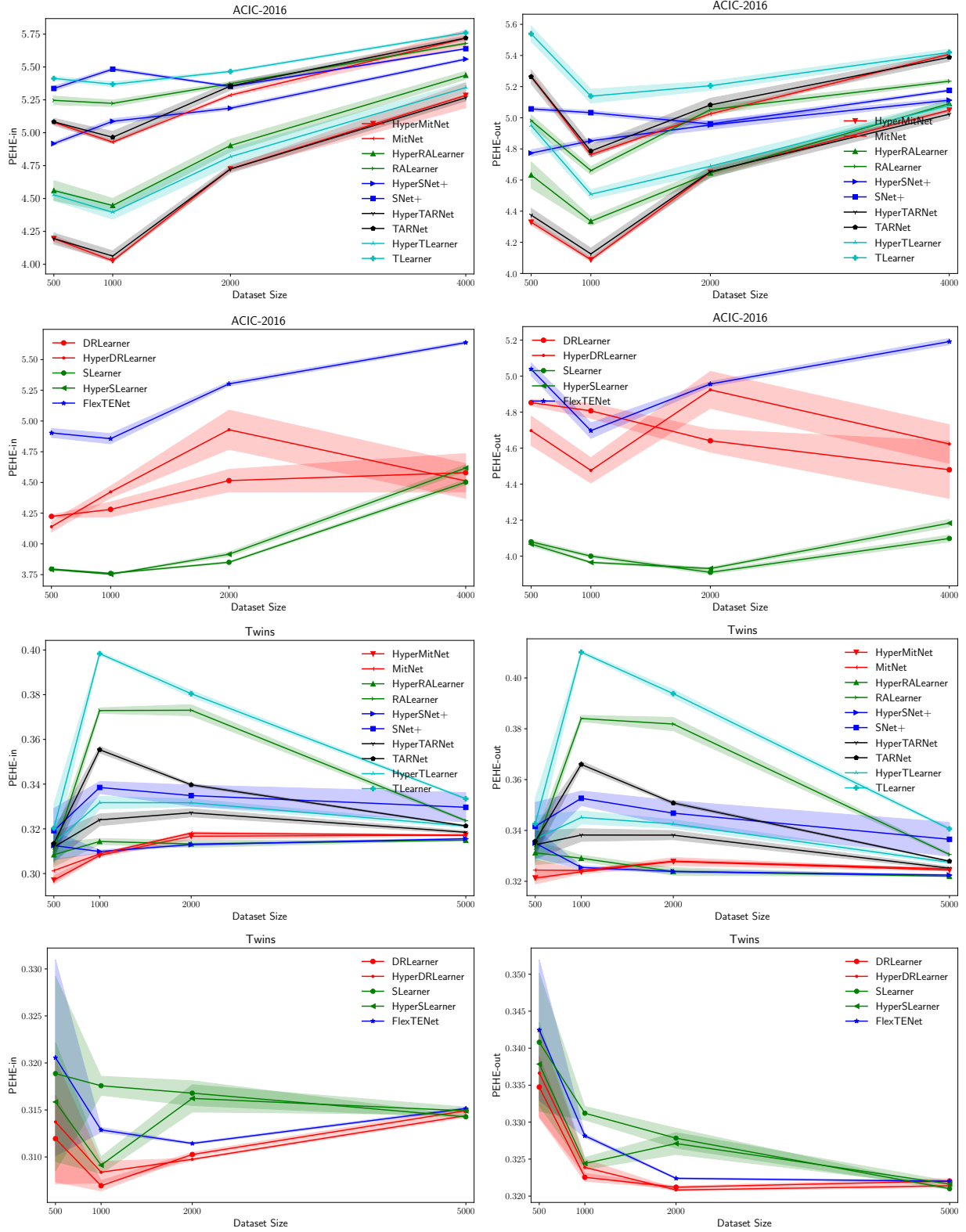
Figure 7: Effect of dataset size on the performance of learners using ACIC-2016 and Twins datasets where left panel shows results for PEHE-in and right panel shows results for PEHE-out (shaded region shows one standard error).
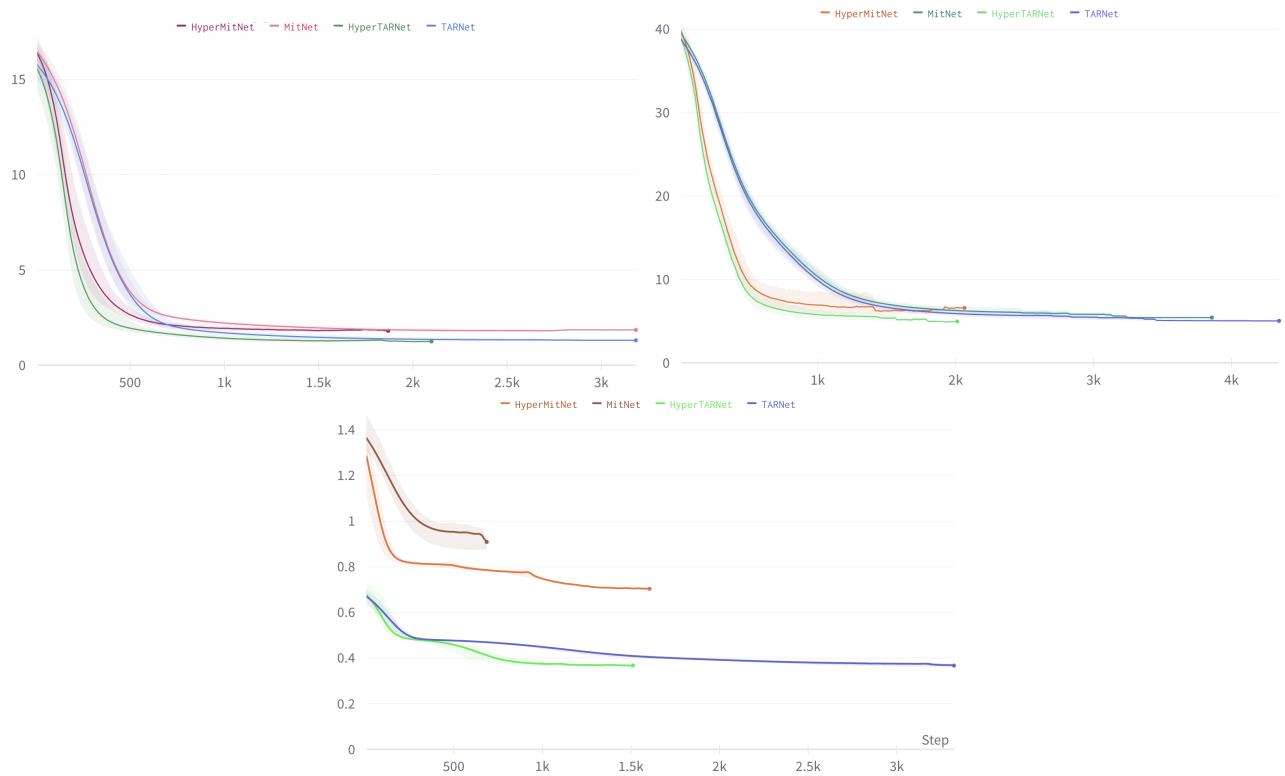
Figure 8: Convergence for validation loss against weight update steps for HyperITEs and corresponding baselines on IHDP (top left), ACIC-2016 (top right) and Twins (bottom) benchmarks (shaded region is difference of max and min values).