Estimating average causal effects from patient trajectories

Dennis Frauen
LMU Munich
frauen@lmu.de

Tobias Hatt
ETH Zurich
thatt@ethz.ch

Valentyn Melnychuk
LMU Munich
melnychuk@lmu.de

Stefan Feuerriegel
LMU Munich & University Hospital Bern
feuerriegel@lmu.de

ABSTRACT

In medical practice, treatments are selected based on the expected causal effects on patient outcomes. Here, the gold standard for estimating causal effects are randomized controlled trials; however, such trials are costly and sometimes even unethical. Instead, medical practice is increasingly interested in estimating causal effects among patient subgroups from electronic health records, that is, observational data. In this paper, we aim at estimating the average causal effect (ACE) from observational data (patient trajectories) that are collected over time. For this, we propose DeepACE: an end-to-end deep learning model. DeepACE leverages the iterative G-computation formula to adjust for the bias induced by time-varying confounders. Moreover, we develop a novel sequential targeting procedure which ensures that DeepACE has favorable theoretical properties, i. e., is doubly robust and asymptotically efficient. To the best of our knowledge, this is the first work that proposes an end-to-end deep learning model for estimating time-varying ACEs. We compare DeepACE in an extensive number of experiments, confirming that it achieves state-of-the-art performance. We further provide a case study for patients suffering from low back pain to demonstrate that DeepACE generates important and meaningful findings for clinical practice. Our work enables medical practitioners to develop effective treatment recommendations tailored to patient subgroups.

CCS CONCEPTS

• **Mathematics of computing** → **Probability and statistics**; • **Applied computing** → **Health informatics**; • **Computing methodologies** → **Neural networks**; **Artificial intelligence**.

KEYWORDS

causal inference, neural networks, treatment effect estimation, longitudinal data, average causal effect

ACM Reference Format:
Dennis Frauen, Tobias Hatt, Valentyn Melnychuk, and Stefan Feuerriegel. 2022. Estimating average causal effects from patient trajectories. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22).* ACM, Washington, DC, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Causal effects of treatments on patient outcomes are hugely important for decision-making in medical practice [41]. These estimates inform medical practitioners in the expected effectiveness of treatments and thus guide treatment selection. Notwithstanding, information on causal effects is also relevant for other decision-making of other domains such as public health [9] and marketing [38].

Causal effects can be estimated from either randomized controlled trials (RCTs) or observational studies [27]. Even though RCTs present the gold standard for estimating causal effects, they are often highly costly, unfeasible in practice, or even unethical (e. g., medical professionals cannot withhold effective treatment to patients in need) [27]. Therefore, medical practice is increasingly relying on observational data to study causal relationships between treatments and patient outcomes. Nowadays, electronic health records are readily available, capture patient trajectories with high granularity, and thus provide rich observational data in medical practice [2].

In this paper, we aim at estimating causal effects from observational data in form of patient trajectories. Patient trajectories encode medical histories in a time-resolved manner and are thus of longitudinal form. However, estimating causal effects from observational data is subject to challenges [26]. One reason, because the underlying treatment assignment mechanism is usually confounded with the patient outcomes. Another reason is that confounders may vary over time, which introduces additional dependencies and treatment-confounder feedback. As an example, consider a physician who assigns medications to patients over time. Here, the patient outcome is a measurement of some vital signs indicating the patient's health status. Typically, the physician selects treatments based on a patient's time-varying covariates such as blood pressure or heart rate, yet which also affect the patient outcome and are influenced by previous treatments. Hence, methods for estimating causal effects must adjust for time-varying confounders in order to produce unbiased estimates.

While there are works on estimating individualized causal effects [4, 15, 16], we are interested in **average causal effects (ACEs)**. As such, ACEs give the expected different in health outcomes when applying different treatment interventions at the level of patient subgroups. Estimating ACEs is especially important for medical practice for a number of reasons [26]. (i) There are many settings in practice where interventions affect whole populations such as in public health. For example, a government might be interested in the different effects of a stay-home order on COVID-19 spread for vaccinated vs. non-vaccinated people, or what the effect of sugar tax is on diabetes onsets among society. Here, treatments affect subgroups of people, thus necessitating *average* causal effects. (ii) In medical practice, personalization is typically done by varying treatment recommendations across patient subgroups. To

KDD '22, August 14–18, 2022, Washington, DC, USA
2022. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

this end, medical research seeks to find increasingly more granular subgroups [20]. These should identify differences in disease dynamics and thus capture different phenotypes [2], so that different treatment guidelines are tailored to each subgroup [10]. To do so, one must again compare causal effects across patient cohorts, that is, *average* causal effects across patient cohorts. Motivated by such considerations from medical practice, our work aims at time-varying ACE estimation.

Proposed method: We propose an end-to-end deep learning model to estimate time-varying ACEs, called DeepACE. DeepACE combines a recurrent neural network and feed-forward neural networks to learn conditional expectations of factual and counterfactual outcomes under complex non-linear dependencies, based on which we then estimate time-varying ACEs. In DeepACE, we address time-varying confounding by leveraging the G-formula, which expresses the ACE as a sequence of nested conditional expectations based on observational data as. Existing methods are limited in that these learn the nested conditional expectations *separately* by performing an iterative procedure [35]. In contrast, our end-to-end model DeepACE makes it possible to learn them *jointly*, leading to a more efficient use of information across time.

We further develop a *sequential targeting procedure* by leveraging results from semi-parametric estimation theory in order to improve the estimation quality of DeepACE. The sequential targeting procedure perturbs ("targets") the outputs of DeepACE so that our estimator satisfies a semi-parametric efficient estimating equation. To achieve this, we propose a targeting layer and a targeted regularization loss for training. We then derive that DeepACE provides a doubly robust and asymptotically efficient estimator.

Our main **contributions**:[1]

(1) We propose DeepACE: the first end-to-end neural network for estimating time-varying average causal effects using observational data. DeepACE builds upon the iterative G-computation formula to address time-varying confounding.
(2) We develop a novel sequential targeting procedure which ensures that DeepACE provides a doubly robust and asymptotically efficient estimator.
(3) We perform an extensive series of computational experiments using state-of-the-art models for time-varying ACE estimation, establishing that DeepACE achieves a superior performance. We further demonstrate that DeepACE generates important findings based on a medical case study for patients suffering from low back pain.

2 RELATED WORK

Estimating causal effects from observational data can be loosely grouped into static and longitudinal settings (Table 1).

2.1 Causal effect estimation in the static setting

Extensive works focus on treatment effect estimation in static settings. Two important methods that adopt machine learning for *average* treatment effect estimation in the static setting are: (i) targeted maximum likelihood estimation (TMLE) [36] and (ii) DragonNet [31]. TMLE is a plugin estimator that takes a (machine learning)

[1]Code available at https://anonymous.4open.science/r/DeepACE-6797 (Upon acceptance, we replace the link and point to a public GitHub repository).

Table 1: Key methods for causal effect estimation. This paper: ACE for longitudinal settings.

		Static setting	Longitudinal setting
Causal effects	Individual (=ITE)	e. g., TARNet [30], causal forest [39]	RMSNs [16], CRN [4], G-Net [15]
	Average (=ACE)	e. g., TMLE [36], DragonNet [31]	g-methods [19], LTMLE [33, 35], **DeepACE** (ours)

model as input and perturbs the predicted outcomes, so that the final estimator satisfies a certain efficient estimating equation. This idea builds upon semi-parametric efficiency theory and is often called *targeting*. Any estimator satisfying the efficient estimating equation is guaranteed to have desirable asymptotic properties. On the other hand, DragonNet is a neural network that incorporates a targeting procedure into the training process by extending the network architecture and adding a tailored regularization term to the loss function. This allows the model parameters to adapt simultaneously to provide a targeted estimate. To the best of our knowledge, there exists no similar targeting procedure for longitudinal data, and, to fill this gap, we later develop a tailored *sequential* targeting procedure.

In general, causal effect estimation for static settings is different from longitudinal settings due to time-varying confounding and treatment-confounder feedback [26]. The latter can appear when dealing with longitudinal observational data. Hence, methods for static causal effect estimation are biased when they are applied to longitudinal settings, thus leading to an inferior performance. For this reason, we later use methods for time-varying causal effect estimation as our prime baselines. Results for static baselines are reported in Appendix B.

2.2 Causal effect estimation in longitudinal settings

Causal effect estimation in longitudinal settings is often also called *time-varying* causal effect estimation. Here, we distinguish individual and average causal effects.

Individual causal effects: There is a growing body of work on adapting neural networks for estimating *individual* causal effects in the longitudinal setting (also called individual treatment effects or ITE for short). These methods predict counterfactual outcomes conditioned on patient trajectories, and then output causal effects by taking differences between the factual and counterfactual outcome. Recurrent marginal structural networks (RMSNs) [16] use inverse probability weighting to learn a sequence-to-sequence model that addresses bias induced by time-varying confounders. The counterfactual recurrent network (CRN) [4] adopts adversarial learning to build balanced representations. The G-Net [15] incorporates the G-formula into a recurrent neural network and applies Monte Carlo sampling.

In this paper, we are instead not interested in individual but average causal effects. Still, we later include the above state-of-the-art methods from ITE estimation (i. e., RMSNs, CRN, and G-Net) as baselines (we average the individual estimates). There are other methods for predicting counterfactual outcomes over time (e. g., [22, 29, 32]), which are not applicable due to different settings or assumptions (e. g., stronger unconfoundedness assumptions).

Average causal effects: Several methods for time-varying ACE estimation originate from epidemiological literature. Here, common are so-called g-methods [19]. Examples of g-methods include marginal structural models [27], G-computation via the G-formula [26], and structural nested models [24]. The previous methods make linearity assumptions and are consequently not able to exploit non-linear dependencies within the data. Nevertheless, we include the g-methods as baselines.

We are aware of only one work that leverages machine learning methods for time-varying ACE estimation: longitudinal targeted maximum likelihood estimation (LTMLE) [33]. LTMLE has two components: (i) LTMLE makes use iterative G-computation. Mathematically, the G-formula can be expressed in terms of nested conditional expectations, which can then be learned successively by using arbitrary regression models. As such, each conditional expectation is learned *separately*, which is known as iterative G-computation. Our method, DeepACE, follows a similar approach but estimates the conditional expectations *jointly*. (ii) LTMLE targets the estimator by perturbing the predicted outcomes in each iteration to make them satisfy an efficient estimating equation. In contrast to LTMLE, our sequential targeting procedure is incorporated into the model training process, which allows all model parameters to be learned simultaneously. Later, we implement two variants of LTMLE as baselines, namely LTMLE with a generalized linear model (glm) [33] and LTMLE with a super learner [35].

Research gap: To the best of our knowledge, there exists no *end-to-end* machine learning model for time-varying ACE estimation. Hence, DeepACE is the first neural network that simultaneously learns to perform iterative G-computation and to apply a sequential targeting procedure. By learning all parameters jointly, we expect our end-to-end model to provide more accurate estimation results.

3 PROBLEM SETUP

3.1 Setting

We build upon the standard setting for estimating time-varying ACEs [26, 33]. For each time step $t \in \{1, \dots, T\}$, we observe (time-varying) patient covariates $X_t \in \mathbb{R}^p$, treatments $A_t \in \{0, 1\}$, and outcomes $Y_{t+1} \in \mathbb{R}$. For example, we would model critical care for COVID-19 patients by taking blood pressure and heart rate as time-varying patient covariates, ventilation as treatment, and respiratory frequency as outcome. Modeling the treatments A_t as binary variables is consistent with prior works [31, 36] and is standard in medical practice [27]: should one apply a treatment or not?

At each time step t, the treatment A_t directly affects the next outcome Y_{t+1}, and the covariates X_t may affect both A_t and Y_{t+1}. All X_t, A_t, and Y_{t+1} may have direct effects on future treatments, covariates, and outcomes. The corresponding causal graph is shown in Fig. 1. For notation, we denote the observed trajectory at time t by $\mathcal{H}_t = (\bar{X}_t, \bar{A}_{t-1})$, where $\bar{X}_t = (X_1, \dots, X_t)$ and $\bar{A}_{t-1} = (A_1, \dots, A_{t-1})$. We always consider the lagged outcomes Y_t to be included in the covariates X_t.

We have further access to an observational dataset $\mathcal{D}$, that consists of N independent patient trajectories $\mathcal{D}$ for patients $i \in \{1, \dots, N\}$, i. e., $\mathcal{D} = (\{x_t^{(i)}, a_t^{(i)}, y_{t+1}^{(i)}\}_{t=1}^{T})_{i=1}^{N}$. Such patient trajectories are nowadays widely available in electronic health records [2]. For notation, we use a superscript (i) to refer to patients (we omit it unless needed).

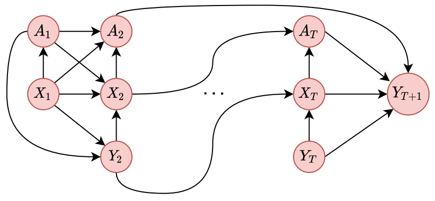

Figure 1: One possible example of a causal graph describing the data-generating process.

We build upon the potential outcomes framework [28], which was extended in [27] to the longitudinal setting. We denote $Y_{t+1}(\bar{a}_t)$ as the potential outcome, which would have been observed at time $t+1$ if a treatment intervention $\bar{a}_t = (a_1, \dots, a_t)$ was applied. Note that $Y_{t+1}(\bar{a}_t)$ is unobserved if the treatment intervention $\bar{a}_t$ does not coincide with the treatment assignments $\bar{A}_t$ in the observational dataset. This is also known as the fundamental problem of causal inference [21].

3.2 ACE estimation

Given two intended treatment assignments $\bar{a}_T = (a_1, \dots, a_T)$ and $\bar{b}_T = (b_1, \dots, b_T)$, we define the expected potential outcomes as

$$\theta^a = \mathrm{E}\left[Y_{T+1}(\bar{a}_T)\right] \text{ and } \theta^b = \mathrm{E}\left[Y_{T+1}(\bar{b}_T)\right]. \tag{1}$$

Objective: We aim at estimating the average causal effect (ACE):

$$\psi = \theta^a - \theta^b. \tag{2}$$

To do so, we build upon the following standard assumptions in causal inference [27]:

Assumption 1 (Consistency). *The treatment assignment $\bar{A}_t = \bar{a}_t$ implies $Y_{t+1}(\bar{a}_t) = Y_{t+1}$, i. e., potential and observed outcomes coincide for all $t \in \{1, \dots, T\}$.*

Assumption 2 (Positivity). *If $P(\bar{A}_{t-1} = \bar{a}_{t-1}, \bar{X}_t = \bar{x}_t) > 0$, then $P(A_t = a_t \mid \bar{A}_{t-1} = \bar{a}_{t-1}, \bar{X}_t = \bar{x}_t) > 0$ for all $a_t \in \{0, 1\}$.*

Assumption 3 (Sequential ignorability). *For all $\bar{a}_t \in \{0, 1\}^t$ and $t \in \{1, \dots, T\}$, it holds that $Y_{t+1}(\bar{a}_t)) \perp\!\!\!\perp A_t \mid \mathcal{H}_t$.*

Assumption 3 implies that there are no unobserved confounders that influence both treatment assignments and outcomes. Together, Assumptions 1–3 allow to identify the ACE ψ from observational data [21].

3.3 Iterative G-computation

The expected potential outcome θ^a can be expressed in terms of the observational data via the well-known G-formula [23]. For our setting, we consider a variant that uses iterated conditional expectations [3, 25] to write θ^a as

$$\theta^a = \mathrm{E}\Big[\dots \mathrm{E}\Big[\mathrm{E}\left[Y_{T+1} \,\middle|\, \bar{X}_T, \bar{A}_T = \bar{a}_T\right] \,\Big|\, \bar{X}_{T-1}, \bar{A}_{T-1} = \bar{a}_{T-1}\Big] \dots\Big]. \tag{3}$$

For a better understanding of Eq. (3), we can equivalently write the parameter θ^a as the result of an iterative process. More precisely, we introduce recursively defined conditional expectations Q_t^a that depend on the covariates $\bar{X}_{t-1}$ and interventions $\bar{a}_{t-1}$ via

$$Q_t^a = Q_t\left(\bar{X}_{t-1}, \bar{a}_{t-1}\right) = \mathrm{E}\left[Q_{t+1}^a \mid \bar{X}_{t-1}, \bar{A}_{t-1} = \bar{a}_{t-1}\right] \quad (4)$$

for $t \in \{2, \ldots, T+1\}$, and initialize $Q_{T+2}^a = Y_{T+1}$. Then, the expected potential outcome can be written as

$$\theta^a = \mathrm{E}\left[Q_2^a\right]. \quad (5)$$

In Eq. (4), the covariates $X_T, X_{T-1}, \ldots$ are successively integrated out, and the θ^a is obtained in Eq. (5) by averaging.

In the following, we review iterative G-computation [35], which is an iterative procedure that leverages Eq. (4) and Eq. (5) to estimate the expected potential outcome θ^a and, subsequently, the average causal effect ψ. Iterative G-computation estimates the conditional expectations Q_t^a by using a regression model for all $t \in \{2, \ldots, T+1\}$. Then, θ^a can be estimated by taking the empirical mean in Eq. (5). The full algorithm is in Alg. 1.

Algorithm 1: Iterative G-computation [25, 35]

```
$\hat{Q}_{T+2}^a \leftarrow Y_{T+1}$
for $t \in \{T+1, T, \ldots, 2\}$ do
    $\hat{Q}_t(\cdot) \leftarrow$ Regress $\hat{Q}_{t+1}^a$ on $\left(\bar{X}_{t-1}, \bar{A}_{t-1}\right)$
    $\hat{Q}_t^a \leftarrow \hat{Q}_t\left(\bar{X}_{t-1}, \bar{a}_{t-1}\right)$
end
$\hat{\theta}^a \leftarrow \frac{1}{N}\sum_{i=1}^N \hat{Q}_2^{a^{(i)}}$
```

Iterative G-computation only requires estimating conditional expectations. This is usually an easier task as compared to estimating conditional distributions, which is done by other G-computation methods [26]. However, iterative G-computation in the above form is subject to drawbacks. In particular, one has to specify T separate regression models that are trained separately. Each model uses the predictions of its already trained predecessor as training labels. This may lead to a training procedure which is comparatively unstable. The reason is that each training objective does not take into account how prediction errors propagate through time. For example, a model $\hat{Q}_{T+1}(\cdot)$ could generate predictions that approximate Y_{T+1} well (leading to a small regression objective) but that cause other models $\hat{Q}_{t+1}(\cdot)$ for $t < T$ to become unstable. However, when training $\hat{Q}_{t+1}(\cdot)$, the model $\hat{Q}_{T+1}(\cdot)$ is already fixed and cannot be trained anymore.

Need for an end-to-end model: We postulate that an end-to-end model overcome the above drawbacks. In particular, an end-to-end model can share information across time steps and, thereby, should be able to generate more accurate estimates for time-varying ACEs. Motivated by this, our end-to-end model learns all parameters jointly.

4 DeepACE: END-TO-END DEEP LEARNING FOR ACE ESTIMATION

Overview: We propose a novel end-to-end deep learning model for ACE estimation, called DeepACE. DeepACE is motivated by the idea of iterative G-computation. It is trained with observational data from patient trajectories and a specific treatment intervention $\bar{a}_T$, based on which it learns the conditional expectations Q_t^a from Eq. (4).

DeepACE consists of two main components: (i) a **G-computation layer** (Sec. 4.1), which produces initial estimates of the Q_t^a by minimizing a tailored G-computation loss, and (ii) a **targeting layer** (Sec. 4.2), which applies perturbations in a way that the final estimator satisfies an efficient estimating equation. The overall model architecture is shown in Fig. 2.

DeepACE is trained by combining (i) a *G-computation loss* $\mathcal{L}_Q$, (ii) a *propensity loss* $\mathcal{L}_g$, and (iii) a *targeting loss* $\mathcal{L}_{\text{tar}}$ into a joint loss $\mathcal{L}$ as described later. The outputs of DeepACE can then be used to provide ACE estimates (Sec. 4.3). We further show that DeepACE provides a doubly robust and asymptotically efficient estimator (Sec. 4.4). Finally, we provide implementation details are described in Sec. 4.5.

4.1 G-computation layer

The G-computation layer takes the observational data $\mathcal{D}$ and a specific treatment intervention $\bar{a}_T$ as input. For each time step $t \in \{2, \ldots, T+1\}$, it generates two outputs: (i) a factual output $\hat{Q}_t^A$ for $Q_t\left(\bar{X}_{t-1}\bar{A}_{t-1}\right)$, and (ii) a counterfactual output $\hat{Q}_t^a$ for $Q_t\left(\bar{X}_{t-1}, \bar{a}_{t-1}\right)$ according to Eq. (4). The factual outputs $\hat{Q}_t^A$ are trained to estimate the one-step shifted counterfactual outputs $\hat{Q}_{t+1}^a$, while the counterfactual outputs $\hat{Q}_t^a$ are obtained by implicitly evaluating $\hat{Q}_t\left(\bar{X}_{t-1}, \cdot\right)$ at $\bar{A}_{t-1} = \bar{a}_{t-1}$ as done in Alg. 1.

4.1.1 Architecture. The architecture of the **G-computation layer** is shown in Fig. 2 (bottom). In the G-computation layer, we use a long short-term-memory (LSTM) layer [11] to process the input data. We choose an LSTM due to its ability to learn complex non-linear dynamics from patient trajectories while addressing the vanishing gradient problem which frequently occurs when using recurrent neural networks.

We feed the data twice into the LSTM: (i) with the observed treatments $\bar{A}_T$ (factual forward pass), and (ii) once with the treatment intervention $\bar{a}_T$ (counterfactual forward pass). Based on this, we computed the hidden LSTM states as follows. At each time step t, the factual forward pass leads to a factual hidden LSTM state h_t^A depending on the factual trajectory $\mathcal{H}_t = (\bar{X}_t, \bar{A}_{t-1})$, and the counterfactual forward pass leads to a counterfactual hidden LSTM state h_t^a depending on the past covariates $\bar{X}_t$ and interventions $\bar{a}_{t-1}$.

Both hidden states h_t^A and h_t^a are processed further. In the factual forward pass, we feed the factual hidden state h_t^A together with the current observed treatment A_t into a fully-connected feed-forward network FF_t^Q. The network FF_t^Q generates a factual output $\hat{Q}_{t+1}^A$ for $Q_{t+1}(\bar{X}_t, \bar{A}_t)$ according to Eq. (4). In the counterfactual forward pass, we feed the counterfactual hidden state h_t^a also FF_t^Q and replace the treatment input A_t with the current intervention a_t. As a result, the network FF_t^Q generates a counterfactual output $\hat{Q}_{t+1}^a$ for $Q_t^a = Q_{t+1}(\bar{X}_t, \bar{a}_t)$.

4.1.2 G-computation loss. We design a tailored loss function, such that we mimic Algorithm 1. For this, we denote the outputs of the G-computation layer for a patient i at time t by $\hat{Q}_{t+1}^{A^{(i)}}(\eta)$ and $\hat{Q}_{t+1}^{a^{(i)}}(\eta)$. Here, we explicitly state the dependence on the model parameters (i. e., the LSTM and feed forward layers), which we denote by η. We

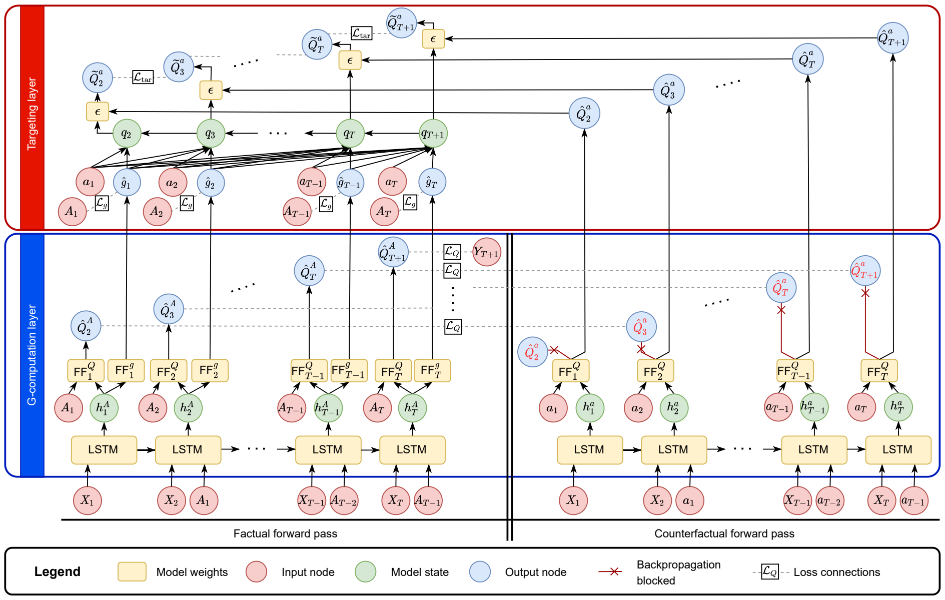

Figure 2: DeepACE consisting of the G-computation layer and the targeting layer.

define the *G-computation loss* as

$$\mathcal{L}_Q(\eta) = \frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\left(\left(\hat{Q}^{A^{(i)}}_{T+1}(\eta) - y^{(i)}_{T+1}\right)^2 + \sum_{t=2}^{T}\left(\hat{Q}^{A^{(i)}}_{t}(\eta) - \hat{Q}^{a^{(i)}}_{t+1}(\eta)\right)^2\right). \tag{6}$$

By way of how $\mathcal{L}_Q$ is constructed, each counterfactual output $\hat{Q}^a_{t+1}$ is used as a prediction objective by the previous factual output $\hat{Q}^A_t$. Recall that, in Algorithm 1, the counterfactual estimates $\hat{Q}^a_{t+1}$ are obtained only by evaluating the learned conditional expectation at $\bar{A}_t = \bar{a}_t$. Therefore, we only want the factual outputs $\hat{Q}^A_t$ to learn the counterfactual outputs $\hat{Q}^a_{t+1}$ and not the other way around. Hence, when training the model with gradient descent-based optimization, we block the gradient backpropagation through the counterfactual FF^Q_t during the counterfactual forward pass.

4.2 Targeting layer

For our sequential targeting procedure, we now introduce a **targeting layer**. The motivation is as follows: In principle, we could estimate the expected potential outcome θ^a by first training the G-computation layer, and subsequently following Eq. (5) and taking the empirical mean over the first counterfactual outputs $\hat{Q}^a_2$. Instead, we propose to leverage results from semi-parametric estimation theory, as this allows us to construct an estimator with better theoretical properties, namely double robustness and asymptotic efficiency.[2] For this purpose, we design our targeting layer so that it estimates the treatment assignments, i. e., the so-called propensity scores

$$g_t(\mathcal{H}_t) = P(A_t = 1 \mid \mathcal{H}_t) = \mathrm{E}(A_t \mid \mathcal{H}_t). \tag{7}$$

[2]For an overview on semi-parametric estimation theory, we refer to [13].

We then use the propensity scores to perturb the counterfactual outputs $\hat{Q}^a_t$ to make them satisfy an efficient estimating equation. To formalize this, we first provide the mathematical background and subsequently describe how we implement the targeting layer.

4.2.1 Mathematical background. In the following, we summarize the general framework under which semi-parametric efficient estimators can be obtained. Let $\hat{Q}^a = (\hat{Q}^a_2, \dots, \hat{Q}^a_{T+1})$ be estimators of the conditional expectations $(Q^a_2, \dots, Q^a_{T+1})$ from Eq. 4, and let $\hat{g} = (\hat{g}_1, \dots, \hat{g}_T)$ be estimators of the propensity scores $(g_1, \dots, g_T)$, where $g_t = g_t(\mathcal{H}_t)$. Furthermore, let $\hat{\theta}^a$ be an estimator of θ^a.

Ideally, we would like to obtain a tuple of estimators $(\hat{Q}^a, \hat{g}, \hat{\theta}^a)$ with the following properties:

(1) *Double robustness*: If either $\hat{Q}^a$ or $\hat{g}$ are consistent, $\hat{\theta}^a$ is a consistent estimator of θ^a.
(2) *(Semi-parametric) asymptotic efficiency*: If both $\hat{Q}^a$ and $\hat{g}$ are consistent, $\hat{\theta}^a$ achieves the smallest variance among all asymptotically linear estimators of θ^a.

It can be shown that, asymptotically, the tuple $(\hat{Q}^a, \hat{g}, \hat{\theta}^a)$ fulfills properties (1) and (2) if it satisfies the following *efficient estimating equation* [13]

$$\frac{1}{N}\sum_{i=1}^{N}\phi\left(\hat{Q}^{a^{(i)}}, \hat{g}^{(i)}, \hat{\theta}^a\right) = 0, \tag{8}$$

where ϕ is the efficient influence function of $(\hat{Q}^a, \hat{g}, \hat{\theta}^a)$. We call an estimator that satisfies Eq. (8) "targeted". For the longitudinal setting, ϕ has a closed form (derived in [33]) that is given by

$$\phi\left(Q^a, g, \theta^a\right) = \left(Q^a_2 - \theta^a\right) + \sum_{t=2}^{T+1}\left(Q^a_{t+1} - Q^a_t\right)\left(\prod_{\ell=1}^{t-1}\frac{\mathbb{1}(A_\ell = a_\ell)}{g_\ell(\mathcal{H}_\ell)}\right),$$

where we used the convention that $Q^a_{T+2} = Y_{T+1}$ and where $\mathbb{1}(\cdot)$ denotes the indicator function.

4.2.2 Implementation. We use our targeting layer to perturb the initial estimates produced by the G-computation layer in order to satisfy Eq. (8). Specifically, we propose a sequential targeting procedure by adding a model parameter that is jointly trained with the other model parameters. A tailored regularization term ensures that the efficient estimating estimation from Eq. (8) is satisfied.

Inputs to the targeting layer are (i) the counterfactual outputs $\hat{Q}^a_{t+1}(\eta)$ of the G-computation layer and (ii) predictions $\hat{g}_t(\eta)$ of the propensity scores $g_t(\mathcal{H}_t)$, where, η denotes the trainable parameters of the G-computation layer. To allow for gradient backpropagation, we obtain the counterfactual outputs $\hat{Q}^a_{t+1}(\eta)$ from an second (identical) output of FF^Q_t, where the gradient flow is not blocked during training. Furthermore, we generate the propensity estimates $\hat{g}_t(\eta)$ by adding separate feed-forward networks FF^g_t on top of the factual hidden states h^A_t.

In the following, we describe how the targeting layer applies perturbations to generate targeted outputs $\widetilde{Q}_{t+1}$. We recursively define perturbation values $q_{T+2}(\eta) = 0$ and

$$q_t(\eta) = q_{t+1}(\eta) - \prod_{\ell=1}^{t-1} \frac{\mathbb{1}(A_\ell = a_\ell)}{\hat{g}_\ell(\eta)} \tag{9}$$

for $t \in \{2, \ldots, T+1\}$. The perturbation values are used to create the targeted network outputs via

$$\widetilde{Q}^a_t(\eta, \epsilon) = \hat{Q}^a_t(\eta) + \epsilon q_t(\eta) \tag{10}$$

for $t \in \{2, \ldots, T+2\}$, where ϵ is an additional network parameter that is trained together with η. Note that, by definition, we have that $\widetilde{Q}^a_{T+2}(\eta, \epsilon) = Y_{T+1}$.

4.2.3 Loss. We use two regularization terms in order to train the targeting layer. First, we define the *propensity loss*

$$\mathcal{L}_g(\eta) = \frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\sum_{t=1}^{T} \mathrm{BCE}\left(\hat{g}^{(i)}_t(\eta), a^{(i)}_t\right), \tag{11}$$

where BCE denotes binary cross-entropy loss. The propensity loss ensures that the propensity networks learn to predict the propensity scores $g_t(\mathcal{H}_t)$. Second, we define our *targeting loss*

$$\mathcal{L}_{\mathrm{tar}}(\eta, \epsilon) = \frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\sum_{t=2}^{T+1} \left(\widetilde{Q}^{a^{(i)}}_{t+1}(\eta, \epsilon) - \widetilde{Q}^{a^{(i)}}_t(\eta, \epsilon)\right)^2. \tag{12}$$

We show in Sec.4.4 that our targeting loss forces the outputs $\widetilde{Q}_{t+1}$ to satisfy the efficient estimating from Eq. (8) and thus makes them "targeted".

In contrast to other sequential targeting methods (e. g., LTMLE [33]) that apply targeting perturbations iteratively over time, our procedure allows the entire model to be learn jointly. We later show that this gives more accurate ACE estimates.

4.3 ACE estimation

To estimate the ACE ψ for two treatment interventions $\bar{a}_T$ and $\bar{b}_T$, we train two separate DeepACE models for each $\bar{a}_T$ and $\bar{b}_T$. Then, we estimate ψ via

$$\hat{\psi} = \frac{1}{N}\sum_{i=1}^{N}\left(\widetilde{Q}^{a^{(i)}}_2 - \widetilde{Q}^{b^{(i)}}_2\right), \tag{13}$$

where $\widetilde{Q}^{a^{(i)}}_2$ and $\widetilde{Q}^{b^{(i)}}_2$ denote the two targeted DeepACE outputs for patient i at the time step $t = 2$.

4.4 Theoretical results

The following theorem ensures that our combination of targeting layer and regularization in DeepACE indeed produces a targeted estimator. Here, $\mathcal{L}(\eta, \epsilon)$ denotes the joint loss (defined in Sec. 4.5).

THEOREM 1. *Let $(\hat{\eta}, \hat{\epsilon})$ be a stationary point of $\mathcal{L}(\eta, \epsilon)$. Then, for any $\beta > 0$, the estimator*

$$\widetilde{\theta}^a = \frac{1}{N}\sum_{i=1}^{N}\widetilde{Q}^{a^{(i)}}_2(\hat{\eta}, \hat{\epsilon}) \tag{14}$$

is targeted, i. e., DeepACE satisfies the efficient estimating equation from Eq. (8).

PROOF. See Appendix A. □

By Theorem 1, the DeepACE estimator $\widetilde{\theta}^a$ is doubly robust, i. e., $\widetilde{\theta}^a$ is consistent, even if either the targeted outputs $\widetilde{Q}^a_{t+1}$ or the propensity estimates $\hat{g}_t$ are misspecified.

Assuming that the true conditional expectations Q^a_{t+1} and propensity scores g_t are contained in a suitable hypothesis class, the initial outputs $\hat{Q}^a_{t+1}$ from the G-computation layer and the propensity estimates $\hat{g}$ will converge to Q^a_{t+1} and g_t with growing sample size due to the construction of $\mathcal{L}_Q$ and $\mathcal{L}_g$ (provided that $\alpha > 0$). The next corollary shows that this implies asymptotic efficiency of $\widetilde{\theta}^a$.

COROLLARY 1. *If the initial outputs $\hat{Q}^a_{t+1}$ from the G-computation layer and propensity estimates $\hat{g}_t$ of DeepACE are consistent for Q^a_{t+1} and g_t, then also the targeted outputs $\widetilde{Q}^a_{t+1}$ are consistent for Q^a_{t+1}. In particular, the DeepACE estimator $\widetilde{\theta}^a$ is asymptotically efficient for θ^a.*

PROOF. See Appendix A. □

4.5 Model training

Overall loss: To train DeepACE, we combine the above into an overall loss

$$\mathcal{L}(\eta, \epsilon) = \mathcal{L}_Q(\eta) + \alpha\mathcal{L}_g(\eta) + \beta\mathcal{L}_{\mathrm{tar}}(\eta, \epsilon), \tag{15}$$

where α and β are constants that control the amount of propensity and targeting regularization, respectively.

Implementation: We implemented DeepACE using the the PyTorch Lightning framework. We incorporated variational dropout [8] in the LSTM. This allows us to provide uncertainty estimates using DeepACE. We used the Adam optimizer [14] with 100 epochs. The feed-forward are set to one layer each, and the layer sizes are subject to hyperparameter tuning. Details on our hyperparameter tuning and training are in Appendix D.

5 EXPERIMENTS

5.1 Baselines

We compare DeepACE against state-of-the-art methods for time-varying causal effect estimation, see Table 1. The baselines are selected from recent literature on causal effect estimation [15, 33].

(1) **G-methods:** Here, we use: (i) a marginal structural network (**MSN**) [27] with inverse probability weighting, (ii) **iterative G-computation** as in Algorithm 1, (iii) **G-computation** via the parametric G-formula [26], and (iv) a structural nested mean model (**SNMM**) with g-estimation [24].

(2) **Longitudinal targeted maximum likelihood estimation (LTMLE)** [33]: We implement LTMLE in two variants. The first variant (i) uses generalized linear models **(glm)** to estimate the conditional expectations. The second variant (ii) uses the **super learner** algorithm [34], which builds upon a cross-validation algorithm to combine different regression models into a single predictor.

(3) **Deep learning for ITE estimation:** Additionally, we include (i) recurrent marginal structural networks (**RMSNs**) [16], (ii) a counterfactual recurrent network (**CRN**) [4], and (iii) **G-Net** [15]. Different from our method, these baselines predict individual as opposed to average causal effects. We obtain ACE estimates by averaging the predicted ITEs.

Implementation and hyperparameter tuning details are in Appendix C. For completeness, we also reported results for static baselines in Appendix B but note that these aim at a different setting and have thus an inferior performance.

5.2 Experiments using synthetic data

Setting: Synthetic data are commonly used to evaluate the effectiveness of causal inference methods (e. g., [5, 31, 36]). For real-world data, the counterfactual outcomes are never observed, and, hence, the ground-truth ACE is unknown. In contrast, synthetic data allow us to have access to counterfactual outcomes. Therefore, we can successfully compute the ground-truth and thus benchmark the performance.

We generate patient trajectories $\mathcal{D} = \left(\{x_t^{(i)}, a_t^{(i)}, y_{t+1}^{(i)}\}_{t=1}^{T}\right)_{i=1}^{N}$ from a data-generating process similar to [5]. The time-varying covariates $X_t \in \mathbb{R}^p$ follow a non-linear autoregressive process

$$X_t = \tanh\left(\sum_{i=1}^{h} \left(\alpha_i X_{t-i} + \beta_i \gamma_i (2A_{t-i} - 1)\right) + \epsilon_t^X\right) \tag{16}$$

for some lag $h \geq 1$, randomly sampled weights $\alpha_i, \beta_i \sim \mathcal{N}(1/(i+1), 0.02^2)$, $\gamma_i \sim \text{unif}\{-1, 1\}$ (discrete uniform distribution), and noise $\epsilon_t^X \sim \mathcal{N}_p(0, \text{diag}_p(0.1^2))$. The treatments $A_t \in \{0, 1\}$ are selected as $A_t = \mathbf{1}_{\{\pi_t > 0.5\}}$, where π_t are treatment probabilities. They depend on the past observations via

$$\pi_t = \sigma\left(\tan\left(\prod_{i=1}^{h} \bar{X}_{t-i}\right) + \frac{1}{p} Y_t + \epsilon_t^A\right), \tag{17}$$

where σ denote the sigmoid function and $\epsilon_t^A \sim \mathcal{N}(0, 0.2^2)$ is noise. Finally, the outcomes $Y_{t+1} \in \mathbb{R}$ are determined via

$$Y_{t+1} = \bar{X}_t + \sum_{i=1}^{h} \left(w_i \, (2A_{t-i+1} - 1)\right) + \epsilon_{t+1}^Y \tag{18}$$

for weights $w_i = (-1)^{i+1} \frac{1}{i}$ and noise $\epsilon_{t+1}^Y \sim \mathcal{N}(0, 0.1^2)$.

Results: We generate $N = 1000$ patient trajectories over $T = 15$ time steps and sample $p = 6$ time-varying covariates with lag $h = 5$. At the same time, we use the same process and noise to generate counterfactual data for three setups with different treatment interventions $\bar{a}_T$ and $\bar{b}_T$.[3] For each method and setup, we calculate the absolute error between estimated and ground-truth averaged over 5 different runs with random seeds.

[3]Here $\bar{b}_T$ is fixed to the zero intervention (no treatment applied), and $\bar{a}_T$ is chosen as $(\mathbb{1}(k \leq i \leq \ell))_{i \in \{1,\ldots,T\}}$ for $k \in \{1, 3, 5\}$ and $\ell \in \{10, 13, 15\}$.

Table 2: Results on synthetic data (mean ± standard deviation).

Method	Setup 1	Setup 2	Setup 3
(1) G-METHODS			
MSM [27]	0.24 ± 0.18	0.27 ± 0.19	0.21 ± 0.10
Iterative G-computation [35]	0.26 ± 0.23	0.34 ± 0.26	0.32 ± 0.27
G-formula (parametric) [26]	0.14 ± 0.10	0.44 ± 0.25	0.86 ± 0.45
SNMM [24]	0.39 ± 0.04	0.47 ± 0.02	0.35 ± 0.02
(2) LTMLE			
LTMLE (glm) [33]	0.20 ± 0.19	0.25 ± 0.19	0.33 ± 0.24
LTMLE (super learner) [33, 34]	0.13 ± 0.08	0.14 ± 0.12	0.19 ± 0.13
(3) DEEP LEARNING FOR ITE ESTIMATION			
RMSNs [16]	0.20 ± 0.10	0.52 ± 0.32	0.95 ± 0.52
CRN [4]	0.20 ± 0.10	0.52 ± 0.32	0.95 ± 0.52
G-Net [15]	0.18 ± 0.08	0.46 ± 0.26	0.97 ± 0.38
DeepACE w/o targeting (ours)	**0.04 ± 0.03**	0.10 ± 0.09	**0.15 ± 0.11**
DeepACE (ours)	**0.04 ± 0.02**	**0.09 ± 0.07**	0.18 ± 0.08

lower = better (best in bold)

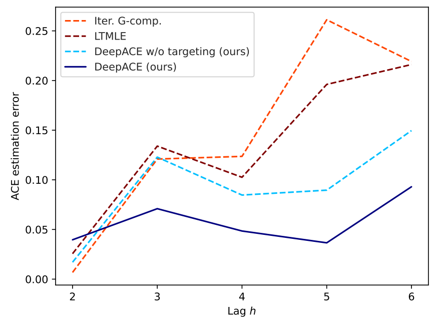

Figure 3: Performance comparison. Shown: mean estimation error averaged over 5 runs.

The results are shown in Table 2. All baselines are clearly outperformed by DeepACE on all three experiments. The best-performing baseline is LTMLE with the super learner. This is reasonable as it is the only baseline available that is both tailored for ACE estimation and makes use of machine learning. The linear methods (i. e., g-methods, LTMLE, and glm) are not able to capture the non-linear dependencies within the data and thus achieve an inferior performance. The ITE baselines use recurrent neural networks and should thus be able to learn non-linearities but, nevertheless, are inferior. This is attributed to the fact that they are designed for estimating individual rather than average causal effects.

To compare DeepACE with iterative G-computation, we compute the estimation errors of both methods over different time lags $h \in \{1, \ldots, 5\}$. The results are shown in Fig. 3, showing the effectiveness of DeepACE for long-range dependencies as common in modern EHRs.

Ablation study: We also analyze DeepACE but where the targeting layer is removed (Table 2). Both variants of DeepACE outperform the baselines. The variant without targeting performs well but we are careful with interpretations due to the simple nature of the synthetic data and rather relegate conclusions to real-world data as in the following.

Table 3: Results on semi-synthetic data (mean ± standard deviation).

Method	Setup 1	Setup 2	Setup 3
(1) G-methods			
MSM [27]	2.35 ± 0.64	3.06 ± 0.67	2.47 ± 0.95
Iterative G-computation [35]	0.81 ± 0.35	0.72 ± 0.72	1.90 ± 1.06
G-formula (parametric) [26]	0.32 ± 0.27	0.31 ± 0.20	0.32 ± 0.27
SNMM [24]	0.28 ± 0.33	0.52 ± 0.26	1.96 ± 2.36
(2) LTMLE			
LTMLE (glm) [33]	0.82 ± 0.35	0.72 ± 0.72	1.84 ± 1.13
LTMLE (super learner) [33, 34]	0.96 ± 1.01	0.92 ± 1.17	0.76 ± 0.47
(3) Deep learning for ITE estimation			
RMSNs [16]	2.35 ± 0.14	2.32 ± 0.18	2.36 ± 0.14
CRN [4]	2.53 ± 0.03	2.53 ± 0.04	2.52 ± 0.04
G-Net [15]	0.67 ± 0.15	0.65 ± 0.17	0.67 ± 0.15
DeepACE w/o targeting (ours)	**0.18 ± 0.17**	0.25 ± 0.11	0.21 ± 0.07
DeepACE (ours)	**0.18 ± 0.14**	**0.12 ± 0.14**	**0.16 ± 0.10**

lower = better (best in bold)

5.3 Experiments using semi-synthetic data

Setting: We create semi-synthetic data that enables us to evaluate DeepACE using real-world data while having access to the ground-truth ACE. For this purpose, we use the MIMIC-III dataset [12], which includes electronic health records from patients admitted to intensive care units.

We use a preprocessing pipeline [40] to extract 10 time-varying covariates $X_t \in \mathbb{R}^{10}$ over $T = 15$ time steps. Then, treatments $A_t \in \{0, 1\}$ are simulated as binary variables with treatment probabilities

$$\pi_t = \sigma\left(\sum_{i=1}^{h}\left(\frac{(-1)^i}{1-i}\left(\bar{X}_{t-i} + \tanh(Y_{t-i})\right)\right) - \tanh\left(\ell_{t-1} - \frac{T}{2}\right) + \epsilon_t^A\right), \quad (19)$$

where $\epsilon_t^A \sim \mathcal{N}(0, 0.5^2)$ is noise and ℓ_t is the current treatment level, defined by $\ell_t = \ell_{t-1} + 2(A_t - 1)\bar{X}_t \tanh(Y_t)$ and initialization $\ell_0 = T/2$. Finally, outcomes are generated via

$$Y_{t+1} = 5\sum_{i=1}^{h}\left(\frac{(-1)^i}{1-i}\tanh\left(\sin(\bar{X}_{t-i}A_{t-i} + \cos(\bar{X}_{t-i}A_{t-i})\right)\right) + \epsilon_t^Y, \quad (20)$$

where $\epsilon_t^Y \sim \mathcal{N}(0, 0.1^2)$ is noise.

Results: We generate $N = 1000$ patient trajectories with lag $h = 8$. We compare three setups with different treatment interventions that are chosen analogous to Sec. 5.2. The results are shown in Table 3. Again, DeepACE outperforms all baselines by a large margin.

Ablation study: We repeat the experiments with DeepACE but where the targeting layer is removed (Table 3). This thus demonstrates the importance of our targeting procedure for achieving a superior performance.

5.4 Case study using real-world data

Setting: We demonstrate the value of DeepACE for a real-world patient trajectories collected in a clinical study. For this purpose, we analyze data from $N = 928$ patients with low back pain (LBP) [20]. The dataset consists of pain as a time-varying outcomes recorded via assessments at $T = 3$ time steps over the course of a year. Pain intensity is our outcome of interest, Y_{t+1}. As treatment, we consider whether a patient has been doing physical labor or was exempt such as by recommendation of a medical professional. Covariates are given by perceived disability and 127 risk factors (e. g., age, gender). Hence, we are interested in the causal effect of whether patients have been allowed/disallowed to do physical labor on pain intensity. As of now, little is understood about potential between-patient heterogeneity in LBP progression. Due to that, medical research is interested in identifying different phenotypes whereby LBP is classified into different subgroups with clinical meaningful interpretation. We thus estimate average causal effects for two patient cohorts, namely (1) severe LBP and (2) mild LBP [20]. We repeated this for $K = 20$ evaluations with variational droupout for uncertainty estimation.

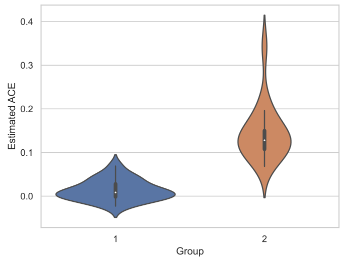

Figure 4: Violin charts (boxplots) showing the distributions of DeepACE estimates (standardized) for two patient subgroups.

Results: We make several observations relevant for medical practice (Fig. 4): (i) Both ACEs are positive, which is line with medical knowledge implying that physical labor may worsen LBP progression. (ii) Surprisingly, the ACE is larger for the group with mild LBP. While this may not sound not intuitive at first, it matches common medical practice: patients with mild LBP are rarely recommended to stop physical labor. Hence, our results suggest that physical labor should be also stoped for mild LBP, as this would eliminate negative effects for observed pain. (iii) The variance is much larger for mild LBP, indicating that impact of physical labor and thus quality of care is more heterogeneous.

6 DISCUSSION

Need for DeepACE: Estimating causal effects from observational data requires custom methods that adjust for time-varying confounding. For the above experiments, we also implemented a standard LSTM but found that it was inferior to the other baselines (results omitted for space). This is to be expected [26]: a standard LSTM does not address treatment-confounder feedback because of which it has a large generalization error and is even biased [1]. As a remedy, we add to a growing stream of research that develops tailored methods for estimating causal effects.

Strengths: DeepACE improves on state-of-the-art methods for time-varying ACE estimation in several ways. (i) We incorporate iterative G-computation into an end-to-end deep learning model. Our experiments confirm that this achieves superior estimation results. (ii) DeepACE provides a targeted estimator that is both doubly robust and asymptotically efficient. To achieves this, DeepACE does not rely on an iterative targeting procedure as, e. g., LTMLE, but instead incorporates a novel sequential targeting during the end-to-end training procedure. This is inspired by DragonNet [31] but the original targeting is for static settings, whereas we are the first to handle longitudinal settings. (iii) DeepACE is an end-to-end model, which is an advantage when dealing with patient subgroups. The entire data from all subgroups can be used for learning. In contrast, LTMLE or iterative G-computation can only use data from a single subgroup separately, thereby impeding performance.

Practical considerations: As it is with other research in causal inference from observational data, one challenge for causal effect estimation is that one only observes factual but not counterfactual outcomes. For practice in medicine and industry, this has implications as the performance cannot be evaluated directly but would necessitate RCTs. However, this also pinpoints the need for DeepACE: RCTs are costly and oftentimes even unethical (e. g., one cannot withhold effective treatments from patients to estimate causal effects) [27]. Here, a powerful remedy is offered by methods for estimating causal effects from observational data and, in particular, our DeepACE.

Use cases: DeepACE fulfills important needs for decision-making in medicine and industry. In medicine, current guidelines typically make recommendations at the level of patient subgroups [10]: treatments are selected that promise the desired effects on health groups for patients from a specific subgroup, and, as such, are based on *average* causal effects. By offering more accurate ACE estimation, DeepACE will enable better decision-making for patient subgroups and will thus contribute to more effective care. Here, we expect EHRs and post-approval drug monitoring to offer valuable data for input. Moreover, DeepACE may also inform RCT design where one tests new treatment strategies that were generated from observed patient trajectories. Beyond that, ACE estimation is also relevant, for example, in marketing (e. g., what does the treatment effect of price discounts on purchase behavior differ for online vs. offline customers condition on their their past purchase history?).

Conclusion: In this paper, we proposed DeepACE for time-varying ACE estimation. To the best of our knowledge, DeepACE is the first end-to-end deep learning model for that purpose. By estimating causal effects of treatments on health outcomes, DeepACE informs the choice of effective treatments for patient subgroups.

REFERENCES

[1] Ahmed M. Alaa and Mihaela van der Schaar. 2018. Limits of estimating heterogeneous treatment effects: Guidelines for practical algorithm design. *ICML* (2018).

[2] Ahmed Allam, Stefan Feuerriegel, Michael Rebhan, and Michael Krauthammer. 2021. Analyzing patient trajectories with artificial intelligence. *Journal of medical Internet research* 23, 12 (2021), e29812.

[3] Heejung Bang and James M. Robins. 2005. Doubly robust estimation in missing data and causal inference models. *Biometrics* 61, 4 (2005), 962–973.

[4] Ioana Bica, Ahmed M. Alaa, James Jordon, and Mihaela van der Schaar. 2020. Estimating counterfactual treatment outcomes over time through adversarially balanced representations. *ICLR* (2020).

[5] Iona Bica, Ahmed M. Alaa, and Mihaela van der Schaar. 2020. Time series deconfounder: Estimating treatment effects over time in the presence of hidden confounders. *ICML* (2020).

[6] Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James M. Robins. 2018. Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal* 21, 1 (2018), C1–C68.

[7] Dylan J. Foster and Vasilis Syrgkanis. 2019. Orthogonal statistical learning. *arXiv preprint arXiv:1901.09036v3* (2019).

[8] Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. *NeurIPS* (2016).

[9] Thomas A. Glass, Steven N. Goodman, Miguel A. Hernán, and Jonathan M. Samet. 2013. Causal inference in public health. *Annual review of public health* 34 (2013), 61–75.

[10] Margaret A. Hamburg and Francis S. Collins. 2010. The path to personalized medicine. *The New England Journal of Medicine* 363, 4 (2010), 301–304.

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[12] Alistair E. W. Johnson et al. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data* 3, 1 (2016), 160035.

[13] Edward H. Kennedy. 2016. Semiparametric theory and empirical processes in causal inference. *Statistical causal inferences and their applications in public health research* (2016).

[14] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *ICLR* (2015).

[15] Rui Li, Stephanie Hu, Mingyu Lu, Yuria Utsumi, Prithwish Chakraborty, Daby M. Sow, Piyush Madan, Jun Li, Mohamed Ghalwash, Zach Shahn, and Lehman Li-wei. 2021. G-Net: A recurrent network approach to G-computation for counterfactual prediction under a dynamic treatment regime. *ML4H* (2021).

[16] Bryan Lim, Ahmed M. Alaa, and Mihaela van der Schaar. 2018. Forecasting treatment responses over time using recurrent marginal structural networks. *NeurIPS* (2018).

[17] Ruoqi Liu, Changchang Yin, and Ping Zhang. 2020. Estimating individual treatment effects with time-varying confounders. *ICDM* (2020).

[18] Sean McGrath, Victoria Lin, Zilu Zhang, Lucia C. Petito, Roger W. Logan, Miguel A. Hernán, and Jessica G. Young. 2020. gfoRmula: An R package for estimating the effects of sustained treatment strategies via the parametric G-formula. *Patterns* 1, 3 (2020).

[19] Ashley I. Naimi, Stephen R. Cole, and Edward H. Kennedy. 2017. An introduction to G-methods. *International Journal of Epidemiology* 46, 2 (2017), 756–762.

[20] Anne Molgaard Nielsen, Peter Kent, Lise Hestbaek, Werner Vach, and Alice Kongsted. 2017. Identifying subgroups of patients using latent class analysis: Should we use a single-stage or a two-stage approach? A methodological study using a cohort of patients with low back pain. *BMC Musculoskeletal Disorders* 18, 1 (2017), 57.

[21] Judea Pearl. 2009. *Causality.* Cambridge University Press, New York City.

[22] Zhaozhi Qian, Yao Zhang, Ioana Bica, Angela M. Wood, and Mihaela van der Schaar. 2021. SyncTwin: Treatment effect estimation with longitudinal outcomes. *NeurIPS* (2021).

[23] James M. Robins. 1986. A new approach to causal inference in mortality studies with a sustained exposure period: Application to control of the healthy worker survivor effect. *Mathematical Modelling* 7 (1986), 1393–1512.

[24] James M. Robins. 1994. Correcting for non-compliance in randomized trials using structural nested mean models. *Communications in Statistics - Theory and Methods* 23, 8 (1994), 2379–2412.

[25] James M. Robins. 1999. Robust estimation in sequentially ignorable missing data and causal inference models. *Proceedings of the American Statistical Association on Bayesian Statistical Science* (1999), 6–10.

[26] James M. Robins and Miguel A. Hernán. 2009. *Estimation of the causal effects of time-varying exposures.* CRC Press, Boca Raton.

[27] James M. Robins, Miguel A. Hernán, and Babette Brumback. 2000. Marginal structural models and causal inference in epidemiology. *Epidemiology* 11, 5 (2000), 550–560.

[28] Donald B. Rubin. 1978. Bayesian inference for causal effects: The role of randomization. *The Annals of Statistics* 6, 1 (1978), 34–58.

[29] Peter Schulam and Suchi Saria. 2017. Reliable decision support using counterfactual models. *NeurIPS* 30 (2017).

[30] Uri Shalit, Fredrik D. Johansson, and David Sontag. 2017. Estimating individual treatment effect: Generalization bounds and algorithms. *ICML* (2017).

[31] Claudia Shi, David M. Blei, and Victor Veitch. 2019. Adapting neural networks for the estimation of treatment effects. *NeurIPS* (2019).

[32] Hossein Soleimani, Adarsh Subbaswamy, and Suchi Saria. 2017. Treatment-response models for counterfactual reasoning with continuous-time, continuous-valued interventions. *UAI* (2017).

[33] Mark J. van der Laan and Susan Gruber. 2012. Targeted minimum loss based estimation of causal effects of multiple time point interventions. *The International Journal of Biostatistics* 8, 1 (2012).

[34] Mark J. van der Laan, Eric C. Polley, and Alan E. Hubbard. 2007. Super learner. *Statistical Applications in Genetics and Molecular Biology* 6 (2007), 1–23.

[35] Mark J. van der Laan and Sherri Rose. 2018. *Targeted Learning in data science.* Springer, Cham.

[36] Mark J. van der Laan and Donald B. Rubin. 2006. Targeted maximum likelihood learning. *The International Journal of Biostatistics* 2, 1 (2006).

[37] Stijn Vansteelandt and Arvid Sjolander. 2016. Revisiting G-estimation of the effect of a time-varying exposure subject to time-varying confounding. *Epidemiologic Methods* 5, 1 (2016).

[38] Hal R. Varian. 2016. Causal inference in economics and marketing. *PNAS* 113, 27 (2016), 7310–7315.

[39] Stefan Wager and Susan Athey. 2018. Estimation and inference of heterogeneous treatment effects using random forests. *JASA* 113, 523 (2018), 1228–1242.

[40] Shirly Wang, Matthew B.A. McDermott, Geeticka Chauhan, Marzyeh Ghassemi, Michael C. Hughes, and Tristan Naumann. 2020. MIMIC-Extract: A data extraction, preprocessing, and representation pipeline for MIMIC-III. *CHIL* (2020).

[41] Azam M. Yazdani and Eric Boerwinkle. 2015. Causal inference in the age of decision medicine. *Journal of Data Mining in Genomics & Proteomics* 6, 1 (2015).

A PROOFS

Proof of Theorem 1. We show that the tuple $\left(\widetilde{Q}^a(\hat{\eta}, \hat{\epsilon}), \bar{g}(\hat{\eta}), \widetilde{\theta}^a\right)$ satisfies the efficient estimating equation in Eq. (8). At any stationary point $(\hat{\eta}, \hat{\epsilon})$, we have that

$$
\begin{aligned}
0 &= \frac{\partial}{\partial \epsilon} \mathcal{L}(\eta, \epsilon)\Big|_{(\eta,\epsilon)=(\hat{\eta},\hat{\epsilon})} && (21)\\
&= \beta \frac{\partial}{\partial \epsilon} \mathcal{L}_{\text{tar}}(\eta, \epsilon)\Big|_{(\eta,\epsilon)=(\hat{\eta},\hat{\epsilon})} && (22)\\
&= \frac{\beta}{NT} \sum_{i=1}^{N} \sum_{t=2}^{T+1} \left(\widetilde{Q}_{t+1}^{a^{(i)}}(\hat{\eta}, \hat{\epsilon}) - \widetilde{Q}_{t}^{a^{(i)}}(\hat{\eta}, \hat{\epsilon})\right)\left(q_{t+1}(\hat{\eta}) - q_t(\hat{\eta})\right) && (23)\\
&= \frac{\beta}{NT} \sum_{i=1}^{N} \sum_{t=2}^{T+1} \left(\widetilde{Q}_{t+1}^{a^{(i)}}(\hat{\eta}, \hat{\epsilon}) - \widetilde{Q}_{t}^{a^{(i)}}(\hat{\eta}, \hat{\epsilon})\right) \prod_{\ell=1}^{t-1} \frac{\mathbb{1}(a_\ell^{(i)} = a_\ell)}{\hat{g}_\ell(\hat{\eta})} && (24)\\
&= \frac{\beta}{NT} \sum_{i=1}^{N} \phi\left(\bar{Q}_{T+1}^a(\hat{\eta}, \hat{\epsilon}), \bar{g}(\hat{\eta}), \widetilde{\theta}^a\right). && (25)
\end{aligned}
$$

Multiplying both sides with $\frac{T}{\beta}$ yields the result. □

Proof of Corollary 1. We show that consistency of $\hat{Q}_{t+1}^a$ and $\hat{g}_t$ implies consistency of the targeted outputs $\widetilde{Q}_{t+1}^a$. Then, asymptotic efficiency follows from Theorem 1.

The key argument is similar as in [31]. By assumption, the G-computation loss $\mathcal{L}_Q$ and the propensity loss $\mathcal{L}_g$ are asymptotically minimized by $\hat{Q}_{t+1}^a = Q_{t+1}^a$ and $\hat{g}_t = g_t$ (due to finite Vapnik–Chervonenkis (VC) dimension). In the following, we show that the overall loss $\mathcal{L}$ is asymptotically minimized by additionally setting $\hat{\epsilon} = 0$. Hence, the targeted outputs $\widetilde{Q}_{t+1}^a$ are consistent because DeepACE can set the perturbation parameter to $\hat{\epsilon} = 0$, which implies $\widetilde{Q}_{t+1}^a = \hat{Q}_{t+1}^a$ for all $t \in \{1, \dots, T\}$.

Because the targeting layer only adds a single parameter to the model, a finite VC dimension is preserved. We show now that $\hat{\epsilon} = 0$ asymptotically minimizes each summand of the targeting loss $\mathcal{L}_{\text{tar}}$. The last summand for $t = T + 1$ is minimized (squared loss) at

$$\hat{Q}_{T+1}^a + \hat{\epsilon} q_{T+1} = \mathrm{E}[Y_{T+1} \mid \bar{X}_T, \bar{A}_T = \bar{a}_T] = Q_{T+1}^a, \quad (26)$$

which is indeed achieved at $\hat{\epsilon} = 0$. The other summands for $t \in \{2, \dots, T\}$ are minimized at

$$
\begin{aligned}
\hat{Q}_t^a + \hat{\epsilon} q_t &= \mathrm{E}[\widetilde{Q}_{t+1}^a \mid \bar{X}_{t-1}, \bar{A}_{t-1} = \bar{a}_{t-1}] && (27)\\
&= Q_t^a + \hat{\epsilon}\mathrm{E}[q_{t+1}^a \mid \bar{X}_{t-1}, \bar{A}_{t-1} = \bar{a}_{t-1}], && (28)
\end{aligned}
$$

which is also achieved at $\hat{\epsilon} = 0$. □

B RESULTS FOR STATIC BASELINES

There are several methods for causal effect estimation in the static setting. However, these do not take into account time-varying confounding and are biased in the longitudinal setting. Because of that, we refrained from reporting them in our main paper. To show that DeepACE also outperforms static methods, we implemented four state-of-the-art methods for estimating static causal effects and evaluated them on our synthetic and semi-synthetic datasets. Here, we follow [17]: we consider DragonNet [31], TARNet [30], double machine learner (DML) [6], and doubly robust learner (DR) [7]. The latter two methods are meta-learners, which we instantiate via causal forests [39]. Results are in Table 4. Overall, DeepACE is superior by a large margin.

C IMPLEMENTATION OF BASELINES

We provide a detailed overview of our baselines. Of note, our baselines represent state-of-the-art methods for causal effect estimation in longitudinal settings [4, 15, 16, 33].

C.1 G-methods

Generalized methods (g-methods) [19] are a class of statistical models for time-varying ACE estimation originally used in epidemiology [27]. G-methods can be loosely categorized into marginal structural models, G-computation via the G-formula, and structural nested models. We use models of all three categories.

C.1.1 Marginal structural model (MSM) [27]. MSMs express the expected potential outcome θ^a directly as a function of the treatment intervention $\bar{a}_T = (a_1, \dots, a_T)$. In our case, we consider the model

$$\theta^a = \beta_0 + \sum_{t=1}^{T} \beta_t a_t \quad (29)$$

with parameters $\beta = (\beta_0, \dots, \beta_T)$. The parameters β can be estimated via inverse probability weighting. More precisely, the stabilized inverse probability weight for patient i is defined as

$$SW_i = \frac{\prod_{t=1}^{T} f_{A_t|\bar{A}_{t-1}}\left(A_t^{(i)}, \bar{A}_{t-1}^{(i)}\right)}{\prod_{t=1}^{T} g_{A_t|\mathcal{H}_t}\left(A_t^{(i)}, \bar{A}_{t-1}^{(i)}, \bar{X}_t^{(i)}\right)}, \quad (30)$$

where $f_{A_t|\bar{A}_{t-1}}$ and $g_{A_t|\mathcal{H}_t}$ denote the conditional densities of A_t given $\bar{A}_{t-1}$ and $\mathcal{H}_t = (\bar{A}_{t-1}, \bar{X}_t)$, respectively. Both conditional densities can be estimated via standard logistic regressions.

Robins [27] showed that the parameters β of the MSN can be consistently estimated by performing a weighted linear regression. Here, the observed treatments A are regressed on the observed outcomes Y_{T+1} and each observation i is weighted by SW_i. Once the estimates $\hat{\beta}$ are obtained, we estimate the ACE via

$$\hat{\psi} = \sum_{t=1}^{T} \hat{\beta}_t (a_t - b_t), \quad (31)$$

where $\bar{a}_T$ and $\bar{b}_T$ denote the treatment interventions.

C.1.2 G-computation [26, 35]. The G-formula from Eq. (3) can be used to estimate θ^a in an iterative manner. We include Algorithm 1 as a baseline in which we use linear regression to estimate the conditional expectations.

An equivalent way to write the G-formula from Eq. (3) is

$$\theta^a = \int_{\mathbb{R}^{p\times\cdots\times p}} \mathrm{E}\left[Y_{T+1} \mid \bar{X}_T = \bar{x}_T, \bar{A}_T = a\right] \prod_{t=1}^{T} f_{X_t|\bar{X}_{t-1},\bar{A}_{t-1}}\left(x_t \mid \bar{x}_{t-1}, \bar{a}_{t-1}\right) \,\mathrm{d}\bar{x}_T, \quad (32)$$

where $f_{X_t|\bar{X}_{t-1},\bar{A}_{t-1}}$ denotes the conditional density of X_t given $\bar{X}_{t-1}$ and $\bar{A}_{t-1}$. The conditional densities and the conditional expectation $\mathrm{E}\left[Y_{T+1} \mid \bar{X}_T = \bar{x}_T, \bar{A}_T = a\right]$ can be estimated by parametric regression models and are subsequently plugged into Eq. (32). This method is also known as *parametric G-computation* [26]. We use the algorithm from [18] as another baseline. Here, the conditional densities are essentially modeled as conditional normal distributions, where both mean and variance are estimated by generalized linear regression models.

Table 4: Results for state-of-the-art baselines for treatment effect estimation in static settings (mean ± standard deviation).

Method	Synthetic data			Semi-synthetic data		
	Setup 1	Setup 2	Setup 3	Setup 1	Setup 2	Setup 3
DragonNet [31]	0.15 ± 0.03	0.51 ± 0.14	1.17 ± 0.49	2.40 ± 1.57	3.75 ± 2.37	2.37 ± 1.85
TARNet [30]	0.12 ± 0.02	0.46 ± 0.17	1.17 ± 0.49	0.34 ± 0.16	0.44 ± 0.18	0.59 ± 0.35
DML (causal forest) [6, 39]	0.17 ± 0.09	0.43 ± 0.32	0.66 ± 0.77	0.78 ± 1.24	0.54 ± 0.53	1.06 ± 0.08
DR (causal forest) [7, 39]	0.16 ± 0.05	0.43 ± 0.35	0.61 ± 0.86	0.55 ± 0.47	0.77 ± 1.24	0.78 ± 1.24
DeepACE (ours)	**0.04 ± 0.02**	**0.09 ± 0.07**	**0.18 ± 0.08**	**0.18 ± 0.14**	**0.12 ± 0.14**	**0.16 ± 0.10**

lower = better (best in bold)

C.1.3 *Structural nested mean model (SNMM) [24].*

A structural nested mean model specifies the marginal effects of the treatment interventions at each time step. In our case, we consider the model given by

$$\mathrm{E}\left[Y_{T+1}(\bar{a}_t, 0) - Y_{T+1}(\bar{a}_{t-1}, 0)\right] = \beta_t a_t \tag{33}$$

for all $t \in \{1, \dots, T\}$ with parameters $\beta = (\beta_1, \dots, \beta_T)$. Here, $Y_{T+1}(\bar{a}_t, 0)$ denotes the potential outcome that is observed if the treatment intervention $\bar{a}_t$ is applied until time t, and no treatment is applied afterwards.

SNMM uses a method called g-estimation [26] to estimate the model parameters β. G-estimation is based on solving certain estimating equations, which is implied by Eq. (33) in combination with the assumptions from Sec. 3.1. For our experiments, we use the g-estimation method from [37] to obtain estimates $\hat{\beta}$.

C.2 LTMLE

LTMLE [33] extends iterative G-computation (Algorithm 1) by targeting the estimates $\hat{Q}_t^a$ after each iteration step t. For a detailed description, we refer to [33] and [35].

In our experiments, we consider two different variants of LTMLE: variant (i) estimates the conditional expectations Q_t^a and the propensity scores g_t with generalized linear models, and variant (ii) applies the super learner algorithm [34], which uses k-fold cross-validation to find an optimal weighted combination of machine learning base models. We set $k = 3$ and use a generalized linear model, random forest, xgboost, and a generalized additive model (GAM) with regression splines as base models.

C.3 Deep learning for ITE estimation

We included state-of-the-art baselines that predict future potential outcomes conditional on the patient trajectories $\mathcal{H}_t$ using deep learning. These are RMSNs [16], CRN [4] and G-Net [15]. To this end, we implemented all models as described in the respective references. Of note, these baselines aim at predicting individual counterfactual outcomes rather than estimating average causal effects. Because of that, we then obtain the ACE by averaging the respective predicted outcomes and subsequently subtracting them. For hyperparameter tuning, we refer to Appendix D.

C.3.1 *RMSNs [16] and CRN [4].*

Both baselines are based on a encoder-decoder architecture, and consider the setting where treatment interventions $\bar{a}_{(t,t+\tau-1)} = (a_t, \dots, a_{t+\tau-1})$ are applied from some time t to $t + \tau - 1$. The encoder builds a representation Φ_t of the patient trajectory $\mathcal{H}_t$. This is then used by the decoder together with the treatment intervention $\bar{a}_{(t,t+\tau-1)}$ to predict the future potential outcome $Y_{t+\tau}(\bar{a}_{(t,t+\tau-1)})$. In the encoder, we set $\mathcal{H}_1 = X_1$ as input so that treatment interventions can span the complete time frame from $t = 1$ to $\tau = T$.

RMSNs address time-varying confounding by re-weighting the training loss using inverse probability of treatment weights similar to Eq. (30). These weights are estimated using two separate LSTMs for nominator and denominator. In contrast, CRN adopts adversarial training techniques to build balanced representations that are non-predictive of the treatment.

C.3.2 *G-Net [15].*

G-Net uses the G-formula from Eq. (32) conditioned on the history $\mathcal{H}_1$ to estimate the outcomes $Y_{T+1}(\bar{a}_T)$ via Monte Carlo sampling (we use 100 samples). For this purpose, the conditional densities $f_{X_t|\bar{X}_{t-1}, \bar{A}_{t-1}}$ are estimated by learning the corresponding conditional expectations $\mathrm{E}[X_t|\bar{X}_{t-1}, \bar{A}_{t-1}]$ via an LSTM-based model. One can then sample from $f_{X_t|\bar{X}_{t-1}, \bar{A}_{t-1}}$ by drawing from the empirical distributions of the residuals on some holdout set that is not used to estimate the conditional expectations. We used 20 % of the training data for the holdout dataset.

D HYPERPARAMETER TUNING

We use Pytorch lightning for our implementation. We performed hyperparameter tuning for all deep learning model (including DeepACE) on all datasets by splitting the data into a training set (80%) and a validation set (20%). We then performed 30 random grid search iterations and chose the set of parameter that minimized the factual MSE on the validation set. The hyperparameter search ranges are shown in Table 5.

Table 5: Hyperparameter tuning ranges.

Hyperparameter	Tuning range
Hidden layer size(es)	p, $2p$, $3p$, $4p$
Learning rate	$\sim$ log-unif (0.0001, 0.001)
Batch size	64, 128
Dropout probability	0, 0.1, 0.2, 0.3

p = network input size

After training models with optimal hyperparameters, we used the full datasets to estimate the ACEs. Each model is trained over 100 epochs with the Adam optimizer [14]. During training, all LSTM networks use variational dropout [8]. We set the regularization parameters to $\alpha = 0.1$ and $\beta = 0.05$. Note that it is infeasible to include α or β into the hyperparameter tuning process because we only have access to the factual data (and, hence, α and β may shrink to zero in order to minimize the factual loss). In doing so, we are consistent with several state-of-the-art methods for causal effect estimation [4, 31].