G-Net: A Deep Learning Approach to G-computation for Counterfactual Outcome Prediction Under Dynamic Treatment Regimes

Rui Li\(^1\), Zach Shahn\(^{2,3}\), Jun Li\(^3\), Mingyu Lu\(^1\), Prithwish Chakraborty\(^{2,3}\), Daby Sow\(^{2,3}\), Mohamed Ghalwash\(^{2,3}\), Li-wei Lehman\(^1\)

\(^1\)Massachusetts Institute of Technology
\(^2\)IBM Research, \(^3\)MIT-IBM Watson AI Lab

Abstract

Counterfactual prediction is a fundamental task in decision-making. G-computation is a method for estimating expected counterfactual outcomes under dynamic time-varying treatment strategies. Existing G-computation implementations have mostly employed classical regression models with limited capacity to capture complex temporal and nonlinear dependence structures. This paper introduces G-Net, a novel sequential deep learning framework for G-computation that can handle complex time series data while imposing minimal modeling assumptions and provide estimates of individual or population-level time-varying treatment effects. We evaluate alternative G-Net implementations using realistically complex temporal simulated data obtained from CVSim, a mechanistic model of the cardiovascular system.

1 Introduction

Counterfactual prediction is a fundamental task in decision-making. It entails the estimation of expected future trajectories of variables of interest under alternative courses of action (or treatment strategies) given observed history. Treatment strategies of interest are usually time varying (meaning they comprise decisions at multiple time points) and dynamic (meaning the treatment decision at each time point is a function of history up to that time point).

As an example, consider the problem of fluid administration in Intensive Care Units (ICUs)\(^1\). It is frequently necessary for physicians to adopt strategies that administer large volumes of fluids to increase blood pressure and promote blood perfusion through organs in septic patients. However, such strategies can lead to fluid overload, which can have serious adverse downstream effects such as pulmonary edema. Fluid administration strategies are time varying and dynamic because at each time point physicians decide the volume of fluid to administer based on observed patient history (e.g. blood pressure and volume of fluid already administered) up to that time point. To aid in the choice between alternative dynamic fluid administration strategies, it would be desirable to obtain counterfactual predictions of a patient’s probability of developing fluid overload (and other outcomes of interest) were they to follow each alternative strategy going forward given their observed covariate history up to the current time.

Counterfactual prediction is an inherently causal task in that it must account for the causal effects of following different treatment strategies. When treatment strategies of interest are time-varying, so-called “g-methods”\(^2\)\(^3\) are required to estimate their effects. G-methods include g-computation\(^4\)-\(^6\), structural nested models\(^7\)\(^8\), and marginal structural models\(^9\)\(^10\). Of these methods, g-computation is best suited for estimating effects of general dynamic treatment strategies conditional on high dimensional patient histories\(^11\).

G-computation works by estimating the conditional distribution of relevant covariates given covariate and treatment history at each time point, then producing Monte Carlo estimates of counterfactual outcomes by simulating forward patient trajectories under treatment strategies of interest. Regression model(s) for the covariates and outcomes at each time point conditional on observed history are a critical component of this method. While any regression models could in theory be input to the G-computation algorithm, most existing G-computation implementations have employed simple regression models with limited capacity to capture complex temporal and nonlinear dependence structures. In recent years, sequential deep learning methods such as Recurrent Neural Networks (RNNs) have achieved state of the art performance in predictive modeling of complex time series data while imposing minimal
modeling assumptions. In this paper, we propose G-Net, a sequential deep learning framework for G-computation. G-Net admits the use of recurrent networks such as LSTMs to model covariates in a manner suitable for G-computation. The G-Net framework is flexible and allows for various configurations depending on the problem at hand. To the best of our knowledge, this is the first work to investigate a RNN based approach to G-computation.

Unfortunately, it is impossible to reliably evaluate counterfactual predictions on real data, since only the outcomes corresponding to treatment strategies that were actually followed can be observed. Consequently, to explore and evaluate various implementations of G-Net, we used simulated data in which counterfactual ground truth can be known. We used CVSim [12], a well established mechanistic model of the cardiovascular system, to estimate counterfactual simulated patient trajectories under various fluid and vasopressor administration strategies. These experiments provide a template for causal model evaluation using complex and physiologically realistic simulated longitudinal data.

2 Related Work

Several recent works have proposed a deep learning framework for counterfactual prediction from observational data, including [13,15]. However, these works have mostly focused on learning point exposure as opposed to time-varying treatment effects, which are the focus of this paper.

G-computation for estimating time-varying treatment effects was first proposed by Robins [3]. Illustrative applications of the general approach are provided in [6,16], and summaries of g-computation (and other “g-methods” for estimating time-varying treatment effects) can be found in [2,3]. The g-computation algorithm takes arbitrary regression models as inputs. While most applications (e.g. [6,16]) have thus far employed classical generalized linear models, there is no conceptual barrier to using more complex machine learning regression models. RNNs, and in particular LSTMs, have achieved state of the art performance on a wide variety of time series regression tasks, including healthcare related tasks [17,19]. However, despite the popularity and success of RNNs for time series regression, we have not seen in the literature any “deep” implementation of g-computation.

Recently, Lim et al. [20] plugged RNN regression models into history adjusted marginal structural models (MSM) [21] to make counterfactual predictions. However, these MSMs can only make counterfactual predictions under static time-varying treatment strategies that do not depend on recent covariate history. For example, a history adjusted MSM [20] could estimate the probability of fluid overload given patient history under the (static) treatment strategy “give 1 liter fluid each hour for the next 3 hours”, but it could not estimate the probability of fluid overload given patient history under the (dynamic) treatment strategy “each hour for the next 3 hours, if blood pressure is less than 65 then give 1 liter fluids, otherwise give 0 liters”. History adjusted MSMs cannot estimate effects of time-varying treatment strategies that respond to changes in the patient’s health history, but g-computation can. Further, g-computation is able to straightforwardly estimate the distribution of a counterfactual outcome under a time-varying treatment strategy. This is not straightforward to do with history adjusted MSMs.

Schulam and Saria [22] propose Counterfactual Gaussian Processes, an implementation of continuous time g-computation, and like us apply their method to ICU data. They only consider static time-varying treatment strategies, though it appears that their method might straightforwardly be extended to handle dynamic strategies as well. An advantage of Gaussian processes is interpretable uncertainty quantification. However, GPs are intractable for large datasets since they have time complexity of $O(N^3)$, where $N$ is the number of observations [23]. Sparse GPs, which introduce $M$ inducing points, have at least $O(M^2N)$ time complexity. RNNs are more scalable. Recurrent dropout can also be employed in RNN based implementations to produce uncertainty estimates that approximate posterior distributions from a Gaussian process [24].

3 G-computation for counterfactual prediction

Our goal is to predict patient outcomes under various future treatment strategies given observed patient histories. Let:

- $t \in \{0, \ldots, K\}$ denote time, assumed discrete, with $K$ being the end of followup;
- $A_t$ denote the observed treatment action at time $t$;
- $Y_t$ denote the observed value of the outcome at time $t$;
- $L_t$ denote a vector of covariates at time $t$ that may influence treatment decisions or be associated with the outcome;
- $\bar{X}_t$ denote the history $X_0, \ldots, X_t$ and $\bar{X}_t$ denote the future $X_t, \ldots, X_K$ for arbitrary time varying variable $X$.

At each time point, we assume the causal ordering $(L_t, A_t, Y_t)$. Let $H_t \equiv (L_t, A_{t-1})$ denote patient history preceding treatment at time $t$. A dynamic treatment strategy $g$ is a collection of functions $\{g_0, \ldots, g_K\}$, one per time point, such that $g_t$ maps $H_t$ onto a treatment action at time $t$. A simple dynamic strategy for fluid volume might be $g_t(H_t) = .5 \times 1\{bp_t < 65\}$, i.e. give .5 liters of fluid if mean arterial blood pressure is less than 65 at time $t$.

Let $Y_t(g)$ denote the counterfactual outcome that would be observed at time $t$ had, possibly contrary to fact, treatment strategy $g$ been followed from baseline [4]. Further, let $Y_t(A_{m-1}, g_m)$ with $t \geq m$ denote the counterfactual
outcome that would be observed had the patient received their observed treatments $\bar{A}_{m-1}$ through time $m - 1$ then followed strategy $g$ from time $m$ onward.

In counterfactual point prediction, our goal is to estimate expected counterfactual patient outcome trajectories

$$\{E[Y_t(\bar{A}_{m-1}, g_m)|H_m], t \geq m\} \quad (1)$$

given observed patient history through time $m$ for any $m$ and any specified treatment strategy $g$. We might also be interested in estimating the counterfactual outcome distributions at future time points

$$\{p(Y_t(\bar{A}_{m-1}, g_m)|H_m), t \geq m\}. \quad (2)$$

If we do not condition on anything in $H_m$, then (1) is an expectation (and (2) a distribution) over the full population. If we condition on a small subset of variables contained in patient history, then (1) is an expectation (and (2) a distribution) over a sub-population. If we condition on all elements of a patient history, then (1) is still technically only an expectation (and (2) a distribution) over a hypothetical sub-population with the exact patient history conditioned on, but in this case (1) and (2) practically amount to what is usually meant by personalized prediction.

Under the below standard assumptions, we can estimate (1) and (2) through g-computation (4).

1. Consistency: $\bar{Y}_K(\bar{A}_K) = \bar{Y}_K$
2. Sequential Exchangeability: $Y_t \perp A_t|H_t \quad \forall t$
3. Positivity: $P(A_t = a(H_t)) > 0 \forall \{H_t : P(H_t) > 0\}$

Assumption 1 states that the observed outcome is equal to the counterfactual outcome corresponding to the observed treatment. Assumption 2 states that there is no unobserved confounding of treatment at any time and any future outcome. Assumption 2 would hold, for example, under the conditions depicted in Figure 1. Positivity states that the counterfactual treatment strategy of interest has some non-zero probability of actually being followed. Under the assumption that we specify certain predictive models correctly such that their predictions extrapolate to parts of a joint distribution that they were not trained on, positivity is not strictly necessary.

Under assumptions 1-3, for $t = m$ we have simply that

$$p(Y_m(\bar{A}_{m-1}, g_m)|H_m) = p(Y_m|H_m, A_m = g_m(H_m)), \quad (3)$$

i.e. the conditional distribution of the counterfactual is simply the conditional distribution of the observed outcome given patient history and given that treatment follows the strategy of interest. For $t > m$, things are slightly more complex because we need to adjust for time-varying confounding. With $X_{t,j} = X_{1, t}, \ldots, X_{j}$ for any random variable $X$, under Assumptions 1-3 the g-formula yields

$$p(Y_t(\bar{A}_{m-1}, g_m) = y|H_m) = \int_{A_{m+1:t}} p(Y_t = y|H_m, L_{m+1:t} = l_{m+1:t}, A_{m+1:t} = g(H_{m+1:t})) \times \prod_{j=m+1}^{t+1} p(L_j = l_j|H_m, L_{m+1:j-1} = l_{m+1:j-1}, A_{m,j-1} = g(H_m, l_{m+1:j-1})). \quad (4)$$

It is not generally possible to compute this integral in closed form, but it could be approximated through Monte-Carlo simulation. We repeat the recursive process shown in Algorithm [1] $M$ times. (Here the outcome $Y_t$ is without loss of generality deemed to be a variable in the vector $L_{t+1}$.)

At the end of this process, we have $M$ simulated draws of the counterfactual outcome for each time $t = \{m, \ldots, K\}$. For each $t$, the empirical distribution of these draws constitutes a Monte-Carlo approximation of the counterfactual outcome distribution (2). The sample averages of the draws at each time $t$ are an estimate of the conditional expectations (1) and can serve as point predictions for $Y_t(\bar{A}_{m-1}, g_m)$ in a patient with history $H_m$.

Key to the g-computation algorithm is the ability to simulate from joint conditional distributions $p(L_t|L_{t-1}, \bar{A}_{t-1})$ of the covariates given patient history at time $t$. Of course, in practice we do not have prior knowledge of these conditional distributions and need to estimate them from data. Most implementations use generalized linear regression models to estimate the conditional distributions of the covariates. Often, these models do not capture temporal dependencies present in the patient data. We propose the G-Net for this task.

## 4 G-Net Design

The proposed G-Net framework depicted in Figure [2] enables the use of sequential deep learning models to estimate conditional distributions $p(L_t|L_{t-1}, \bar{A}_{t-1})$ of covariates given history at each time point and perform the G-
Algorithm 1: G-Computation (One simulation)

1. Set \( a_m^* = g_m(H_m) \)
2. Simulate and record \( y_m^* \) from \( p(Y_m | H_m, A_m = a_m^*) \)
3. Simulate \( l_{m+1}^* \) from \( p(L_{m+1} | H_m, A_m = a_m^*) \)
4. Set \( a_{m+1}^* = g_m(H_m, l_{m+1}^*, a_m^*) \)
5. Simulate and record \( y_{m+1}^* \) from \( p(Y_{m+1} | H_m, L_{m+1} = l_{m+1}^*, A_m = a_m^*, A_{m+1} = a_{m+1}^*) \)
6. Simulate \( l_{m+2}^* \) from \( p(L_{m+2} | H_m, L_{m+1} = l_{m+1}^*, A_m = a_m^*, A_{m+1} = a_{m+1}^*) \)
7. Continue simulations through time \( K \)

computation algorithm described in Algorithm 1 to simulate covariates under various treatment strategies. In this setting, without loss of generality, we set \( Y_t \) as one of the co-variates in \( L \) for notational simplicity.

Let \( L_0, \ldots, L_{t-1} \) denote \( p \) components of the vector \( L_t \), where each component \( L_j \) could be multivariate. We impose an arbitrary ordering \( L^0, L^1, L^2, \ldots, L^p, A_1, A_2, \ldots, A_k \) and estimate the conditional distributions of each \( L_j \) given all variables preceding it in this ordering. At simulation time, we exploit the basic probability identity

\[
p(L_t | L_{t-1}, \hat{A}_{t-1}) = p(L_t^0 | L_{t-1}, \hat{A}_{t-1}) \times p(L_t^1 | L_{t-1}^0, L_{t-1}, \hat{A}_{t-1}) \times \cdots \times p(L_t^p | L_{t-1}^{p-1}, L_{t-1}, \hat{A}_{t-1})
\]

to simulate from \( p(L_t | L_{t-1}, \hat{A}_{t-1}) \) by sequentially simulating each \( L_j \) from \( p(L_t^j | L_{t-1}^0, \ldots, L_{t-1}^{j-1}, L_{t-1}, \hat{A}_{t-1}) \).

To obtain the estimates for the conditional distribution of covariates, G-Net admits sequential modeling of patients’ data using the group decomposition and arbitrary ordering outlined before. As shown in the yellow boxes in Figure 2 at each time \( t \), a representation \( R_t \) of the patient history can be computed as

\[
R_t = r_t(L_t, \hat{A}_t; \Theta)
\]

where \( \Theta \) represents model parameters learned during training.
tion passing the covariates. In other configurations (e.g., using the selector in Figure 2), \( r_t \) can be used to provide abstractions of histories using sequential learning architectures such as RNN. This formulation of \( r_t \) allows for a great deal of flexibility in how information is shared across variables and time.

Estimates from each of the \( p \) covariate groups can then be obtained, as shown in Figure 2 by successive estimation of conditional expectations of covariates. Specifically, the conditional expectation of each \( L_{t+1}^j, 0 \leq j < p \) given the representation of patient history \( R_t \) and the other variables from time \( t+1 \) that precede it in the arbitrary predefined ordering is estimated by the functions \( f_t^j \). The complete sequence can be given as follows:

\[
L_{t+1}^0 = f_t^0(R_t; \Lambda_0) \\
L_{t+1}^1 = f_t^1(R_t, L_{t+1}^0; \Lambda_1) \\
\vdots \\
L_{t+1}^j = f_t^j(R_t, [L_{t+1}^1, \ldots, L_{t+1}^{j-1}]; \Lambda_j) \\
\vdots 
\]

(7)

where, each of the \( f_t^j \) represents the specialized estimation function for group \( j \) and \( \Lambda_j \) are the learnable parameters for the same. Depending on modeling choices, \( f_t^j \) can be sequential models specialized for group \( j \) or models focusing only on the time step \( t \) (e.g. linear models).

Given G-Net parameters, the distribution of the Monte Carlo simulations produced by G-computation Algorithm 1 constitute an estimate of uncertainty about a counterfactual prediction. But this estimate ignores uncertainty about the G-Net parameter estimates themselves. One way to incorporate such model uncertainty would be to fit a Bayesian model and, before each Monte Carlo trajectory in G-computation, draw new network parameters from their posterior distribution. These Monte Carlo draws would be from the posterior predictive distribution of the counterfactual outcome. Bayesian deep learning can be prohibitively computationally intensive, but can be approximated through dropout [24]. First, we fit the RNN component of the G-Net using recurrent dropout, with dropout masks at each layer held constant across time as described in [24]. Let \( M \) denote a dropout mask constant across time for each layer of the RNN, \( \Theta(M) \) denote estimated RNN parameters with mask \( M \) applied, and \( p_M \) denote the distribution from which \( M \) was sampled during training. Then, during g-computation, we add step 0 to the beginning of Algorithm 1 before each simulation:

**Step 0 (dropout)**: Sample \( D^* \sim p_D \) and compute all simulations plugging \( E \Theta(D) | L_t^0, L_t^1, \ldots, L_t^{j-1}, L_{t-1}^1, \Lambda_{t-1} \) into (5). Then draws of \( \Theta(D) \) are from an approximation to the posterior distribution of \( \Theta \) under a particular Gaussian Process prior.

Therefore, the Monte Carlo simulations obtained from g-computation incorporating dropout (i.e. adding step 0 as above) approximate draws from the posterior predictive distribution of the counterfactual outcome.

The parameters, \( \Theta \) and \( \Lambda \), are learned by optimizing a loss function forcing the G-Net to accurately estimate covariates \( L_t \) at each time point \( t \) using standard gradient descent techniques. It is to be noted that it is necessary to use teacher-forcing (using observed values of \( L_{t+1}^j \) as arguments to \( f_t \) in equation ) to obtain unbiased estimates of \( f_t \), as shown in Figure 2.

**5 Simulation Experiments Using CVSim**

To evaluate counterfactual predictions, it is necessary to use simulated data in which counterfactual ground truth for outcomes under alternative treatment strategies is known. To this end, we performed experiments on data generated by CVSim [25], a program that simulates the dynamics of the human cardiovascular system.

**Data Generation:** We generated an ‘observational’ dataset \( D_o \) under treatment regime \( g_o \) and two ‘counterfactual’ datasets \( D_{c1} \) and \( D_{c2} \) under treatment regimes \( g_{c1} \) and \( g_{c2} \). The data generating processes producing \( D_o \) and \( D_{c1} \) were the same except for the treatment assignment rules. For each \( j \), \( g_{cj} \) was identical to \( g_o \) for the first \( m - 1 \) simulation time steps before diverging to a different treatment rule for time steps \( m \) to \( K \) as illustrated in Figure 3.

A CVSim 6-compartment circulatory model takes as inputs 28 variables that together govern a hemodynamic system. It then deterministically simulates forward in time a set of
25 output variables according to a collection of differential equations (parameterized by the input variables) modeling hemodynamics. Important variables in CVSim include arterial pressure (AP), central venous pressure (CVP), total blood volume (TBV), and total peripheral resistance (TPR). In real patients, physicians observe AP and CVP and seek to keep them above a clinically safe threshold. They do this by intervening on TBV (through fluid administration) and TPR (through vasopressors).

We defined simulated treatment interventions that were designed to mimic the impact of fluids and vasopressors. These simulated interventions alter the natural course of the simulation by increasing either TBV (in the case of the simulated fluids intervention) or TPR (in the case of the simulated vasopressor intervention). We generated patients by randomly initiating baseline inputs (which we hid from our G-Nets to make this a stochastic modeling problem) within plausible physiologic ranges, then using CVSim to simulate covariates forward, intervening according to the relevant treatment strategy at each timestep. Full details of the simulation process can be found in the Appendix.

Under (stochastic) observational treatment strategy \( g_o \), the probability of receiving a non-zero vasopressor or fluid dose at a given time increases as MAP and CVP decrease according to a logistic regression function. Given that a dose is non-zero, the exact amount is drawn from a normal distribution with mean inversely proportional to MAP and CVP. Since all drivers of treatment under \( g_o \) are observed in our data, the sequential exchangeability assumption holds and g-computation may be validly applied.

\( g_{c1} \) is similar to \( g_o \), except it is a deterministic treatment strategy and the functions linking treatment and dose to covariates have different coefficients. Under \( g_{c2} \), treatment is always withheld. Again, details are in the Appendix.

**Experiment Setup:** We set ourselves the task of training a G-Net on \( D_o \) and using it to predict the trajectories of patients in \( D_{c,j} \) for time steps \( m \) to \( K \) for each \( j \). This setup is designed to evaluate the performance of a G-Net in a situation in which we observe data from past patients (\( D_o \)) who received usual care (\( g_o \)) for \( K \) timesteps and would like to predict how a new patient who has been observed for \( m \) timesteps would fare were they to follow a different treatment strategy of interest (\( g_{c,j} \)) for timesteps \( m \) to \( K \). This is a standard use case for counterfactual prediction. \( D_{c,j} \) provides ground truth for a collection of patients whose trajectories follow just the path we are interested in predicting. Thus, by aggregating predictive performance metrics across simulated patients in \( D_{c,j} \), we generate measures of the population level performance of our G-Net at the counterfactual prediction task for which it was intended.

As shown in Table 1, we explore two specific criteria: (a) using sequential vs. Identity functions for \( r_t \) and (b) using sequential models of entire patient history vs. linear models focusing only on current time point for \( f_t \). This provides us 4 different implementations of G-Net.

The best parameters for each model found from grid search are as follows. For \( M2(LSTM) \), the hidden dimension for the representational LSTM is 30, the hidden dimension for the categorical LSTM is 5, and the learning rate is .001. For \( M3(LSTM) \), the hidden dimension for the categorical LSTM is 10, the hidden dimension for the continuous LSTM is 30, and the learning rate is .005. For \( M4(LSTM) \), the hidden dimension for the representational LSTM is 75, and the learning rate is .005. For all models, the batch size used was 64. These parameters were the ones used to achieve the results presented in the experiments and results section.

**Evaluation:** We evaluate the accuracy (Mean Squared Error - MSE) and calibration of the counterfactual simulations generated by our G-Nets as follows. Say \( D_{c,j} \) comprises \( N_c \) trajectories of random variable \((L^c_{ij}, A^{c}_{ij})\). Given observed history \( H^{c}_{mi} = (L^{c}_{mi}, A^{c}_{m-1i}) \) for patient \( i \) from

| With pass thru \( r_t \) | With sequential \( r_t \) |
|-------------------------|-------------------------|
| \( f_c:LR \) | \( f_c:RNN \) |
| \( M1(Linear) \) | \( M3(LSTM) \) |
| \( \tau_t: \) Identity | \( \tau_t: \) Identity |
| \( p = 2 \) | \( p = 2 \) |
| \( (f_0, f_1) : \) linear layers. | \( (f_0, f_1) : \) linear layers. |

| With pass thru \( r_t \) | With sequential \( r_t \) |
|-------------------------|-------------------------|
| \( f_c:LR \) | \( f_c:RNN \) |
| \( M2(LSTM) \) | \( M4(LSTM) \) |
| \( \tau_t: \) LSTM | \( \tau_t: \) LSTM |
| \( p = 2 \) | \( p = 2 \) |
| \( (f_0, f_1) : \) LSTMs. | \( (f_0, f_1) : \) LSTMs |

Figure 4: 100 G-Net simulated trajectories (blue) and ground truth (red) for one patient under \( g_{c1} \).
Figure 5: Performance of various models: MSE and calibration over time for \( g_{c1} \) and \( g_{c2} \)

\( D_{cj} \), a G-Net \( G \) fit to \( D_o \) produces \( M \) (in our experiments, 100) simulations of the counterfactual covariate trajectory \( \{ \tilde{L}_{t_i}^{h,cj}(H_{m1}^c, G, k) : t \in m : K ; k \in 1 : M \} \). These simulated trajectories are the light blue lines in Figure 4.

The G-Net’s point prediction of \( L_{t_i} \) is its estimate of \( E[L_{t_i}(g_{cj})|H_{m} = H_{m1}] \), i.e. the average of the \( M \) simulations \( \tilde{L}_{t_i}(G) = \frac{1}{M} \sum_{k=1}^{M} L_{t_i}^{h,cj}(H_{m1}^c, G, k) \). This is the dark blue line in Figure 4.

If \( L_{t_i} \) has dimension \( d \), we compute the MSE of counterfactual predictions by a G-Net \( G \) in the dataset \( D_c \) as

\[
\frac{1}{N_c(K-m)d} \sum_{i=1}^{N_c} \sum_{t=m}^{K} \sum_{h=1}^{d} (L_{t_i}^{h,CF} - \tilde{L}_{t_i}^{h,CF}(G))^2.
\]

We assess the calibration of a G-Net \( G \) as follows. Given lower and upper quantiles \( \alpha_{low} \) and \( \alpha_{high} \), the calibration measures the frequency with which the actual counterfactual covariate \( L_{t_i}^{h,cj} \) is between the \( \alpha_{low} \) and \( \alpha_{high} \) quantiles of the \( M \) simulations \( \{ \tilde{L}_{t_i}^{h,cj}(H_{m1}^c, G, k) : k \in 1 : M \} \). If this frequency is approximately \( \alpha_{high} - \alpha_{low} \), then \( G \) is well calibrated.

Experiments/Results: Using CVSim, we generated a total of 10,000 trajectories in \( D_o \) (\( N_o = 10,000 \)), of which 80% were used for training, and the remaining 20% for validation. For testing, we generated 500 observations in the \( D_{cj} \) datasets (\( N_c = 500 \)). We included a total of 18 output variables (including all variables influencing treatment assignment under \( g_o \)) from CVSim to construct \( D_o \) and \( D_{cj} \); each trajectory is of length 64 time steps (\( d=18, K=64 \)). In each \( D_{cj} \), the switching time point \( m \) from \( g_o \) to \( g_c \) is fixed at 34 for all trajectories (\( m = 34 \)).

We fit the four models described in Table 1 to the training portion of \( D_o \) (80%), and used the remaining portion as validation to tune our model hyperparameters. Next, given observed covariate history through 34 time steps and treatment history through 33 time steps of each trajectory in each \( D_{cj} \), we computed the MSE and calibration of the G-Nets’ counterfactual predictions for time steps 34 to 64.

Figure 5 (a) and (b) illustrates the performance of the various G-Net architectures in terms of MSE over time. Overall, for this experiment we found M3 (Identity function for \( r_{tf} \) and LSTM for the \( f_{ti} \) functions), performed best over both treatment strategies. Also, a comparative analysis of M1 vs M3 (both with Identity representation function for \( r_{tf} \)) and M2 vs M4 (both with LSTM representation functions for \( r_{tf} \)) showed that G-Net provide better estimates of conditional distributions by admitting sequential prediction models focusing on the entire patient history compared to prediction models focusing on a single time point.

Figure 5 (c) and (d) depicts calibration for the candidate models. All of the calibration coverage rates are below the nominal levels (they should be .5), though the RNN based
G-Nets again perform better than the linear model implementation. This is in part because the counterfactual predictive density estimates in these experiments do not take into account uncertainty about model parameter estimates, which could be addressed with dropout as discussed in Section 4.

The G-Net can also be used to estimate population average counterfactual outcomes and treatment effects, quantities sometimes more relevant to policy decisions than individual level counterfactual predictions. Figure 6 displays G-Net (M3 from Table 1) estimates and true values of population average trajectories for select variables under $g_{c:1}$. Figure 7 shows estimates and true values of the population average treatment effect of following $g_{c:1}$ as opposed to $g_{c:2}$ on select variables. We see that the G-Net does a good job of estimating these population level quantities of interest.

6 Discussion and Future Work

In the G-Net we have introduced a novel and flexible framework for counterfactual prediction under dynamic treatment strategies through G-computation using sequential deep learning models. We have illustrated several implementations of the G-Net in a realistically complex simulation setting where we had access to ground truth counterfactual outcomes. In particular, we considered alternative approaches to representation learning that either share representations of patient history across predictive tasks or keep them separate. In the particular implementation we considered, shared representations seemed to aid simple linear classifiers but harm LSTMs.

The G-Net framework’s flexibility means that there are many other alternative implementations to explore. For example, we might consider alternative architectures for representing patient history, such as attention mechanisms or memory networks.

Another direction of future work is incorporation of prior causal knowledge. For example, if it is known which variables are confounders and which merely predictive of the outcome, we might include weights in the loss function emphasizing faithful representation and prediction of confounding variables compared to non-confounders.

References

[1] Simon Finfer, John Myburgh, and Rinaldo Bellomo. Intravenous fluid therapy in critically ill adults. *Nature Reviews Nephrology*, 14(9):541, 2018.

[2] Miguel Hernan and James Robins. *Causal Inference*. Chapman and Hall, Forthcoming.

[3] James Robins and Miguel Hernan. Estimation of the causal effects of time varying exposures. In Garrett Fitzmaurice, Marie Davidian, Geert Verbeke, and
James Robins. 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*, 1986.

James Robins. A graphical approach to the identification and estimation of causal parameters in mortality studies with sustained exposure periods. *Journal of Chronic Diseases*, 1987.

Sarah Taubman, James Robins, Murray Mittleman, and Miguel Hernan. Intervening on risk factors for coronary heart disease: An application of the parametric g-formula. *International journal of epidemiology*, 2009.

James Robins. Correcting for non-compliance in randomized trials using structural nested mean models. *Communications in Statistics-Theory and Methods*, 1994.

Stijn Vansteelandt and Marshall Joffe. Structural nested models and g-estimation: The partially realized promise. *Statistical Science*, 2014.

James Robins, Miguel Hernan, and Babette Brumback. Marginal structural models and causal inference in epidemiology. *Epidemiology*, 2000.

Liliana Orellana, James Robins, and Andrea Rotnitzky. Dynamic regime marginal structural mean models for estimation of optimal dynamic treatment regimes. *The international journal of biostatistics*, 2008.

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

Thomas Heldt, Ramakrishna Mukkamala, George B Moody, and Roger G Mark. Cvsim: an open-source cardiovascular simulator for teaching and research. *The open pacing, electrophysiology & therapy journal*, 3:45, 2010.

Onur Atan, James Jordan, and Mihaela van der Schaar. Deep-treat: Learning optimal personalized treatments from observational data using neural networks. In *Proceedings of AAAI*, 2018.

M Ahmed Alaa, Michael Weisz, and Mihaela van der Schaar. Deep counterfactual networks with propensity-dropout. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

Jinsung Yoon, James Jordan, and Mihaela van der Schaar. Ganite: Estimation of individualized treatment effects using generative adversarial nets. In *ICLR*, 2018.

Jessica Young, Lauren Cain, James Robins, Eilis OR-eilly, and Miguel Hernan. Comparative effectiveness of dynamic treatment regimes: An application of the parametric g-formula. *Statistics in biosciences*, 2011.

Nenad Tomašev, Xavier Glorot, Jack W Rae, Michal Zielinski, Harry Askham, Andre Saraiva, Anne Mortram, Clemens Meyer, Suman Ravuri, Ivan Protsyuk, et al. A clinically applicable approach to continuous prediction of future acute kidney injury. *Nature*, 572 (7767):116, 2019.

Cao Xiao, Edward Choi, and Jimeng Sun. Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review. *Journal of the American Medical Informatics Association*, 25(10):1419–1428, 06 2018.

Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3504–3512. Curran Associates, Inc., 2016.

Bryan Lim, Ahmed Alaa, and Mihaela van der Schaar. Forecasting treatment responses over time using recurrent marginal structural networks. In *Neural Information Processing Systems (NIPS)*., 2018.

Mark van der Laan, Maya Petersen, and Marshall Joffe. History-adjusted marginal structural models and statically-optimal dynamic treatment regimens. *The international journal of biostatistics*, 2005.

Peter Schulam and Suchi Saria. Reliable decision support using counterfactual models. In *Neural Information Processing Systems (NIPS)*., 2017.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *International Conference on Artifical Intelligence and Statistics*, 2009.

Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*, 2016.

T Heldt, R Mukkamala, GB Moody, and RG Mark. CVSim: An open-source cardiovascular simulator for teaching and research. *Open Pacing, Electrophysiol & Ther J*, 3:45–54, 2010.

Andrew Rhodes, Waleed Evans, Laura E.and Alhazzani, Mitchell M. Levy, Massimo Antonelli, Ricardo Ferrer, Anand Kumar, Jonathan E. Sevransky, et al. Deep-treat: Learning optimal personalized treatments from observational data using neural networks. In *Proceedings of AAAI*, 2018.
Charles L. Sprung, Mark E. Nunnally, et al. Surviving sepsis campaign: International guidelines for management of sepsis and septic shock: 2016. Critical Care Medicine, 2017. URL https://www.ncbi.nlm.nih.gov/pubmed/28101605
A Appendix

A.1 CVSim

CVSim is an open-source cardiovascular simulator, aiming for education and research purpose developed by Thomas et al. In this work, we focus on CVSim-6C which consists of 6 components, functioning as pulmonary and systemic veins, arteries, and micro-circulations respectively. CVSim-6C is regulated by arterial baroreflex system to simulate Saha's lumped hemodynamic model. The aggregate model is capable of simulating pulsatile waveforms, cardiac output and venous return curves, and spontaneous beat-to-beat hemodynamic variability. In this work, we modified and built on CVSim by adding stochastic-components to it for the purposes of evaluating our counterfactual simulators. We call our stochastic simulation engine S-CVSim.

A.2 Inputs of S-CVSim

By varying hemodynamic parameters of CVSim, it can simulate cardiovascular system under various conditions. We, therefore, sample values of a subset of model parameters while initiating simulation at time 0 to obtain ideal distribution of trajectories. These model parameters are listed at Table 2. Note that this does not necessarily mean values of input covariates would not exceed or drop below such range at any time t, where t > 0.

Table 2: Names and corresponding ranges of input parameters of S-CVSim.

| INPUT COVARIATES                        | RANGE     |
|-----------------------------------------|-----------|
| Total Blood Volume                      | 1,500 - 6,000 |
| Nominal Heart Rate                      | 40 - 160  |
| Total Peripheral Resistance             | 0.1 - 1.4 |
| Arterial Compliance                     | 0.4 - 1.1 |
| Pulmonary Arterial Compliance           | 0.1 - 19.9|
| Total Zero-pressure filling Volume      | 500 - 3,500|
| Pulmonary Arterial Compliance           | 2.0 - 3.4 |
| Pulmonary Microcirculation Resistance   | 0.4 - 1.00|

A.3 Outputs of S-CVSim

At each time t, CVSim-6C generates hemodynamic data including vascular resistance and varieties of type of flow, pressure, and volume. In addition to original 25 hemodynamic outputs, we introduce 3 new outputs, including systolic blood pressure, diastolic blood pressure, and mean arterial pressure, bringing the total number of output variables to 28. For this work, we only predict a subset of output covariates highlighted in Table 3. A complete list of output are listed in Table 3.

- **Systolic Blood Pressure** (SBP) is defined as the highest measured arterial blood pressure while heart contracting.
- **Diastolic Blood Pressure** (DBP) is defined as the lowest measured arterial blood pressure while heart contracting.
- **Mean Arterial Pressure** (MAP) is defined as the average pressure in a patient’s arteries during one cardiac cycle as follow formula. $\text{MAP} = \frac{2\times\text{DBP} + \text{SBP}}{3}$

Table 3: Outputs of S-CVSim. $L_t$ denotes all the output of S-CVSim at time $t$; whereas, $L_t^{\text{obs}}$ denotes a subset of output e.g. $L_t^{\text{map}}$ represents mean arterial pressure at time t. Covariates highlighted with * are the selected output.

| OUTPUT COVARIATES                        |                      |
|-----------------------------------------|----------------------|
| Left Ventricle Pressure*                | LVP                  |
| Left Ventricle Flow*                    | LVQ                  |
| Left Ventricle Volume                   | LVV                  |
| Left Ventricle Contractility*           | LVC                  |
| Right Ventricle Pressure*               | RVP                  |
| Right Ventricle Flow*                   | RVQ                  |
| Right Ventricle Volume                  | RVV                  |
| Right Ventricle Contractility*          | RVC                  |
| Central Venous Pressure*                | CVP                  |
| Central Venous Flow                     | CVQ                  |
| Central Venous Volume                   | CVV                  |
| Arterial Pressure*                      | AP                   |
| Arterial Flow*                          | AQ                   |
| Arterial Volume*                        | AV                   |
| Pulmonary Arterial Pressure             | PAP                  |
| Pulmonary Arterial Flow                 | PAQ                  |
| Pulmonary Arterial Volume               | PAV                  |
| Pulmonary Venous Pressure               | PVP                  |
| Pulmonary Venous Flow                   | PVQ                  |
| Pulmonary Venous Volume*                | PVV                  |
| Heart Rate*                             | HR                   |
| Arteriolar Resistance*                  | AR                   |
| Venous Tone*                            | VT                   |
| Total Blood Volume*                     | TBV                  |
| Intra-thoracic Pressure*                | PTH                  |
| Mean Arterial Pressure*                 | MAP                  |
| Systolic Blood Pressure*                | SBP                  |
| Diastolic Blood Pressure                | DBP                  |

A.4 Simulation Process

To obtain observational data for our intended purpose, we implemented and built two extra events, Disease, $S_t$ and Treatment, $A_t$. Note that at each time $T$, both $S_t$ and $A_t$ could happen simultaneously but $S_t$ also happens before $A_t$. With these stochastic events, $A_t$ and $S_t$, $D_{obs}$ and $D_{CF}$
can be simulated and obtained on S-CVSim.

A.4.1 Data Generation, $D_o, D_c$

To simulate a patient trajectory under treatment strategy $g$, we obtain $D_o$ and $D_c$ with the following algorithm for each individual trajectory. $D_o$, training set, consists of 10k trajectories based on $g_o$. $D_c$, test set, consists of 1k trajectories based on $g_c$.

- Initialize input variables $V_1, \ldots, V_N$ by drawing from independent uniform distributions using a predefined plausible physiological ranges for each variable as Table 2 suggested.
- For $t$ in $0:K$
  - Generate $L_t^o$ as $F_{sim,t}(V, L_{t-1}^o, A_{t-1})$, where $A_{t-1}$ is taken to be 0.
  - If $g = g_c$ and $t \geq \frac{k}{2}$,
    * Generate $A_t$ as $g_c(L_t)$
  - Else
    * Generate $A_t$ as $g_o(L_t)$

A.4.2 Disease Simulation, $S_t$

We introduce the concept $S_t$ to simulate hemodynamic instability of cardiovascular system such as sepsis and bleeding at each timestep. $S_t$ consists of two events, sepsis and blood loss. In module of $S_t$, we denote that $P(S_t|L_t) = 0.05$, and $P(S_{\text{sepsis}}|S_t) = P(BloodLoss|S_t) = 0.5$; Note that blood loss and sepsis events are mutually exclusive in the simulation process at any time $t$.

- When sepsis happens, $L_{t+1}^{\text{pr}} = \alpha_{\text{pr}} * L_t^{\text{pr}}$ where $0 < \alpha_{\text{pr}} \leq 0.7$, meaning $L_t^{\text{pr}}$, total peripheral resistance, would decrease in $\alpha_{\text{pr}}$ at next time $t + 1$
- When blood loss happens, $L_{t+1}^{\text{bw}} = \alpha_{\text{bw}} * L_t^{\text{bw}}$ where $0 < \alpha_{\text{bw}} \leq 0.95$, meaning that $L_t^{\text{bw}}$, total blood volume would decrease in $\alpha_{\text{bw}}$ at next time $t + 1$.

A.4.3 Treatment Simulation, $A_t$

We developed two treatment strategy $g_o$, policy for observational regime and $g_c$, policy for counterfactual policy, to simulate clinical treatment and validate our model. Under any given $g$, $A_{t}^{\text{any}}$ is defined as $g(L_t)$. Similar to disease simulation, $A_{t}^{\text{any}}$ is either $A_{t}^{\text{pr}}$, increasing quantity of total blood volume or $A_{t}^{\text{bw}}$, increasing quantity of arterial resistance. The probability of choosing fluids is equal to vasopressors but will not be administered at the same time.

The dosage of $A_t$ depends on a subset of $L_t$ which indicates hemodynamic balance. More specific, since adequate blood pressure is an important clinical goal[26], we denote mean arterial pressure, MAP, of 65 mmHg and central venous pressure, CVP, of 10 mmHg as target goals. Therefore, $\Delta_{\text{map},t} \equiv 65 - L_t^{\text{map}}$ and $\Delta_{\text{cvp},t} \equiv 10 - L_t^{\text{cvp}}$ as proxies of how much dosage should be delivered. The following section will discuss the difference of $A_t$ between $g_{CF}$ and $g_{obs}$.

A.4.4 Observational Regime, $g_o$

Under $g_{obs}$, probability and dosage of treatment are denoted as the followings.

- Probability of treatment, $P(A_t|L_t) = \frac{1}{1 + e^{-x}}$, where $x = C_1 * \Delta_{\text{map}} + C_2 * \Delta_{\text{cvp}} + C_0$.
- If we administer fluids, we generate the dose (in mL) $A_1^t \sim \max(0, \beta_1^1 * \Delta_{\text{map},t} + \beta_2^1 * \Delta_{\text{cvp},t} + N(1500, 1000))$.
- If we administer vasopressors, we generate the dose $A_2^t \sim \max(0, \beta_1^2 * \Delta_{\text{map}} + \beta_2^2 * \Delta_{\text{cvp}} + N(0, 1))$ if $U \sim Uniform$.

A.4.5 Counterfactual Regime, $g_c$

Under $g_{CF}$, probability and dosage of treatment are denoted as the followings.

- Probability of treatment, $P(A_t|L_t) = 1$ if and only if $L_t^{\text{sbp}} \leq 100$ and ShockIndex $[26]$ $\frac{L_t^{\text{hr}}}{L_t^{\text{sbp}}} \leq 0.8$.
- If we administer fluids, we generate the dose (in mL) $A_1^t \sim \max(0, \beta_1^1 * \Delta_{\text{map},t} + \beta_2^1 * \Delta_{\text{cvp},t})$.
- If we administer vasopressors, we generate the dose $A_2^t \sim \max(0, \beta_1^2 * \Delta_{\text{map}} + \beta_2^2 * \Delta_{\text{cvp}})$ if $U \sim Uniform$.

We experimented with multiple parameters and opted to use $C0 = 0.02, C1 = 0.06, C2 = 0.24, \beta_1^1 = 10, \beta_1^2 = 60, \beta_2^1 = 0.1, \beta_2^2 = 0.15$.

Figure 8: Causal DAG for $D_o$ generated under observational regime $g_o$. Green arrows denote directed paths from treatments to outcomes. Treatment only depends on current covariates. All covariates with arrows into treatment (i.e. MAP, CVP, and $\Delta_P$) are observed, satisfying the sequential exchangeability assumption as in Figure 1.

The DAG in Figure 8 depicts the causal structure of $D_o$. Note that only observed covariates have arrows pointing into treatment (treatment is actually only a function of CVP, MAP, and $\Delta_P$), so that there is no unobserved confounding and g-computation may be applied.