A FOUNDATION MODEL FOR POLICY IN-CONTEXT LONGITUDINAL CAUSAL EFFECT ESTIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Longitudinal causal effect estimation is complicated by time-varying confounding and treatment-confounder feedback loops. Existing methods generally require data-specific nuisance-model fitting, and many are tailored to prespecified treatment sequences or prediction horizons, leading to time-consuming computation at deployment. We introduce *Policy In-Context Longitudinal causal Effect estimation with Prior-data Fitted Networks* (**PICLE-PFN**), the first foundation model for longitudinal causal effect estimation that supports *dynamic* treatment policies. PICLE-PFN represents target policies through in-context history-decision demonstrations, providing a unified interface for a broad class of treatment policies. Pretrained over synthetic data sampled from longitudinal structural causal models, PICLE-PFN amortizes causal effect estimation across datasets, policies, and horizons. Given a new observational dataset and target policy, it directly predicts conditional average potential outcomes (CAPOs) in one forward pass without task-specific training or hyperparameter tuning. Across synthetic and semi-synthetic longitudinal benchmarks, PICLE-PFN achieves comparable and lower RMSE and substantially improved uncertainty calibration, compared to state-of-the-art conventional estimators trained and tuned separately for each dataset.

1 INTRODUCTION

Clinical care often involves a sequence of treatment decisions that evolve with a patient's changing condition (Murphy, 2003; Chakraborty & Moodie, 2013). Estimating the effects of these time-varying treatments is therefore essential for guiding effective care. Doing so from observational data is particularly challenging: covariates can simultaneously be consequences of earlier treatments and determinants of later treatment decisions and outcomes, creating treatment–confounder feedback that complicates causal effect estimation (Robins, 1986; Hernán & Robins, 2010).

Existing methods decompose the longitudinal counterfactual target into a sequence of local nuisance regressions, which are fitted to each target dataset and combined through a causal estimation procedure (Robins, 1986; 2000; Bang & Robins, 2005). This decomposition introduces two practical challenges. First, estimation errors can propagate and accumulate over the intervention horizon (Chen et al., 2026a). Second, existing methods face a trade-off between flexibility across causal queries and estimation variance. Some estimators fit nuisance models that can be reused across many policies and horizons, but this flexibility can come at the cost of high variance, e.g., through products of inverse propensities or by modeling the full longitudinal data-generating process (Li et al., 2021; Frauen et al., 2025). Other estimators reduce variance by fitting only the outcome regressions needed for a particular policy and horizon, but changing the query then requires retraining and potentially new hyperparameter tuning (Hess et al., 2026; Shirakawa et al., 2024). Moreover, many existing methods (Lim, 2018; Bica et al., 2020; Melnychuk et al., 2022; Seedat et al., 2022) are designed for static treatment sequences, which apply prescribed fixed treatment decisions throughout the trajectory without considering time-varying covariates. Yet clinically relevant counterfactual questions may involve treatment regimes that adapt decisions to time-varying patient characteristics. There is a need for an estimator that supports both static and dynamic policies across different horizons, without retraining for each new target.

To address these challenges, we explore an alternative approach based on in-context learning (ICL). Recent tabular foundation models demonstrate that pretraining on synthetic tasks can produce models that adapt in context to new datasets and prediction problems without parameter updates (Holl-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: **Flexibility of existing methods:** RMSN (Lim, 2018); CRN (Bica et al., 2020); TE-CDE (Seedat et al., 2022); CT (Melnychuk et al., 2022); G-Net (Li et al., 2021); IGC-Net (Hess et al., 2026); DeepACE (Frauen et al., 2023); DeepLTMLE (Shirakawa et al., 2024); D^3-Net (Chen et al., 2026a). ✓ indicates that the capability is supported by the method's released implementation.

Capability	RMSN	CRN	TE-CDE	CT	G-Net	IGC-Net	DeepACE	DeepLTMLE	D^3-Net	**PICLE-PFN**
Dynamic treatment policy	✗	✗	✗	✗	✗	✓	✗	✓	✓	✓
Multidimensional treatment	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓
No retraining across horizons/policies	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓

mann et al., 2023; 2025; QU et al., 2025; 2026). This paradigm offers two opportunities for longitudinal causal estimation. First, in synthetic data, we can intervene on a target policy and directly generate its corresponding multi-step potential outcome. This provides ground-truth supervision for the final causal estimand, avoiding the need to learn and compose a sequence of local nuisance models. Second, diverse pretraining priors can support adaptation to different datasets and causal queries. The remaining gap is representing target policies, including dynamic ones, as context for ICL, and directly learning conditional average potential outcome (CAPO) across different horizons.

To meet these gaps, we introduce PICLE-PFN, the first foundation model for longitudinal CAPO estimation supporting both static and dynamic policies. Our contributions are three-fold: **(i) Policy representation and ICL formulation.** We show that a target policy's decisions on factual histories across the *entire* observed dataset preserve the policy information needed for CAPO identification. This motivates an ICL formulation that combines observational data with policy demonstrations, providing a unified interface for static and dynamic policies over multi-step intervention windows. **(ii) Architecture and amortized estimation.** We design a novel transformer based on Prior-data Fitted Network (PFN). Direct pretraining on synthetic multi-step CAPO targets avoids full generative modeling and explicit nuisance-model composition at deployment. Pretraining across diverse datasets, policies, and horizons enables flexible causal queries in a single forward pass without task-specific optimization. **(iii) Empirical performance.** PICLE-PFN achieves the lowest RMSE in six of seven benchmark settings, with up to 48.7% lower RMSE than the strongest baseline, while being approximately $5000\times$ faster at deployment than a comparably accurate method. It also provides substantially better calibrated uncertainty. Code is available at `https://anonymous.4open.science/r/PICLE-02FE/`.

2 Related Work

Tabular Foundation Model for Time Series and Causal Inference Recent work adopts the PFN approach for time-series forecasting. They either repurpose a pretrained non-longitudinal tabular model by treating time as additional features (Hoo et al., 2026) or explicit model temporal structure (Dooley et al., 2023; Taga et al., 2025; Potapczynski et al., 2026). Our model follows the latter approach, building an architecture to respect the temporal ordering of trajectories. A separate line of work applies PFN to causal inference (Balazadeh et al., 2025; Robertson et al., 2026; Ma et al., 2026). Yet they only focus on the static, single-treatment-assignment setting. In contrast, we consider the longitudinal setting with sequential treatment assignments and time-varying confounders[1].

Longitudinal Causal Effect Estimation. Longitudinal causal estimators address time-varying confounding through modeling treatment, outcome, or both. Treatment-based methods use inverse probability weighting or balanced representations (Robins, 2000; Bica et al., 2020; Seedat et al., 2022; Lim, 2018; Melnychuk et al., 2022), but can suffer from unstable weights under limited overlap (Cole & Hernán, 2008; Petersen et al., 2012). Outcome-based methods use the G-formula, either by simulating covariate trajectories (Li et al., 2021) or recursively regressing outcomes backward in time (McGrath et al., 2020; Frauen et al., 2023; Oprescu et al., 2025; Hess et al., 2026). Doubly robust methods combine treatment and outcome models to reduce sensitivity to misspecification (Van der Laan et al., 2011; Frauen et al., 2025; Shirakawa et al., 2024; Chen et al., 2026a;b).

Despite these advances, existing methods show limited flexibility to handle various causal queries. Many focus on prespecified treatment sequences that do not adapt to patients' evolving status. Some

[1]Concurrent work by Zare et al. (2026) approach this longitudinal setting using a one-step mean rollout that does not generally recover CAPO, and only supports static treatment sequences; see Appendix E.

require retraining or potentially retuning when changing the target policy or horizon. Recent DR-based methods also commonly assume a single-dimension treatment, with extensions to multidimensional treatment spaces requiring nontrivial changes to their models and objectives. In contrast, PICLE-PFN provides all these capabilities within a single pretrained model (Table 1).

3 Problem Setup and In-Context Formulation

We consider n independent and identically distributed (i.i.d.) trajectories observed over discrete time steps $t \in \{1, \dots, T\}$. For each trajectory, we observe time-invariant baseline covariates $B \in \mathbb{R}^{d_B}$, time-varying covariates $L_t \in \mathbb{R}^{d_L}$, and a d_A-dimensional binary treatments $A_t \in \{0,1\}^{d_A}$. Let $Y_{t+1} \in \mathbb{R}$ denote the outcome measured following treatment assignment A_t. An observed trajectory O is thus represented as: $O = (B, L_1, A_1, Y_2, L_2, A_2, Y_3, \dots, L_T, A_T, Y_{T+1}) \sim P_0$, where P_0 is the unknown true data-generating distribution.

For any sequence of random variables X_t, let $X_{s:e} = (X_s, X_{s+1}, \dots, X_e)$ denote the subsequence from time s to e, with $X_{s:e}$ empty when $s > e$. The observed history immediately preceding treatment assignment at time t is denoted as $H_t = (B, L_{1:t}, A_{1:t-1}, Y_{2:t})$, and the full trajectory up to time t as $O_t = (B, L_{1:t}, A_{1:t}, Y_{2:t+1})$.

Under P_0, treatments follow an unknown behavior policy $\pi^* = (\pi_1^*, \dots, \pi_T^*)$, where $\pi_t^*(a \mid h_t) := \Pr(A_t = a \mid H_t = h_t)$ is the probability of assigning treatment a given history h_t.

Target Policies and Target Estimand. We consider deterministic target policies $\pi_{s:e} = (\pi_s, \dots, \pi_e)$, where $\pi_t : \mathcal{H}_t \to \{0,1\}^{d_A}$. This includes both static treatment sequences and history-dependent dynamic policies. Denote $a_{s:e}^{\pi_{s:e}}$ as the treatment sequences that would have been assigned if we apply target policy $\pi_{s:e}$ starting from H_s, and denote $L_{e+1}^{\pi_{s:e}}$ and $Y_{e+1}^{\pi_{s:e}}$ as the potential covariates and outcome at time $e+1$, respectively, under $\pi_{s:e}$.

Given a history $h_s \in \mathcal{H}_s$, our target estimand is **conditional average potential outcome (CAPO)**,

$$\psi_0^{\pi_{s:e}}(h_s) := \mathbb{E}\left[Y_{e+1}^{\pi_{s:e}} \mid H_s = h_s\right].$$

Intuitively, $\psi_0^{\pi_{s:e}}(h_s)$ represents the true expected future outcome at time $e+1$ conditional on observed history h_s and treatment policy $\pi_{s:e}$ over the horizon s to e.

Identification Assumptions. We adopt the following standard assumptions (Shirakawa et al., 2024; Wen et al., 2021):

(a) Consistency. Observed covariates/outcome under realized treatments equal the potential covariates/outcome under that same treatments: $A_{s:e} = a_{s:e}^{\pi_{s:e}} \implies L_{e+1}^{\pi_{s:e}} = L_{e+1}, Y_{e+1}^{\pi_{s:e}} = Y_{e+1}$.
(b) Sequential Ignorability. For every $t \in \{s, \dots, e\}$ and target policy sequence $\pi_{t:e}$ under consideration, $(Y_{t+1}^{\pi_{t:t}}, \dots, Y_{e+1}^{\pi_{t:e}}) \perp A_t \mid H_t$.
(c) Positivity. For histories $h_t \in \mathcal{H}_t$ and $a = \pi_t(h_t)$: $\pi_t^*(a \mid h_t) > 0$.

With these assumptions, a target policy's CAPO is identified. We next show that a policy can be represented by its decisions on observed histories without losing information needed for identification.

3.1 In-context Formulation of Longitudinal Causal Effect Estimation

Target policies can take various forms, ranging from static treatment sequences to complex, black-box decision rules. Parameterizing their underlying mechanisms as model inputs would necessitate a distinct representation for each policy family. To circumvent this, we represent a policy through its decisions on the dataset's factual histories rather than its underlying rules or parameters. We refer to the set of history-decision pairs as a **target policy demonstration**, which serves as a universal interface for any *deterministic policy*. We first show that, under the standard assumptions, the target policy demonstrations suffice to identify the CAPO (proof in Appendix A).

Proposition 3.1 (Sufficiency of Factual History Policy Demonstrations)**.** *Suppose consistency, sequential ignorability, and positivity hold. For every $t \in \{s, \dots, e\}$, construct a population demonstration of a deterministic policy according to*

$$H_t \sim P_0, \qquad A_t^{\pi_t} \sim \pi_t(H_t).$$

Then the collection of population policy demonstration distributions of $(H_t, A_t^{\pi_t})$*, together with* P_0*, identifies the corresponding target CAPO,* $\psi_0^{\pi_{s:e}}(h_s)$*, for* P_0*-almost every* h_s*.*

Proposition 3.1 establishes sufficiency at the population level. In practice, we approximate the population demonstrations with the empirical dataset by applying the policy to the observed histories:

$$D_n^{\pi_{s:e}} = \left\{(H_{i,t}, A_{i,t}^{\pi_t})\right\}_{i=1:n,\ t=s:e} \quad \text{where } A_{i,t}^{\pi_t} \sim \pi_t(H_{i,t}).$$

We note that the policy information resides in the conditional relationship between histories and treatment decisions *across the entire observed dataset*. Any single history-treatment pair generally does not specify this relationship. An estimator must therefore aggregate information across demonstrations to estimate the CAPO. This naturally motivates an ICL formulation: the observed dataset plus policy demonstrations serves as context for inferring the relevant policy behavior.

Besides the policy demonstrations, an estimator also needs to learn the data distribution. Under the ICL formulation, we can provide the observed trajectories $D_{n,e}^{\text{obs}} = \{O_{i,e}\}_{i=1}^n$ as additional context. Because each observed trajectory $O_{i,e}$ already contains the patient's factual history, the corresponding policy demonstration for that patient is the sequence of target actions $(A_{i,t}^{\pi_t})_{t=s}^e$, prescribed for that specific history. This allows us to combine the two contexts efficiently without duplicating the history data, yielding the joint input: $\left\{\left(O_{i,e}, (A_{i,t}^{\pi_t})_{t=s}^e\right)\right\}_{i=1}^n$. We feed this combined context, along with a new query history h_s, into an amortized estimator $\mathcal{F}_\theta$:

$$\hat{\psi}_n^{\pi_{s:e}}(h_s) := \mathcal{F}_\theta\left(D_{n,e}^{\text{obs}}, D_n^{\pi_{s:e}}, h_s\right) = \mathcal{F}_\theta\left(\{(O_{i,e}, (A_{i,t}^{\pi_t})_{t=s}^e)\}_{i=1}^n, h_s\right),$$

where $\mathcal{F}_\theta$ is a shared parameterized predictor trained to estimate $\psi_0^{\pi_{s:e}}(h_s)$ across datasets and policies. Once trained, it estimates CAPO for new datasets, target policies, and query histories through ICL, without dataset- or policy-specific parameter updates.

Supervision for Multi-step Counterfactual Targets. The formulation above specifies the information available at inference time, but not how to supervise the estimator. Although the target CAPO is identified, it is not directly observed as a multi-step training label. For one-step outcome regression, the observed outcome provides a direct supervision signal: regressing Y_{t+1} on (H_t, A_t) estimates the conditional mean needed to evaluate a target policy. Over multiple steps, however, the observed endpoint reflects the subsequent treatments assigned under the behavior policy, rather than the target policy. It therefore does not generally provide a valid regression label for the multi-step CAPO under a target policy. Conventional G-computation methods bridge this gap by fitting local nuisance models and composing their predictions across time (Li et al., 2021; Shirakawa et al., 2024).

Our ICL estimator bypasses explicit nuisance-model fitting and directly maps observational and policy contexts to CAPO, requiring a different source of supervision. We therefore pretrain on synthetic datasets, from which we can generate both the observed data and corresponding CAPO targets through simulation under the target policy. The combination of ICL and synthetic supervision makes the PFN paradigm a natural fit.

4 Method

We introduce PICLE-PFN, a PFN-style model pretrained on tasks sampled from a prior over longitudinal structural causal models and target policies. The prior (§ 4.1) supplies diverse datasets, causal queries, and counterfactual labels; the training objective (§ 4.2) enables direct CAPO estimation without fully generative modeling (summarized in Algorithm 1). The architecture (§ 4.3) integrates observational data and policy demonstrations to jointly predict across intervention windows.

4.1 Prior Sampling

We generate synthetic training tasks by sampling longitudinal SCMs and target policies. Each SCM varies in its dimensions, graph skeleton, and transition mechanisms, capturing nonlinear dynamics, heterogeneous treatment effects, and diverse behavioural policies. We construct the graph by sampling parent variables and their time lags while enforcing acyclicity. Target policies include both static treatment sequences and dynamic rules. For each task, we generate factual trajectories and counterfactual rollouts under the target policy; the latter provide supervision labels whose conditional mean is the CAPO. Appendix B.2 details the sampling procedure.

Algorithm 1 Longitudinal Causal PFN Pre-Training

Input: Prior over longitudinal SCMs Π and target policies Π_{policy}
Sample a longitudinal SCM $\mathcal{M} \sim \Pi$, which includes the behavior policy $\boldsymbol{\pi}^*$
Sample a target counterfactual policy: $\boldsymbol{\pi} \sim \Pi_{\text{policy}}$.
for $i = 1, \ldots, n$ **do**
  Generate an observed trajectory, O_i, of length T under the sampled SCM and behavior policy.
  for $(s, e) \in \mathcal{K}$ **do**
    Obtain target policy demonstration by evaluating π_t on the observed histories at each time step: $A_{i,t}^{\pi_t} = \pi_t(H_{i,t}),$ for $t = s, \ldots, e$.
    Simulate true counterfactual outcome: Branch a counterfactual rollout from $H_{i,s}$ by applying $\pi^{s:e}$ on the evolving counterfactual history, which yields the label $Y_{i,e+1}^{\pi_{s:e}}$.
  end for
end for
Partition the trajectory indices into disjoint context and query sets
Minimize the negative log-likelihood over intervention windows and query trajectories. (Eq.1)

4.2 Training Objective

We train PICLE-PFN to estimate CAPOs through the predictive means of counterfactual outcome distributions, jointly across query trajectories and intervention windows. For start time s and end time e, let $\tau = e - s + 1$ denote the horizon. The set of admissible windows is

$$\mathcal{K} = \{(s, e) : 1 \le s \le e \le T,\ e - s + 1 \le \tau_{\max}\}.$$

Let $[n] = \{1, \ldots, n\}$ index the trajectories in a training task. We partition these indices into disjoint context and query sets $\mathcal{I}_{\text{ctx}}$ and $\mathcal{I}_{\text{qry}}$, with $\mathcal{I}_{\text{ctx}} \cup \mathcal{I}_{\text{qry}} = [n]$, $\mathcal{I}_{\text{ctx}} \cap \mathcal{I}_{\text{qry}} = \emptyset$. For each $(s, e) \in \mathcal{K}$, the inputs are

$$D_{s:e}^{\text{ctx}} = \left\{ \left(O_{i,e}, \{A_{i,t}^{\pi_t}\}_{t=s}^{e} \right) : i \in \mathcal{I}_{\text{ctx}} \right\}, \qquad D_{s:e}^{\text{qry}} = \left\{ (H_{i,s}, A_{i,s}^{\pi_s}) : i \in \mathcal{I}_{\text{qry}} \right\}.$$

The context provides factual trajectories through the endpoint and policy demonstrations throughout the intervention window. Each query provides only its pre-intervention history and initial policy demonstration; information about subsequent policies is supplied by the context demonstrations.

For each query and window, we branch a counterfactual rollout from $H_{i,s}$ and apply the target policy through step e, obtaining a realized potential outcome $Y_{i,e+1}^{\pi_{s:e}}$. For a fixed SCM and target policy, this outcome has conditional expectation equal to the CAPO given $H_{i,s}$. We therefore use single-draw outcomes as a computationally efficient surrogate for direct CAPO supervision, avoiding the repeated rollouts required to approximate conditional means explicitly. We minimize the outcome negative log-likelihood:

$$\mathcal{L}(\theta) = \mathbb{E}\left[-\frac{1}{|\mathcal{I}_{\text{qry}}|\,|\mathcal{K}|} \sum_{i \in \mathcal{I}_{\text{qry}}} \sum_{(s,e) \in \mathcal{K}} \log q_\theta\left(Y_{i,e+1}^{\pi_{s:e}} \mid H_{i,s}, A_{i,s}^{\pi_s}, D_{s:e}^{\text{ctx}} \right) \right], \tag{1}$$

where the expectation is over sampled tasks. At inference, the predictive mean provides the CAPO estimate. Because the model is trained on realized counterfactual outcomes, its predictive spread also includes outcome variability.

4.3 Model Architecture

We parametrize $\mathcal{F}_\theta$ using a PFN-style transformer (Hollmann et al., 2025; QU et al., 2025) with a shared *observed-data backbone* and *window-specific readout tokens*. Each trajectory has one readout token per intervention window (Figure 1). Each transformer layer first updates the observed-data representations through causal attention across time steps and row attention across trajectories, then updates the readout tokens. Each readout cross-attends to its own trajectory under a window-specific mask: for window (s, e), query readouts access only the factual history H_s, while context readouts access the factual trajectory through the target endpoint, including Y_{e+1}. The context readouts additionally incorporate target-policy demonstrations over $s{:}e$. Readout tokens then attend across trajectories, using only context readouts as keys and values, to enable in-context learning. Finally, the query readouts predict CAPO. Readout tokens share model parameters across windows, producing predictions for all admissible windows in a single forward pass. Full details appear in Appendix B.1.

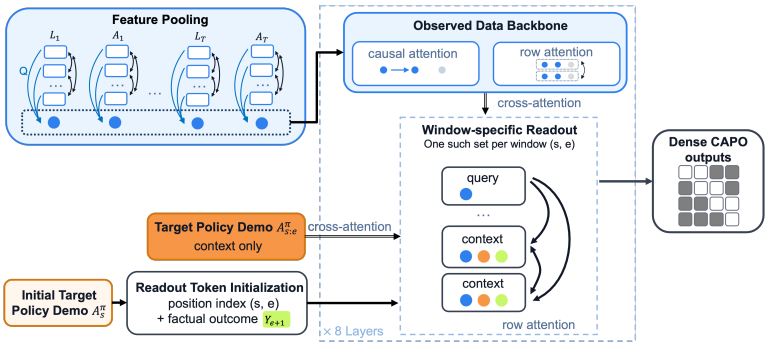


Figure 1: **PICLE-PFN Architecture**. Feature pooling encodes longitudinal covariates and treatments into embeddings processed by a shared backbone with causal temporal attention and row-attention. Window-specific readouts combine window indices, initial policy demonstrations, and factual outcomes (masked for queries). They cross-attend to trajectory-specific backbone representations under window-specific masks: query readouts access only pre-intervention history, while context readouts access factual data through e and additionally attend to target policy demonstrations. Row-attention uses only context keys and values and operates separately within each readout window. Query readouts yield CAPO predictions across admissible windows in one forward pass.

5 Experiments

We first describe the datasets, baselines, and evaluation protocol (§ 5.1). We then report CAPO estimation performance, a sanity check on real-world data prediction, and additional analyses on uncertainty calibration and ablation studies (§ 5.2).

5.1 Experimental Setup

Datasets & Metrics. We use seven dataset settings, where the range of settings spans varying dimensions of treatments (d_A) and covariates (d_L, d_B), as well as static and dynamic treatment policies (Table 2). In the three `mimic_1t_hdp*` settings, their target policy functions have the same functional form as the behavioral policy, but have a scalar perturbation term that controls the deviation from the behavioral policy. The detailed data-generating process can be found in Appendix D.1. Each setting synthesizes 20 datasets with 1000 trajectories in the training (context) set, and 1000 in the test (query) set. We estimate the test set CAPO on five different horizons, $\tau \in \{2, 4, 6, 8, 10\}$. We report RMSE, normalized by the standard deviation of observed outcomes. We additionally evaluate PICLE-PFN on a real-world prediction task as a sanity check, described in § 5.2 and Appendix D.5.

Table 2: Dataset characteristics.

Dataset	(d_A, d_L, d_B)	Policy type
`cancer`	$(2, 1, 1)$	Static
`mimic_y1`	$(3, 27, 3)$	Static
`mimic_y2`	$(3, 27, 3)$	Static
`mimic_1t`	$(1, 27, 3)$	Static
`mimic_1t_hdp0.5`	$(1, 27, 3)$	Dynamic
`mimic_1t_hdp1`	$(1, 27, 3)$	Dynamic
`mimic_1t_hdp2`	$(1, 27, 3)$	Dynamic

Larger HDP values indicate greater deviation from the behavioral policy.

Baselines. We compare against nine baselines. Given their flexibility to handle different settings, not all of them are evaluated in all baselines: (1) **DeepACE** (Frauen et al., 2023), **DeepLTMLE** (Shirakawa et al., 2024), and **D3-Net** (Chen et al., 2026a) can only handle single-dimensional treatment, so they are excluded from `cancer`, `mimic_y1`, and `mimic_y2`; (2) **CRN** (Bica et al., 2020), **RMSN** (Lim, 2018), **TE-CDE** (Seedat et al., 2022), **CT** (Melnychuk et al., 2022), and **G-Net** (Li et al., 2021) don't handle dynamic treatment policies, so they are excluded from `mimic_hdp0.5/1/2`; (3) **IGC-Net** is evaluated on all settings. For every baseline, we tune its hyperparameters (HPs) with Optuna for 2 hours.The HP search spaces are listed in Table S.4. Further details can be found in Appendix D.3.

5.2 Results

Synthetic/Semi-synthetic Data Table 3 shows that PICLE-PFN achieves the lowest RMSE in six of seven settings, reducing RMSE over the best baseline by 48.7% on `cancer` and by up to 15.5% on the remaining MIMIC benchmarks; on `mimic_y1`, it remains comparable, with 3.3% higher

Table 3: **Longitudinal CAPO benchmark across datasets.** Each cell reports the mean±SD normalized RMSE over 20 runs, with RMSE averaged across horizons within each repeat. **Bold** and underlined values denote the best and second-best methods in each column, respectively.

Method	Cancer	MIMIC					
		Y1	Y2	1T	1T-HDP0.5	1T-HDP1	1T-HDP2
PICLE-PFN	**0.19±0.03**	0.99±0.66	**0.65±0.27**	**0.60±0.20**	**0.53±0.13**	**0.55±0.14**	**0.62±0.17**
IGC-Net	0.37±0.04	**0.96±0.60**	0.68±0.17	0.65±0.12	0.63±0.12	0.65±0.15	0.69±0.16
CT	0.41±0.06	1.65±0.56	0.98±0.74	0.73±0.48	—	—	—
RMSN	0.60±0.09	1.49±0.67	1.01±0.22	0.83±0.14	—	—	—
G-Net	0.75±0.04	1.50±0.67	1.23±0.14	1.13±0.16	—	—	—
CRN	0.80±0.06	1.59±0.62	1.27±0.43	0.95±0.54	—	—	—
TE-CDE	1.04±0.05	1.74±0.64	1.48±0.43	1.26±0.11	—	—	—
D3	—	—	—	1.16±0.11	0.97±0.06	0.97±0.04	1.00±0.14
DLTMLE	—	—	—	1.25±0.32	0.97±0.04	1.00±0.04	1.11±0.14
DeepACE	—	—	—	0.76±0.37	—	—	—
Δ vs. best baseline	-48.7%	+3.3%	-5.1%	-8.1%	-15.5%	-14.7%	-10.3%

"—" indicates that a method does not support the corresponding setting.

RMSE. Figure 2 selects the top three models in Table 3 and visualizes their RMSE results for every horizon in each data setting, showing that its advantage holds across horizons (Full results across horizons can be found in Table S.2).

Table S.3 shows measurements of deployment runtime on three representative settings: PICLE-PFN is about $50\times$ faster than the fastest baseline, and $5000\times$ faster than IGC-Net, the only baseline with comparable accuracy. Figure 3 further compares the RMSE ranking and deployment runtime; results are aggregated across horizons (Appendix D.4). Together, these results show that PICLE-PFN combines top-ranking accuracy across diverse settings with substantially faster runtime.

Real-world Data. We further evaluate prediction under the behavioral policy on real-world MIMIC data, using vasopressor administration and mechanical ventilation as treatments and multi-step-ahead diastolic blood pressure as the outcome. To avoid introducing a separately fitted behavioral model, we use factual treatments in IGC-Net's recursive regression targets and as policy demonstrations for PICLE-PFN; implementation details appear in Appendix D.5. IGC-Net is the only baseline that accommodates both multivariate treatments and dynamic policies as in this setting, so we restrict the comparison accordingly. Table 4 shows that PICLE-PFN achieves slightly lower prediction error across all horizons, suggesting that it retains strong predictive performance on factual outcomes in real-world data. We view this experiment as a real-world sanity check, complementary to the synthetic and semi-synthetic benchmarks where counterfactual ground truth is available.

Table 4: **Factual Outcome Prediction on MIMIC.** Normalized RMSE of the factual diastolic blood pressure on the held-out test patients of 10 sampled cohorts.

Method	$\tau = 2$	$\tau = 3$	$\tau = 4$	$\tau = 5$
PICLE-PFN	**0.730±0.030**	**0.737±0.032**	**0.754±0.026**	**0.767±0.037**
IGC-Net	0.736±0.040	0.740±0.038	0.762±0.031	0.779±0.028
PICLE-PFN wins	7/10	5/10	8/10	8/10

Uncertainty Calibration. We compare the empirical coverage of CAPO by the intervals produced by PICLE-PFN and the strongest baseline, IGC-Net. PICLE-PFN outputs a predictive distribution in a single forward pass, from which we extract equal-tailed $100(1-\alpha)\%$ intervals.[2] For IGC-Net, we construct intervals using MC dropout, following its original uncertainty-quantification approach. A query trajectory is covered if its true CAPO lies inside the interval; task-level coverage is the fraction of covered queries. We plot empirical CAPO coverage against the nominal interval level $1-\alpha$, using the diagonal as a reference. Evaluation details are in Appendix D.6.

Figure 4 (left) compares uncertainty calibration on `mimic_y1` and `mimic_1t` through empirical CAPO coverage, averaged across horizons. PICLE-PFN tracks the nominal coverage levels substantially more closely than IGC-Net, whose MC-dropout uncertainty estimates are severely overconfident. Across horizons, PICLE-PFN maintains coverage near 0.9 at the nominal 95% level (Figure 4, middle), while its predictive uncertainty increases, as reflected by wider intervals (Figure 4, right). This is the desired behavior: the model widens its intervals to reflect the growing uncertainty of

[2]The single-draw supervision during pretraining introduces outcome variability, so these should not be interpreted as credible intervals for CAPO.

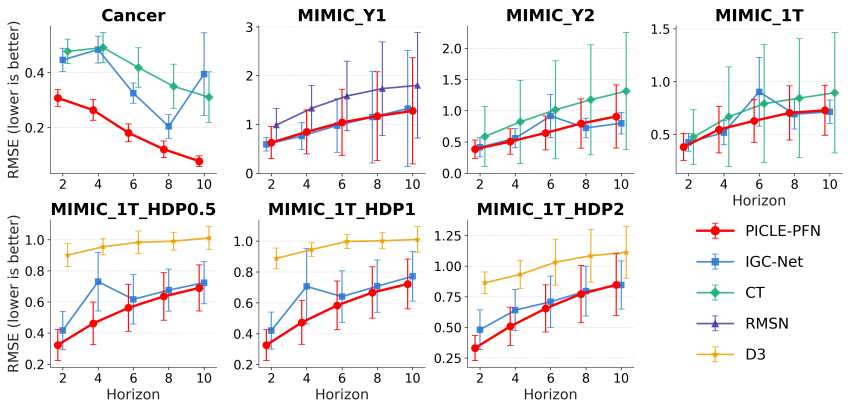


Figure 2: RMSE across horizons of the top-3 methods in each setting. Points are the normalized RMSE averaged over the 20 tasks, error bars are ± 1 SD across tasks

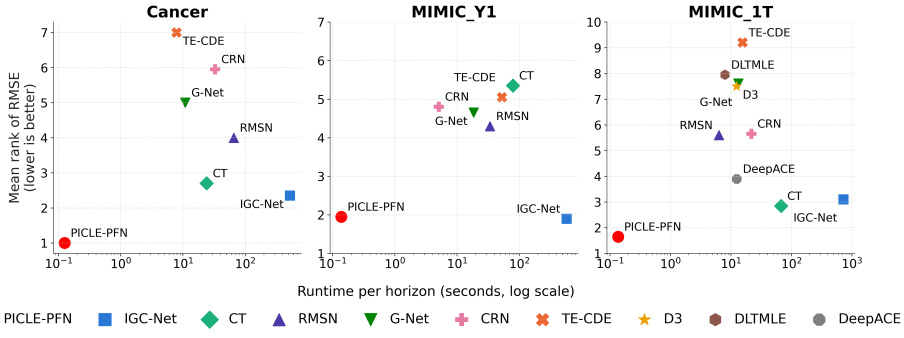


Figure 3: RMSE rank (y-axis) versus inference runtime (x-axis; log scale) across datasets.

longer horizons, and thereby maintains coverage. In contrast, IGC-Net's intervals do not widen with longer horizons, and even narrow in one setting, suggesting its substantial overconfidence across horizons. These results show that PICLE-PFN expresses predictive uncertainty more consistently with its observed CAPO estimation errors.

Unobserved Confounding. We examine how PICLE-PFN and the two strongest baselines, CT and IGC-Net under unobserved confounder. On `mimic_1t`, we draw a confounder $U \sim \mathcal{N}(0, 1)$ that is hidden from all methods and enters the data-generating process twice by shifting the patient's outcome and propensity at every step. We vary the strength of unobserved confounding (details in Appendix D.7). Figure 5(a) shows that PICLE-PFN and CT remain relatively stable across the tested range, whereas IGC-Net's performance deteriorates as confounding becomes stronger.

Varying Context Sample Size. Smaller samples increase finite-sample error and, for our method, provide fewer demonstrations of the target policy. We test robustness under smaller context size, varying the context size on `mimic_1t` (static) and `mimic_1t_hdp1` (dynamic policy). Baselines are re-tuned and re-trained at every size. Figure 5(b) and Table S.5 show that RMSE increases for all methods as sample decreases, but less so for PICLE-PFN than for the baselines.

Target Policies Outside the Prior. Our prior covers both the DGP and the target policy, which raises a natural question: how does PICLE-PFN perform when the deployment policy is not represented in the policy prior? The policy prior contains history-dependent (dynamic) policies, sampled from neural networks and trees, and three types of static policies: always treat, never treat, and treat for the first few steps only. None of these produces an *alternating* treatment sequence. On `mimic_1t` with $\tau = 4$, we evaluate two unseen policies, applied regardless of how the patient responds: treat at every other step, $(1, 0, 1, 0)$, and treat at the first and last steps only, $(1, 0, 0, 1)$. The observed data are identical to those of the original evaluation. Figure 5(c) shows that PICLE-PFN remains robust: its RMSE increases only moderately, from 0.54 under always-treat to 0.62 and 0.59 on the two unseen policies, respectively. PICLE-PFN shows the smallest degradation among the

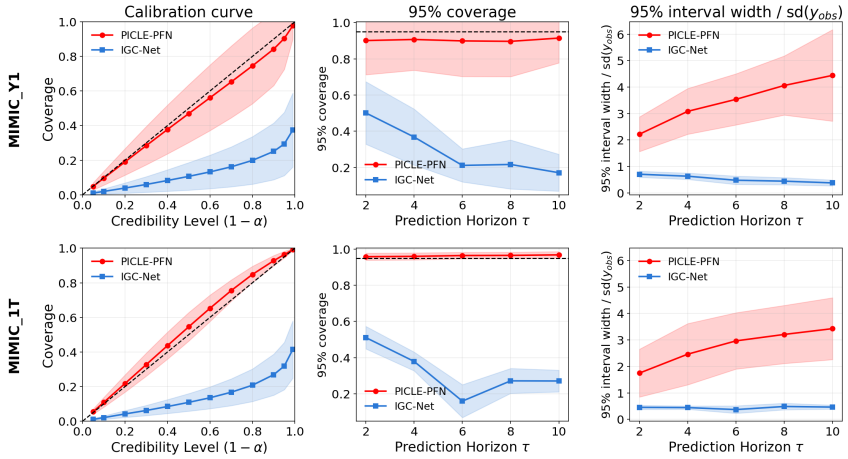


Figure 4: **Uncertainty Calibration Analysis.** Left: empirical coverage against its nominal level, averaged across horizons and 20 tasks; dashed diagonal = perfect calibration. Middle: coverage of the 95% interval per horizon (dashed line = nominal). Right: 95% interval width per horizon.

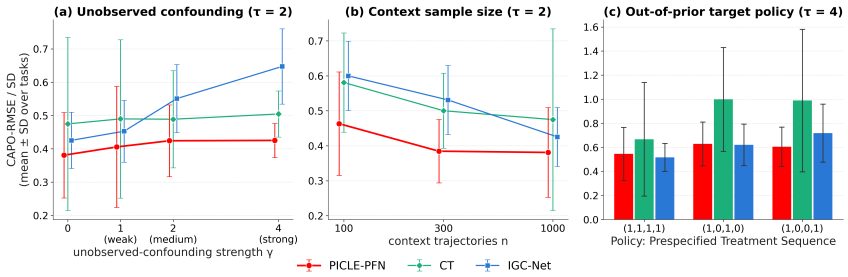


Figure 5: **Ablation studies.** (a) Unobserved confounding. (b) Context sample size. (c) Our-of-prior target policy: Static alternating treatments. Compared with the in-prior always-treat policy.

three methods. The result suggests that in-context conditioning on the target policy can generalize beyond the policy families represented in the prior. The baselines show larger degradation despite being retrained for each target policy. One possible reason is that the new policies are harder to estimate from the data, as they have lower overlap than the always-treat policy.

6 Conclusion, Limitations, and Future Work

We reformulate longitudinal CAPO estimation as an ICL task by representing target policies through decisions on factual histories across the observed dataset. We introduce PICLE-PFN, the first foundation model to longitudinal CAPO estimation supporting both static and dynamic policies. On synthetic and semi-synthetic benchmarks, PICLE-PFN matches or outperforms baselines at a fraction of their deployment runtime. We further evaluate its ability to predict factual outcomes in real-world data. Further analyses demonstrate strong uncertainty calibration, greater resilience to small samples and unobserved confounders, and generalization to policies unseen during pretraining.

Several limitations motivate future work. (i) *Computational complexity.* Quadratic costs in sample size N and trajectory length T (Appendix F) limit scalability. Advances in scalable tabular foundation models (QU et al., 2026; Grinsztajn et al., 2026) may inform more efficient architectures. (ii) *Scope of pretraining.* Our model considers binary treatments and discrete-time trajectories without right censoring. Extending it to broader treatment spaces, continuous-time series, and censored outcomes would expand its applicability. (iii) *Identifying assumptions and prior dependence.* Causal identification requires assumptions such as sequential ignorability; empirical robustness to certain violations does not establish theoretical guarantees under unobserved confounding. Accuracy may also deteriorate when deployment settings differ from the pretraining distribution. Characterizing this sensitivity and developing guarantees under prior misspecification remain important directions.

AI USE STATEMENT

In this work, we used generative AI tools to formulate mathematical claims, assist in the writing of proofs, provide feedback on experiments, implement methods (detailed implementation of model architecture), clean and reformat dataset (adapting IGC-net's datasets), and support qualitative and thematic data analysis.

We have not used generative AI tools to design research methodology, help develop theoretical models or conceptual frameworks, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, interpret results. Assisting with translation, and generating synthetic data sets, are not applicable to this work.

Additionally, we used generative AI tools for create or modify scientific figures or images, suggest experimental parameters, edit a research paper to improve readability, draft parts of a research paper(mostly cumbersome sections in Appendix, e.g. table of hyperparmeter search space), identify relevant literature, propose a title for a research paper.

We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness. We checked texts drafted or polished by LLM and ensured that they did not distort our research idea and our interpretation. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Vahid Balazadeh, Hamidreza Kamkari, Valentin Thomas, Junwei Ma, Bingru Li, Jesse C. Cresswell, and Rahul Krishnan. CausalPFN: Amortized causal effect estimation via in-context learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=RblaNJGx8C.

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

Ioana Bica, Ahmed M Alaa, James Jordon, and Mihaela Van Der Schaar. Estimating counterfactual treatment outcomes over time through adversarially balanced representations. *arXiv preprint arXiv:2002.04083*, 2020.

Bibhas Chakraborty and Erica EM Moodie. *Statistical methods for dynamic treatment regimes*, volume 2. Springer, 2013.

Wenxin Chen, Weishen Pan, Kyra Gan, and Fei Wang. Deep doubly debiased longitudinal effect estimation with ice g-computation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 10646–10657, 2026a.

Wenxin Chen, Weishen Pan, Kyra Gan, and Fei Wang. Smooth multi-policy causal effect estimation in longitudinal settings. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=bIcz7bIZSo.

Stephen R Cole and Miguel A Hernán. Constructing inverse probability weights for marginal structural models. *American journal of epidemiology*, 168(6):656–664, 2008.

Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha V Naidu, and Colin White. Forecastpfn: Synthetically-trained zero-shot forecasting. *Advances in Neural Information Processing Systems*, 36:2403–2426, 2023.

Dennis Frauen, Tobias Hatt, Valentyn Melnychuk, and Stefan Feuerriegel. Estimating average causal effects from patient trajectories. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7586–7594, 2023.

Dennis Frauen, Konstantin Hess, and Stefan Feuerriegel. Model-agnostic meta-learners for estimating heterogeneous treatment effects over time. In *International Conference on Learning Representations*, volume 2025, pp. 85750–85779, 2025.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, et al. Tabpfn-3: Technical report. *arXiv preprint arXiv:2605.13986*, 2026.

Miguel A Hernán and James M Robins. Causal inference, 2010.

Konstantin Hess, Dennis Frauen, Valentyn Melnychuk, and Stefan Feuerriegel. Igc-net for conditional average potential outcome estimation over time. In *International Conference on Learning Representations*, volume 2026, pp. 74263–74298, 2026.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending tabPFN-v2 to time series forecasting. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=KIkQj8VOUY.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Rui Li, Stephanie Hu, Mingyu Lu, Yuria Utsumi, Prithwish Chakraborty, Daby M Sow, Piyush Madan, Jun Li, Mohamed Ghalwash, Zach Shahn, et al. G-net: a recurrent network approach to g-computation for counterfactual prediction under a dynamic treatment regime. In *Machine Learning for Health*, pp. 282–299. PMLR, 2021.

Bryan Lim. Forecasting treatment responses over time using recurrent marginal structural networks. *Advances in neural information processing systems*, 31, 2018.

Yuchen Ma, Dennis Frauen, Emil Javurek, and Stefan Feuerriegel. Foundation models for causal inference via prior-data fitted networks. In *International Conference on Learning Representations*, volume 2026, pp. 79065–79098, 2026.

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

Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. Causal transformer for estimating counterfactual outcomes. In *International conference on machine learning*, pp. 15293–15329. PMLR, 2022.

Susan A Murphy. Optimal dynamic treatment regimes. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 65(2):331–355, 2003.

Miruna Oprescu, David Keetae Park, Xihaier Luo, Shinjae Yoo, and Nathan Kallus. Gst-unet: A neural framework for spatiotemporal causal inference with time-varying confounding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Maya L Petersen, Kristin E Porter, Susan Gruber, Yue Wang, and Mark J Van Der Laan. Diagnosing and responding to violations in the positivity assumption. *Statistical methods in medical research*, 21(1):31–54, 2012.

Andres Potapczynski, Ravi Kiran Selvam, Tatiana Konstantinova, Malcolm Wolff, Kin G. Olivares, Ruijun Ma, Michael W. Mahoney, Andrew Gordon Wilson, Boris N. Oreshkin, and Dmitry Efimov. Time-aware prior fitted networks for zero-shot forecasting with exogenous variables. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=nJARpxp3cF.

Jingang QU, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=0VvD1PmNzM.

Jingang QU, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=SxsyLjIfWB.

Jake Robertson, Arik Reuter, Siyuan Guo, Noah Hollmann, Frank Hutter, and Bernhard Schölkopf. Do-pfn: In-context learning for causal effect estimation. *Advances in Neural Information Processing Systems*, 38:174811–174848, 2026.

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.

James M Robins. Marginal structural models versus structural nested models as tools for causal inference. In *Statistical models in epidemiology, the environment, and clinical trials*, pp. 95–133. Springer, 2000.

Nabeel Seedat, Fergus Imrie, Alexis Bellot, Zhaozhi Qian, and Mihaela van der Schaar. Continuous-time modeling of counterfactual outcomes using neural controlled differential equations. *arXiv preprint arXiv:2206.08311*, 2022.

Toru Shirakawa, Yi Li, Yulun Wu, Sky Qiu, Yuxuan Li, Mingduo Zhao, Hiroyasu Iso, and Mark Van Der Laan. Longitudinal targeted minimum loss-based estimation with temporal-difference heterogeneous transformer. *Proceedings of machine learning research*, 235:45097, 2024.

Ege Onur Taga, Muhammed Emrullah Ildiz, and Samet Oymak. Timepfn: Effective multivariate time series forecasting with synthetic data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 20761–20769, 2025.

Mark J Van der Laan, Sherri Rose, et al. *Targeted learning: causal inference for observational and experimental data*, volume 4. Springer, 2011.

Shirly Wang, Matthew B. A. McDermott, Geeticka Chauhan, Marzyeh Ghassemi, Michael C. Hughes, and Tristan Naumann. Mimic-extract: a data extraction, preprocessing, and representation pipeline for mimic-iii. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, ACM CHIL '20, pp. 222–235. ACM, April 2020. doi: 10.1145/3368555.3384469. URL http://dx.doi.org/10.1145/3368555.3384469.

Lan Wen, Jessica G Young, James M Robins, and Miguel A Hernán. Parametric g-formula implementations for causal survival analyses. *Biometrics*, 77(2):740–753, 2021.

Amirhossein Zare, Amirhessam Zare, Herlock Rahimi, Reza Salarikia, and Mohammad Kashkooli. Causal longitudinal prior-fitted networks for counterfactual outcome prediction. *arXiv preprint arXiv:2606.05797*, 2026.

APPENDIX

A PROOF

Proposition 3.1 (Sufficiency of Factual History Policy Demonstrations)**.** *Suppose consistency, sequential ignorability, and positivity hold. For every $t \in \{s, \dots, e\}$, construct a population demonstration of a deterministic policy according to*

$$H_t \sim P_0, \qquad A_t^{\pi_t} \sim \pi_t(H_t).$$

Then the collection of population policy demonstration distributions of $(H_t, A_t^{\pi_t})$, together with P_0, identifies the corresponding target CAPO, $\psi_0^{\pi_{s:e}}(h_s)$, for P_0-almost every h_s.

Proof. We first recover the deterministic policies from the population demonstrations, then show by backward induction that they uniquely determine the ICE recursion.

For notational convenience, write

$$\pi_t(a \mid h_t) := \mathbf{1}\{a = \pi_t(h_t)\}$$

for the degenerate treatment distribution induced by the deterministic rule π_t. Use the same convention for any other policy considered below.

For each $t \in \{s, \dots, e\}$, let P_0 denote the observed marginal distribution of H_t. By construction,

$$\Pr(A_t^{\pi_t} = a \mid H_t = h_t) = \mathbf{1}\{a = \pi_t(h_t)\} = \pi_t(a \mid h_t).$$

Thus, the joint distribution of $(H_t, A_t^{\pi_t})$ uniquely determines $\pi_t(h_t)$, and hence $\pi_t(\cdot \mid h_t)$, for P_0-almost every h_t.

To connect these policy conditionals to the target estimand, we use the iterative conditional expectation (ICE) representation of the longitudinal g-formula. Define

$$Q_e^\pi(h_e, a_e) = \mathbb{E}_{P_0}\left[Y_{e+1} \mid H_e = h_e, A_e = a_e\right],$$

and

$$V_e^\pi(h_e) = \sum_{a_e:\, \pi_e(a_e|h_e)>0} \pi_e(a_e \mid h_e) Q_e^\pi(h_e, a_e).$$

For $t = e-1, \dots, s$, recursively define

$$Q_t^\pi(h_t, a_t) = \mathbb{E}_{P_0}\left[V_{t+1}^\pi(H_{t+1}) \mid H_t = h_t, A_t = a_t\right],$$

and

$$V_t^\pi(h_t) = \sum_{a_t:\, \pi_t(a_t|h_t)>0} \pi_t(a_t \mid h_t) Q_t^\pi(h_t, a_t).$$

Under the stated longitudinal identification assumptions (Shirakawa et al., 2024; Wen et al., 2021; Hess et al., 2026), the g-formula gives

$$V_s^\pi(h_s) = \mathbb{E}\left[Y_{e+1}^{\pi_{s:e}} \mid H_s = h_s\right] = \psi^{\pi_{s:e}}(h_s)$$

for almost every h_s under P_0.

It therefore suffices to establish uniqueness of V_s^π. Consider two policy sequences π and $\widetilde{\pi}$ that generate the same population demonstration distributions. Let (Q_t^π, V_t^π) and $(Q_t^{\tilde{\pi}}, V_t^{\tilde{\pi}})$ denote their corresponding ICE recursions under the same P_0. Policy recovery implies

$$\pi_t(\cdot \mid h_t) = \widetilde{\pi}_t(\cdot \mid h_t) \qquad P_0\text{-a.e.}$$

We show by backward induction that $V_t^\pi = V_t^{\tilde{\pi}}$ almost everywhere under P_0.

For the base case $t = e$, both terminal regressions are conditional expectations of the same observed outcome under P_0. Hence,

$$Q_e^\pi(H_e, A_e) = Q_e^{\tilde{\pi}}(H_e, A_e) \qquad P_0\text{-a.s.}$$

By positivity,

$$\pi_e(a_e \mid h_e) > 0 \implies \pi_e^*(a_e \mid h_e) > 0.$$

Consequently, for almost every h_e under P_0, the terminal regressions agree on every action contributing to the policy-weighted sum. Together with $\pi_e = \widetilde{\pi}_e$ almost everywhere, this gives

$$V_e^{\pi}(h_e) = V_e^{\tilde{\pi}}(h_e) \qquad P_0\text{-a.e.}$$

For the induction step, suppose $V_{t+1}^{\pi} = V_{t+1}^{\tilde{\pi}}$ almost everywhere under P_0. We first show that any remaining disagreement on an observed-history null set cannot affect Q_t^{π}. Define

$$N_{t+1} = \{h_{t+1} : V_{t+1}^{\pi}(h_{t+1}) \neq V_{t+1}^{\tilde{\pi}}(h_{t+1})\}.$$

The induction hypothesis implies

$$\begin{aligned} P_0(H_{t+1} \in N_{t+1}) = 0 &\Longrightarrow P_0(H_{t+1} \in N_{t+1} \mid H_t, A_t) = 0 \quad P_0\text{-a.s.} \\ &\Longrightarrow Q_t^{\pi}(H_t, A_t) = Q_t^{\tilde{\pi}}(H_t, A_t) \quad P_0\text{-a.s.} \end{aligned}$$

The second implication follows because Q_t^{π} and $Q_t^{\tilde{\pi}}$ average V_{t+1}^{π} and $V_{t+1}^{\tilde{\pi}}$ under the same observed conditional distribution.

We then apply the same positivity argument as in the base case: the regressions agree on all actions contributing to the policy-weighted sum, for almost every observed history. Since $\pi_t = \widetilde{\pi}_t$ almost everywhere,

$$V_t^{\pi}(h_t) = V_t^{\tilde{\pi}}(h_t) \qquad P_0\text{-a.e.}$$

This completes the induction.

In particular,

$$V_s^{\pi}(h_s) = V_s^{\tilde{\pi}}(h_s) \qquad P_0\text{-a.e.}$$

Since the ICE identification result equates V_s^{π} with the target functional, P_0 and the population policy demonstration distributions uniquely determine $\Psi^{\pi_{s:e}}(P_0)(h_s)$ for almost every h_s under P_0. □

B Implementation Details

B.1 Model Architecture

The model operates on two kinds of tokens: *observed data tokens*, which represent the factual trajectories, and *readout tokens*, which combine the factual representations with target policy demonstrations to predict potential outcomes. Both are processed by the same stack of eight transformer layers. Each layer has two parts: an *observed data backbone* that updates the observed data tokens, followed by a *readout update* in which the readout tokens attend to the backbone's output at that layer. The backbone never attends to readout tokens, so its representations are independent of the target policy and intervention window. We first describe the architecture for a single intervention window (s, e), and then explain how the same construction is extended to all admissible windows in parallel.

Feature Pooling and Tokenization. Each trajectory is represented by $2T$ pooled embeddings corresponding to the interleaved sequence $L_1, A_1, \ldots, L_T, A_T$. To encode each L or A vector, we first embed its scalar features and add learned column-identity embeddings. A feature-attention layer models interactions among features using self-attention, followed by attention pooling with learned query tokens to aggregate them into a single embedding (Lee et al., 2019; QU et al., 2025; Grinsztajn et al., 2026). Baseline covariates B are encoded in the same way, and their pooled embedding is added to every L_t. Each sequence position also receives a position embedding over the $2T$ positions plus an L/A type embedding.

The readout tokens $R_{s:e}$ are initialized with three components: (i) Start- and end-index embeddings; (ii) The initial target policy demonstration $A_s^{\pi_s}$; (iii) Outcome: factual outcome Y_{e+1} for context trajectories, and a null value for query trajectories.

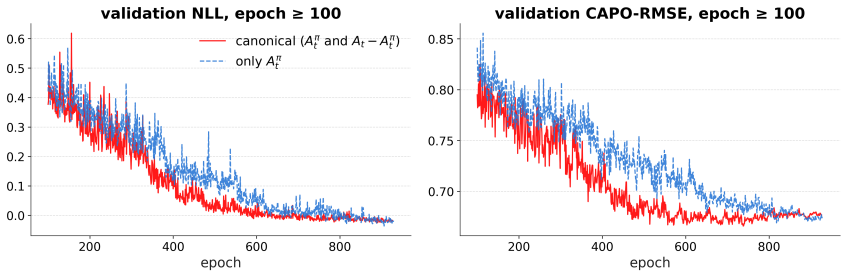


Figure S.1: Ablation study on the target policy demonstration channel. Supplying the differences between demonstrated and factual treatments accelerates convergence.

Observed data backbone. Within each transformer layer, the backbone updates the observed data tokens by (i) causal attention[3] along time, allowing L_t to attend to $(B, L_{1:t}, A_{1:t-1})$ and A_t to attend to $(B, L_{1:t}, A_{1:t})$; (ii) row attention across trajectories at each time position, with keys and values restricted to context trajectories. The output then goes through an MLP layer.

Window-specific readout. Within each transformer layer, after the backbone update, the readout token is updated in three steps. (i) $R_{s:e}$ cross-attends to the backbone representations of its own trajectory at that layer, under an asymmetric, window-specific mask. A query readout accesses only the tokens representing H_s, whereas a context readout accesses the factual sequence through the end of the window, (H_e, A_e); all later backbone tokens are masked. Context readouts additionally cross-attend to embeddings of the target policy demonstrations $\{A_t^{\pi_t}\}_{t=s}^{e}$. We empirically found that additionally supplying the differences between demonstrated and factual treatments, $(A_t^{\pi_t} - A_t)_{t=s}^{e}$ accelerates convergence without changing the information available to the model(Figure S.1). (ii) The readout tokens attend across trajectories, with keys and values restricted to context readouts. (iii) An MLP follows. The backbone and the readout update have separate parameters. After the last layer, a linear head maps each query readout to a bar distribution (Hollmann et al., 2025) over the standardized outcome.

Dense CAPO prediction across windows. The description above considers a single window (s, e). We instantiate one readout token per trajectory for every admissible window. These tokens share the same parameters and differ only in their initialization and window-specific access masks. Readouts of different windows do not interact: cross-trajectory attention is performed separately for each window, so a readout for (s, e) attends only to context readouts for that same (s, e).

B.2 Prior Sampling

We pretrain exclusively on synthetic tasks, each defined by a longitudinal SCM and a target policy. Each task contains 1024 trajectories, of which the first 80% serve as context and the remaining 20% as query trajectories. The SCM specifies the factual data-generating process, including the behavioural policy through its treatment-assignment mechanisms. The target policy may be a prescribed treatment sequence or a dynamic policy that depends on the observed history. We sample each task in five stages.

1. **Dimension and variable types** We sample the numbers of baseline covariates, time-varying covariates, and binary treatment dimensions as $d_B \in \{1, \dots, 32\}$, $d_L \in \{1, \dots, 32\}$, and $d_A \in \{1, 2, 3\}$, respectively. Time-varying covariates may be continuous or ordinal, while baseline covariates may additionally be nominal.
2. **Skeleton.** The SCM starts with the baseline variables (B, L_1), which are generated either jointly through a random DAG or with B independent of L_1. For each coordinate of the subsequent variables $(A_{1:T}, L_{2:T}, Y_{2:T+1})$, we first sample a time-invariant set of *parent variables*. For each selected parent variable, we then sample a set of lag values, with

[3]Here "causal attention" refers only to temporally masked attention: each token may attend to the past and present, but not the future. Following common usage, we retain this term, although the layer does not itself encode or identify causal relationships among variables.

the number of lags drawn as $1 + \text{Poisson}(\lambda)$ and lag values selected using geometrically decaying weights that favour shorter lags. Each resulting variable–lag pair specifies a *parent node* in the longitudinal SCM. Each L coordinate and Y additionally includes its own preceding-step value as a parent node. We constrain the resulting longitudinal graph to be acyclic: cross-step edges point forward in time, contemporaneous edges among L follow an acyclic ordering, and treatment nodes depend only on preceding nodes in history H_t.

3. **SCM mechanisms and behavioural policy.** We draw random functions from seven families: neural networks, decision trees, discretizations (Hollmann et al., 2025), linear functions, quadratic functions with pairwise interactions, smooth functions sampled from a Gaussian process, and random activation functions (sigmoid, softplus, or exponential) (QU et al., 2026). For covariate and outcome coordinates, we explicitly incorporate persistence through a per-coordinate weight sampled from $U(0.5, 0.95)$ on the preceding-step value, together with a smooth time trend. For a coordinate with treatment parents, its mechanism takes the form $g_0(x) + \sum_i a_i g_i(x)$, where x denotes its non-treatment parents and each $a_i \in \{0, 1\}$ denotes a treatment parent at a particular dimension and lag. The functions g_i are drawn independently, allowing treatment effects to vary with the covariates. The treatment-assignment mechanisms define the behavioural policy. For treatment dimension d, we set

$$P(A_{d,t} = 1 \mid H_t) = \sigma\left(\frac{z_d(H_t) + w_d(2A_{d,t-1} - 1) - c_d}{s_d}\right),$$

where z_d is a randomly drawn function of the history, $w_d \geq 0$ controls treatment persistence, and c_d controls treatment prevalence. We draw $s_d \sim \text{LogUniform}(0.05, 5)$ to span assignment mechanisms ranging from nearly deterministic to nearly randomised, and vary c_d to produce treatment prevalences ranging from rare to common. Finally, we clip the assignment probabilities to $[0.01, 0.99]$ to ensure positivity.

4. **Target policy.** For each treatment dimension, we sample with equal probability either a history-dependent policy with its own random mechanism or a static treatment sequence. The history-dependent policy is constructed with its own randomly sampled parents and mechanism drawn the same family as the behavioural policy. Static sequences include always treating, never treating, and treating only for the first $k \in \{1, \ldots, \tau - 1\}$ steps. In half of the tasks, we subsequently blend the sampled target-policy probabilities toward the behavioural probabilities, using a mixing weight drawn from $U(0.3, 1)$. After any blending with the behavioral probabilities, we threshold the resulting probabilities to obtain a deterministic target-policy rule. This strategy produce target policies that remain close to the behavioural policy.

5. **Rollout and supervision.** We first generate factual trajectories from the sampled SCM, drawing exogenous noise for the baseline variables and subsequent transitions. For each intervention start step s, we retain the observed prefix and roll forward under the target policy for up to $\tau_{\max}$ steps, reusing the same exogenous noise realisations. The resulting single-draw counterfactual outcomes serve as training labels. For a fixed SCM and target policy, their conditional expectation given H_s is the CAPO.

B.3 Hyperparameters and Training

Table S.1 lists the hyperparameters of the model described in Appendix B.1. Every attention module, including the feature-attention layer and the attention pooling, uses the same number of heads. The observed data backbone and the readout update have separate parameters in each of the eight layers; a final layer normalization is applied to the readout tokens before the output head. The model has 5.4M parameters, 4.8M of which lie in the transformer layers and the remainder in the tokenizers, embeddings, and output head. We train with AdamW. The learning rate is warmed up linearly to 0.001 and then halved whenever the validation loss has not improved for 40 epochs. Model selection uses a fixed set of 128 held-out validation tasks drawn from the prior; we report the checkpoint with the lowest validation loss. At inference, a task is processed in a single forward pass with no per-task fitting.

Table S.1: PICLE-PFN Architecture Hyperparameters.

Component	**Value**
Transformer layers	8
Embedding width	128
Attention heads	4
MLP hidden width	512
Activation	GELU
Dropout	0
Learned pooling queries for feature pooling	4
Maximum sequence length T	15
Maximum horizon $\tau_{\max}$	10
Output buckets (bar distribution)	100
Total parameters	5.4 M

Table S.2: **Longitudinal CAPO benchmark, per horizon.** Each cell reports the normalized RMSE, mean±SD across the 20 repeats; values in parentheses are the mean±SD of the method's per-task RMSE ranking (↓ better). The **best** and second best methods per benchmark and horizon are highlighted.

Benchmark	Method	$\tau = 2$	$\tau = 4$	$\tau = 6$	$\tau = 8$	$\tau = 10$
Cancer	**PICLE-PFN**	**0.31±0.03 (1.1±0.2)**	**0.26±0.04 (1.0±0.0)**	**0.18±0.03 (1.0±0.0)**	**0.12±0.03 (1.0±0.0)**	**0.08±0.02 (1.0±0.0)**
	IGC-Net	0.45±0.04 (2.2±0.5)	0.48±0.05 (2.6±0.6)	0.33±0.04 (2.1±0.3)	0.20±0.04 (2.0±0.2)	0.39±0.15 (3.0±0.9)
	CT	0.48±0.04 (2.8±0.4)	0.49±0.05 (2.5±0.6)	0.42±0.07 (3.1±0.6)	0.35±0.08 (3.1±0.4)	0.31±0.09 (2.5±0.7)
	RMSN	0.73±0.04 (4.0±0.0)	0.60±0.06 (3.8±0.6)	0.54±0.10 (3.9±0.5)	0.56±0.15 (4.4±0.9)	0.58±0.18 (4.5±1.2)
	G-Net	0.88±0.02 (5.0±0.0)	0.77±0.04 (5.0±0.0)	0.71±0.05 (5.2±0.5)	0.70±0.05 (5.8±0.4)	0.70±0.05 (5.8±0.6)
	CRN	1.06±0.04 (6.8±0.4)	1.00±0.10 (6.3±0.5)	0.74±0.08 (5.8±0.4)	0.63±0.07 (4.6±0.6)	0.58±0.06 (4.2±0.6)
	TE-CDE	1.04±0.02 (6.2±0.4)	1.04±0.05 (6.7±0.5)	1.03±0.05 (7.0±0.0)	1.04±0.06 (7.0±0.0)	1.05±0.06 (7.0±0.0)
MIMIC_Y1	**PICLE-PFN**	0.63±0.33 (1.7±1.0)	0.85±0.45 (1.9±1.1)	1.04±0.68 (1.9±1.1)	1.17±0.91 (2.0±1.1)	**1.28±1.09 (2.2±1.3)**
	IGC-Net	**0.59±0.14 (1.8±0.8)**	**0.76±0.28 (1.7±0.8)**	**0.98±0.55 (1.8±1.1)**	**1.15±0.94 (2.1±1.9)**	1.32±1.19 (2.1±2.0)
	CT	0.92±0.23 (4.4±1.8)	1.43±0.45 (5.2±2.0)	1.78±0.65 (5.5±1.8)	1.98±0.74 (5.4±1.9)	2.13±0.87 (5.2±2.0)
	RMSN	0.99±0.34 (4.6±1.2)	1.33±0.47 (4.6±1.4)	1.58±0.71 (4.5±1.6)	1.73±0.96 (4.2±1.5)	1.80±1.08 (4.2±1.5)
	G-Net	1.27±0.40 (6.2±0.9)	1.36±0.53 (4.7±1.4)	1.50±0.71 (4.5±1.2)	1.64±0.86 (4.2±1.1)	1.75±0.94 (4.0±0.8)
	CRN	0.99±0.18 (4.8±1.7)	1.38±0.46 (5.0±1.9)	1.67±0.72 (4.7±2.0)	1.87±0.86 (4.9±1.9)	2.03±0.96 (4.8±1.9)
	TE-CDE	1.02±0.47 (4.5±1.6)	1.47±0.55 (4.8±1.4)	1.83±0.71 (5.2±1.3)	2.09±0.88 (5.2±1.2)	2.28±1.08 (5.3±1.4)
MIMIC_Y2	**PICLE-PFN**	**0.38±0.15 (1.3±0.6)**	**0.51±0.20 (1.1±0.4)**	**0.64±0.27 (1.1±0.4)**	0.79±0.39 (1.8±1.3)	0.91±0.51 (1.9±1.4)
	IGC-Net	0.41±0.16 (2.1±0.7)	0.56±0.15 (2.2±0.7)	0.92±0.34 (3.4±1.5)	**0.72±0.16 (1.9±0.7)**	**0.80±0.17 (1.9±0.8)**
	CT	0.59±0.48 (3.0±1.4)	0.82±0.67 (3.2±1.4)	1.02±0.78 (3.0±1.7)	1.18±0.88 (3.5±1.7)	1.31±0.94 (3.9±1.7)
	RMSN	0.79±0.21 (4.5±1.2)	0.93±0.24 (4.3±1.0)	1.03±0.24 (4.1±1.4)	1.12±0.27 (4.4±1.4)	1.19±0.32 (4.3±1.8)
	G-Net	0.94±0.10 (5.2±0.9)	1.14±0.14 (5.5±1.0)	1.27±0.17 (5.5±1.2)	1.37±0.19 (6.0±1.4)	1.43±0.20 (6.0±1.5)
	CRN	1.13±0.35 (5.7±0.9)	1.19±0.46 (5.2±1.1)	1.26±0.50 (4.7±1.0)	1.33±0.53 (4.5±1.0)	1.41±0.58 (4.7±1.2)
	TE-CDE	1.31±0.42 (6.2±1.2)	1.56±0.47 (6.4±1.0)	1.56±0.48 (6.2±1.3)	1.50±0.52 (6.0±0.9)	1.46±0.58 (5.3±1.2)
MIMIC_1T	**PICLE-PFN**	0.38±0.13 (1.6±1.1)	0.55±0.22 (1.9±1.7)	**0.63±0.20 (1.6±1.7)**	0.70±0.26 (1.8±1.7)	0.73±0.24 (1.9±2.1)
	IGC-Net	0.43±0.08 (3.5±1.0)	**0.52±0.12 (2.2±0.6)**	0.90±0.33 (5.0±2.4)	**0.69±0.14 (2.2±1.2)**	**0.71±0.11 (2.1±1.1)**
	CT	0.47±0.26 (3.8±1.5)	0.67±0.47 (3.9±1.4)	0.79±0.56 (2.8±1.7)	0.84±0.56 (3.2±1.6)	0.89±0.57 (3.5±1.6)
	RMSN	0.55±0.13 (5.2±0.9)	0.75±0.17 (5.5±0.8)	0.87±0.16 (5.1±1.5)	0.96±0.18 (5.8±1.7)	1.03±0.21 (6.0±1.7)
	G-Net	1.01±0.07 (8.2±1.2)	1.08±0.13 (7.8±1.5)	1.19±0.22 (7.8±1.7)	1.20±0.21 (7.5±1.8)	1.20±0.19 (7.2±1.9)
	CRN	0.59±0.32 (5.5±1.3)	0.84±0.52 (5.9±1.8)	0.99±0.62 (5.0±2.3)	1.12±0.65 (5.8±2.2)	1.23±0.64 (6.4±2.1)
	TE-CDE	1.04±0.05 (8.8±0.9)	1.19±0.13 (9.1±1.2)	1.34±0.19 (9.1±1.3)	1.37±0.14 (9.1±1.1)	1.38±0.11 (9.2±1.0)
	D3	1.07±0.11 (8.4±1.4)	1.10±0.10 (7.5±1.3)	1.15±0.12 (7.2±1.3)	1.24±0.17 (7.5±1.3)	1.24±0.16 (7.3±1.8)
	DLTMLE	1.04±0.11 (8.2±1.3)	1.18±0.20 (8.3±1.3)	1.31±0.71 (7.3±1.4)	1.54±0.71 (8.1±1.5)	1.19±0.20 (7.0±1.5)
	DeepACE	**0.37±0.08 (1.8±0.5)**	0.56±0.19 (3.0±2.0)	1.23±1.72 (4.1±2.6)	0.78±0.14 (4.0±1.4)	0.86±0.23 (3.9±1.8)
MIMIC_1T_HDP0.5	**PICLE-PFN**	**0.32±0.10 (1.0±0.0)**	**0.46±0.14 (1.0±0.0)**	**0.56±0.15 (1.0±0.0)**	**0.64±0.15 (1.1±0.2)**	**0.69±0.15 (1.1±0.3)**
	IGC-Net	0.42±0.12 (2.0±0.0)	0.73±0.19 (2.3±0.7)	0.62±0.16 (2.0±0.2)	0.68±0.14 (1.9±0.2)	0.72±0.14 (1.9±0.3)
	D3	0.90±0.07 (3.5±0.5)	0.95±0.05 (3.5±0.8)	0.98±0.07 (3.6±0.6)	0.99±0.06 (3.4±0.5)	1.01±0.07 (3.5±0.5)
	DLTMLE	0.90±0.07 (3.5±0.5)	0.95±0.05 (3.2±0.4)	0.98±0.05 (3.4±0.5)	1.00±0.04 (3.6±0.5)	1.02±0.06 (3.5±0.5)
MIMIC_1T_HDP1	**PICLE-PFN**	**0.33±0.10 (1.0±0.0)**	**0.47±0.14 (1.0±0.0)**	**0.58±0.16 (1.0±0.0)**	**0.67±0.17 (1.1±0.3)**	**0.72±0.16 (1.1±0.4)**
	IGC-Net	0.42±0.12 (2.0±0.0)	0.71±0.24 (2.2±0.6)	0.64±0.17 (2.0±0.0)	0.71±0.17 (1.9±0.3)	0.77±0.16 (1.9±0.4)
	D3	0.89±0.07 (3.2±0.4)	0.94±0.04 (3.0±0.4)	1.00±0.05 (3.3±0.5)	1.00±0.05 (3.3±0.5)	1.01±0.08 (3.1±0.7)
	DLTMLE	0.90±0.07 (3.8±0.4)	0.97±0.05 (3.8±0.6)	1.02±0.05 (3.7±0.5)	1.05±0.09 (3.7±0.5)	1.06±0.10 (3.8±0.4)
MIMIC_1T_HDP2	**PICLE-PFN**	**0.33±0.10 (1.0±0.0)**	**0.51±0.16 (1.0±0.0)**	**0.65±0.19 (1.1±0.3)**	**0.77±0.23 (1.4±0.5)**	0.85±0.25 (1.6±0.6)
	IGC-Net	0.48±0.16 (2.1±0.4)	0.64±0.17 (2.0±0.2)	0.71±0.21 (1.9±0.3)	0.80±0.20 (1.6±0.5)	**0.85±0.20 (1.5±0.5)**
	D3	0.86±0.09 (3.5±0.5)	0.93±0.12 (3.1±0.5)	1.03±0.19 (3.1±0.4)	1.08±0.22 (3.1±0.3)	1.11±0.21 (3.1±0.5)
	DLTMLE	0.87±0.08 (3.5±0.6)	0.98±0.13 (3.8±0.4)	1.14±0.20 (3.9±0.4)	1.28±0.28 (3.9±0.3)	1.29±0.26 (3.8±0.4)

C Additional Results

Full CAPO RMSE results across all methods and all horizons are provided in Table S.2. Detailed runtime results across horizons are in Table S.3.

Table S.3: **Runtime(seconds) at deployment.** One evaluation task per setting (1000 context, 1000 queries), one A100 GPU, mean±SD over 5 timing repeats. Baselines are measured on their tuned HPs. CT, RMSN, G-Net, CRN, and TE-CDE is trained once then evaluated on all five horizons; runtime is computed as total time of one training and five inference calls, divided by five); IGC-net, D3, DLTMLE, and DeepACE train one model per horizon; runtime is measured as one training and one inference call. PICLE-PFN is pretrained and only runs inference.

Method	Type	Cancer			MIMIC_Y1			MIMIC_1T		
		train	infer	total	train	infer	total	train	infer	total
PICLE-PFN	amortised	—	0.13±0.00	**0.13±0.00**	—	0.14±0.00	**0.14±0.00**	—	0.14±0.00	**0.14±0.00**
IGC-Net	fit per horizon	525±11	0.08±0.00	525±11	582±5	0.08±0.01	582±5	725±10	0.09±0.02	725±10
CT	fit once	23.84±0.57	0.25±0.03	24.09±0.57	78.82±0.47	1.08±0.02	79.90±0.46	66.46±2.04	1.26±0.11	67.72±1.96
RMSN	fit once	65.62±1.36	0.10±0.01	65.71±1.35	33.96±0.55	0.07±0.00	34.03±0.55	6.26±0.49	0.08±0.00	6.34±0.50
G-Net	fit once	4.31±0.42	6.58±0.73	10.88±0.47	12.00±0.45	6.62±0.18	18.62±0.40	4.87±0.37	8.45±0.56	13.32±0.41
CRN	fit once	32.85±0.34	0.10±0.01	32.95±0.34	4.97±0.16	0.13±0.00	5.10±0.17	21.64±0.63	0.13±0.03	21.77±0.62
TE-CDE	fit once	7.76±0.53	0.13±0.01	7.90±0.53	52.86±0.29	0.11±0.00	52.97±0.29	15.46±0.46	0.12±0.00	15.58±0.46
D3	fit per horizon	—	—	—	—	—	—	12.29±0.52	0.14±0.02	12.42±0.53
DLTMLE	fit per horizon	—	—	—	—	—	—	7.85±0.30	0.14±0.02	7.98±0.28
DeepACE	fit per horizon	—	—	—	—	—	—	12.48±0.45	0.02±0.00	12.51±0.45

D Experiment Details

D.1 Benchmark Datasets

Our benchmark settings are adapted from Hess et al. (2026), including one fully synthetic setting and six semi-synthetic setting based on MIMIC-III. For each benchmark, we generate 20 datasets, each containing 1000 training (context) trajectories and 1000 test (query) trajectories of length $T = 15$. We estimate CAPO on the query trajectories at five horizons, $\tau \in \{2, 4, 6, 8, 10\}$, with the intervention start time fixed at $s = 2$. RMSE is first divided by the factual outcome in each synthesized dataset, then macro-averaged across 20 datasets. Mean±SD of the normalized RMSE is reported.

D.1.1 CANCER

A fully synthetic tumor-growth simulator commonly used in causal inference literature (Hess et al., 2026; Bica et al., 2020; Li et al., 2021; Lim, 2018; Melnychuk et al., 2022). It has a single time-varying covariate (tumor volume), two binary treatment dimensions (chemotherapy, radiotherapy), and one baseline covariate (patient type). The target policy in this setting is a *static* always-combined-treatment. The outcome, tumor volume Y_t, evolves as

$$Y_{t+1} = Y_t \left[1 + \rho \log\left(\frac{K}{Y_t}\right) - \alpha_c c_t - \left(\alpha_r d_t + \beta_r d_t^2\right) + \epsilon_t\right],$$

where ρ and K govern tumor growth and carrying capacity, respectively. Treatment effects depend on the chemotherapy drug concentration c_t and radiation dose d_t, with sensitivity parameters α_c, α_r, and β_r. The term ϵ_t introduces stochastic variation.

Treatment assignment depends on recent tumor progression, introducing time-varying confounding. Specifically, the binary chemotherapy and radiotherapy indicators are sampled as

$$A_t^c, A_t^r \sim \text{Bernoulli}\left[\sigma\left(\frac{\gamma}{D_{\max}}\left\{\bar{D}_{15}(\bar{Y}_{t-1} - \frac{D_{\max}}{2})\right\}\right)\right],$$

where σ is the logistic function, $D_{\max}$ is the maximum tumour volume, and $\bar{D}_{15}(\bar{Y}_{t-1})$ denotes the mean tumor diameter over the preceding 15 time steps. γ controls how strongly treatment assignment depends on tumor history. The target policy is always treating with both chemotherapy and radiation. We adopt the simulator parameterization of Hess et al. (2026), to which we refer for further implementation details.

D.1.2 MIMIC_Y1, MIMIC_Y2

These benchmarks, adapted from Hess et al. (2026), combine real-world clinical covariate trajectories with simulated treatments and outcomes. The real-world covariates are obtained using MIMIC-Extract (Wang et al., 2020) and comprise 25 time-varying vital signs, together with gender, ethnicity,

and age as static features. Together with the synthetic outcome and treatments, these two benchmark is characterized by 27 time-varying confounders and 3 treatments.

The simulator first constructs two untreated outcome processes, indexed by $j \in \{1, 2\}$:

$$\widetilde{Y}_t^j = \alpha_s^j \,\text{B-spline}(t) + \alpha_g^j g^j(t) + \alpha_f^j f_Y^j(X_t) + \epsilon_t. \tag{2}$$

The coefficients α_s^j, α_g^j, and α_f^j weight the respective components. The spline component is drawn from a mixture of three cubic splines, while f_Y^j is sampled using a Random Fourier Feature (RFF) approximation to a Gaussian process.

Three binary treatments, indexed by $l \in \{1, 2, 3\}$, are then assigned according to

$$A_t^l \sim \text{Bernoulli}\left(\sigma(\gamma_Y^l Y_{t-1}^{A,l} + \gamma_X^l f_Y^l(X_t) + b^l)\right), \tag{3}$$

where σ denotes the logistic function. Here, $Y_{t-1}^{A,l}$ is an averaged subset of previous l treated outcomes, and f_Y^l is a again a RFF approximation to a Gaussian process. The coefficients γ_Y^l and γ_X^l determine the confounding strength of A^l, while b^l sets the treatment-specific intercept.

Finally, the simulator adds treatment effects to the untreated outcome trajectories via

$$Y_t^j = \widetilde{Y}_t^j + \sum_{i=t-\omega^l}^{t} \frac{\min_{l=1,\dots,d_a} \mathbf{1}_{\{A_i^l=1\}}\, p_i^l\, \beta^{l,j}}{(\omega^l - i)^2}. \tag{4}$$

For treatment l, the parameter $\beta^{l,j}$ controls the maximum effect on outcome j, and ω^l specifies its effect window.

The two simulated outcomes, Y^1 and Y^2, are governed by different confounders in the real-world covariates; thus we evaluate them as separate benchmarks: `mimic_y1` targets Y^1, and `mimic_y2` targets Y^2. The simulator parameters follow Hess et al. (2026), which we refer to for the full specification.

Both `mimic_y1` and `mimic_y2` have 3-dim treatments at each time step; in both settings, we specify the target policy to 'always apply the second and third treatments', i.e. $A_t = (0, 1, 1)$ for τ steps.

D.1.3 MIMIC_1T, MIMIC_1T_HDP0.5, MIMIC_1T_HDP1, MIMIC_1T_HDP2

To accommodate baselines that require a single treatment variable, we construct four variants of `mimic_y2` with a single binary treatment. We retain its simulator parameters and outcome-generation mechanism, but remove treatments A^1 and A^2 and their contributions to Y^2. The remaining treatment, A^3, is assigned according to the original behavior policy.

The four benchmarks share this data-generating process and differ only in their target policies. The `mimic_1t` benchmark uses an always-treat policy, enabling comparisons with baselines restricted to static treatment sequences. The other three benchmarks use deterministic dynamic policies obtained by shifting the logit of the behavior policy:

$$A_t^3 = \mathbb{1}\left\{\gamma_Y^3 Y_{t-1}^{A,3} + \gamma_X^3 f_Y^3(X_t) + b^3 > \zeta\right\}. \tag{5}$$

We set $\zeta \in \{0.5, 1, 2\}$, with the value appended to `hdp` in each benchmark name. Larger values of ζ increase the target treatment probability relative to the behavior policy for a given history. $\zeta = 0$ will yield a deterministic rule that follows the behavior policy's decision boundary.

D.2 Baseline Implementations

We compare against nine baselines. **CRN** (Bica et al., 2020), **RMSN** (Lim, 2018), **TE-CDE** (Seedat et al., 2022), **G-Net** (Li et al., 2021), and **CT** (Melnychuk et al., 2022) are taken from the CT codebase (Melnychuk et al., 2022); **D3-Net** (Chen et al., 2026a), **DeepLTMLE** (Shirakawa et al., 2024), and **DeepACE** (Frauen et al., 2023) from the D3-Net codebase (Chen et al., 2026a); and **IGC-Net** (Hess et al., 2026) from the authors' public implementation.

There is one major change in our implementation of IGC-Net. As published, it estimates the effect of a single *static* treatment sequence shared by every unit. Neither the paper nor the implementation considers dynamic target policies. However, we noticed that it can be readily extended to support such a setting by modifying a step in data preprocessing. Specifically, we extend IGC-Net to our setting by replacing the hardcoded treatment sequence with the per-trajectory treatment decision of the target policy π, leaving the architecture and training objective untouched. This extension goes beyond the published scope of IGC-Net and is made in its favour: it is what allows IGC-Net to be run on every setting, including those with a history-dependent target policy.

G-Net supports dynamic treatment regimes in principle: its Monte Carlo rollout simulates future covariates, on which a regime could be evaluated. The released implementation, however, fixes the full treatment sequence before inference and never re-evaluates a policy on the simulated state, so supporting history-dependent regimes would require non-trivial changes to its inference path. We therefore do not run G-Net in this setting. This omission does not affect our conclusions: under static treatment sequences on the same data (Table S.2, `mimic_1t`), G-Net already underperforms the leading baselines by a wide margin at every horizon.

CRN, RMSN, TE-CDE, G-Net, and CT are encoder–decoder models that are *trained once per task* on the full trajectory window ($T = 15$ steps) and then evaluated at every horizon $\tau \in \{2, 4, 6, 8, 10\}$. D3-Net, DeepLTMLE, DeepACE, and IGC-Net are based on ICE G-computation, which requires horizon-specific architectures, so a separate model is trained for every (setting, horizon) cell.

Not every baseline runs on every setting. D3-Net, DeepLTMLE, and DeepACE are hardwired to a single binary treatment (their propensity head, inverse propensity ratio, and the debiasing objective all assume single-dimensional treatment), so they are evaluated only on the single-treatment MIMIC settings. On the settings with a history-dependent target policy, only IGC-Net, D3-Net, and DeepLTMLE are run: the encoder–decoder models condition on a fixed treatment sequence and cannot take a target policy that depends on the evolving state. Cells that a method cannot serve are left blank in the results tables.

D.3 Hyperparameter Tuning and Evaluation Protocol

For every baseline, we tune its hyperparameters with Optuna (Akiba et al., 2019) for 2 hours on one A100 GPU, using one held-out tuning task drawn from the same generator as, and disjoint from, the evaluation tasks. The selection criterion is held-out *factual* RMSE: counterfactual labels are never used for model or hyperparameter selection. Hyperparameters are tuned once per setting for the encoder–decoder models, and once per (setting, horizon) cell for ICE G-computation-based methods. For instance, IGC-Net, uses $5 \times 2 = 10$ hours of tuning per setting. The search spaces of every baseline are listed in Table S.4.

The selected hyperparameters are then fixed, and a fresh model is trained on each of the 20 evaluation tasks. Every task contains 2000 trajectories, split into 1000 training trajectories and 1000 test trajectories. The baseline is trained on the training half, and scored on the Monte-Carlo simulated CAPO on the test half. This is the same split our model uses: the training half is its in-context set, and the test half its query set, so every method sees identical data.

D.4 Runtime measurement

We additionally compare wall-clock deployment cost, using a dedicated single-GPU timing benchmark that reuses each model's already-tuned HPs (so HP tuning time itself is excluded). PICLE-PFN's cost is inference-only: forward passes of the single fixed checkpoint over a task's query trajectories, with no gradient update. We repeated five forward passes and report the mean. Baselines' cost is training plus inference: each train/inference call is likewise repeated 5 times and averaged.

The architectures of CRN, TE-CDE, CT, RMSN, G-Net are not horizon-specific, so they require one training and 5 inferences, one for each target horizon, matching how they are evaluated. Their runtime is divided over five to represent the average runtime for each target horizon. IGC-Net, D3, D-LTMLE, and DeepACE require separate horizon-specific training for each setting, so they are timed separately at each of the five horizons, and we report the mean of training + inference time across horizons.

Table S.4: Hyperparameter search spaces for the baselines. Each space is searched with Optuna for 2 h on one A100, selecting by held-out factual RMSE. $\mathcal{U}$ and $\log\mathcal{U}$ denote uniform and log-uniform ranges; braces denote categorical choices.

Method	Hyperparameter	Search space
CRN, TE-CDE, G-Net	learning rate	$\log\mathcal{U}[10^{-4},\ 10^{-2}]$
	number of layers	$\{1, 2, 3\}$
	batch size	$\{32, 64, 128, 256\}$
	sequence hidden units	$\{0.5, 1, 2, 3, 4\}\times$ input dim
	balanced representation size	$\{0.5, 1, 2, 3, 4\}\times$ input dim
	FC hidden units	$\{0.5, 1, 2, 3, 4\}\times$ input dim
	dropout	$\mathcal{U}[0,\ 0.5]$
CT	learning rate	$\log\mathcal{U}[10^{-4},\ 10^{-2}]$
	number of layers	$\{1, 2, 3\}$
	attention heads	$\{1, 2, 3\}$
	batch size	$\{32, 64, 128, 256\}$
	sequence hidden units	$\{0.5, 1, 2, 3, 4\}\times$ input dim
	balanced representation size	$\{0.5, 1, 2, 3, 4\}\times$ input dim
	FC hidden units	$\{0.5, 1, 2, 3, 4\}\times$ input dim
	dropout	$\mathcal{U}[0,\ 0.5]$
RMSN	learning rate	$\log\mathcal{U}[10^{-4},\ 10^{-2}]$
	number of layers	$\{1, 2, 3\}$
	batch size	$\{32, 64, 128, 256\}$
	sequence hidden units	$\{0.5, 1, 2, 3, 4\}\times$ input dim
	dropout	$\mathcal{U}[0,\ 0.5]$
	max. gradient norm	$\{0.5, 1, 2\}$
D3-Net, DeepLTMLE	learning rate	$\log\mathcal{U}[10^{-4},\ 10^{-2}]$
	hidden size	$\{16, 32, 64\}$
	number of layers	$\{1, 2, 3\}$
	attention heads	$\{2, 4\}$
	embedding dimension	$\{8, 16, 32, 64, 128\}$
	dropout	$\mathcal{U}[0,\ 0.5]$
	propensity loss weight α	$\mathcal{U}[0.5,\ 2]$
DeepACE	learning rate	$\log\mathcal{U}[10^{-4},\ 10^{-2}]$
	LSTM hidden size	$\{16, 32, 64\}$
	body hidden size	$\{16, 32\}$
	head hidden size	$\{8, 16, 32\}$
	dropout	$\mathcal{U}[0,\ 0.5]$
	propensity loss weight α	$\mathcal{U}[0.5,\ 1]$
	targeting loss weight β	$\mathcal{U}[0.5,\ 1]$
IGC-Net	learning rate	$\log\mathcal{U}[10^{-4},\ 10^{-2}]$
	number of layers	$\{1, 2, 3\}$
	attention heads	$\{1, 2\}$
	sequence hidden units	$\{16, 32, 48\}$
	hidden representation size	$\{8, 16, 32\}$
	FC hidden units	$\{8, 16, 32\}$
	dropout	$\mathcal{U}[0,\ 0.5]$
	batch size	$\{32, 64, 128, 256\}$

D.5 Real-world MIMIC prediction

We follow the cohort construction and data extraction procedures of Hess et al. (2026), to which we refer readers for additional details, and define a factual outcome prediction task as follows.

Cohorts and Data We extract each patient's hourly trajectory over the first 11 hours. Missing values are imputed by forward- and backward-filling within each trajectory, and patients with any remaining missing outcome or covariate values are excluded. We then sample ten cohorts of $n = 3000$ patients each, without replacement within each cohort. Each cohort is split by row order into

1800 training, 600 validation, and 600 test patients. All cohorts and splits are generated once and shared across methods.

We consider two treatments: vasopressor use and mechanical ventilation. The outcome is hourly mean diastolic blood pressure (DBP, mmHg). Time-varying covariates comprise DBP and the 25 vital signs and laboratory measurements selected by Hess et al. (2026) ($d_L = 26$). Static covariates comprise gender, ethnicity, and age.

Prediction task. We evaluate multi-step prediction of factual outcomes under the behavior policy. We fix the start hour at $s = 5$ and evaluate horizons $\tau \in \{2, 3, 4, 5\}$, corresponding to observed DBP at endpoint hours $\{7, 8, 9, 10\}$. For each cohort and horizon, we report RMSE over the 600 test patients, normalized by the standard deviation of their observed endpoint DBP.

Model Fitting IGC-Net is the only baseline that supports both multivariate treatments and dynamic policies, so we only compare with it. We use its architecture as released, but replace each patient's target treatment sequence with their own factual treatment sequence when constructing the iterative G-computation targets. This adapts their algorithm to target outcomes under the behavior policy without explicitly estimating that policy. For each cohort and target horizon, we tune its HP for 2h using the training and validation sets, refit it on their union using the selected HPs, and evaluate it on the test set.

For PICLE-PFN, the training and validation patients jointly form the context set, and the test patients form the query set. We use the context patients' factual treatment sequences as policy demonstrations, providing observed history–action samples from the behavior policy. Neither method explicitly fits a behavior-policy model, avoiding an additional source of estimation error in the comparison.

D.6 Uncertainty calibration.

We measure the calibration of PICLE-PFN and the strongest baseline, IGC-Net. PICLE-PFN outputs a full predictive distribution in a single forward pass, from which the equal-tailed $100(1-\alpha)\%$ interval is read off directly. IGC-Net produces only a point estimate, so we follow the paper and obtain intervals by Monte-Carlo dropout (1000 stochastic forward passes at the tuned dropout rate)[4]. A query trajectory is *covered* if its true CAPO lies inside the interval, and the empirical coverage of a task is the fraction of covered query trajectories. Coverage is measured on the datasets from our own prior, `mimic_y1`, and `mimic_1t`, across horizon $\tau \in \{2, 4, 6, 8, 10\}$. Results on each (dataset, horizon) cell is averaged over 20 tasks.

Major results are reported in Figure 4. We notice that some sampled tasks have few context trajectories following the target policy over the prediction window. This extreme positivity violation forces any method to extrapolate, and it becomes more frequent as the horizon grows. These cases account for the largest deviations from nominal coverage, in either direction (Figure S.2).

D.7 Unobserved Confounding

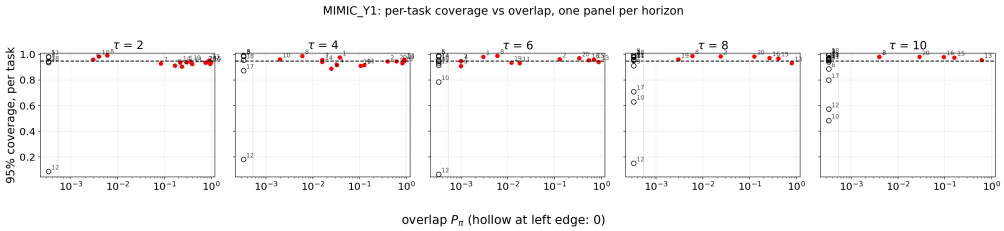
We examine how PICLE-PFN and the two strongest baselines, CT and IGC-Net, degrade when unobserved confounders are present. On `mimic_1t`, we draw a patient-level latent severity $U \sim \mathcal{N}(0, 1)$ that is hidden from all methods and enters the data-generating process twice by shifting the patient's outcome by λU at every step,

$$Y_t^j = \widetilde{Y}_t^j + \sum_{i=t-\omega^l}^{t} \frac{\min_{l=1,\dots,d_a} \mathbf{1}_{\{A_i^l=1\}}\, p_i^l\, \beta^{l,j}}{(\omega^l - i)^2} + \lambda U. \tag{6}$$

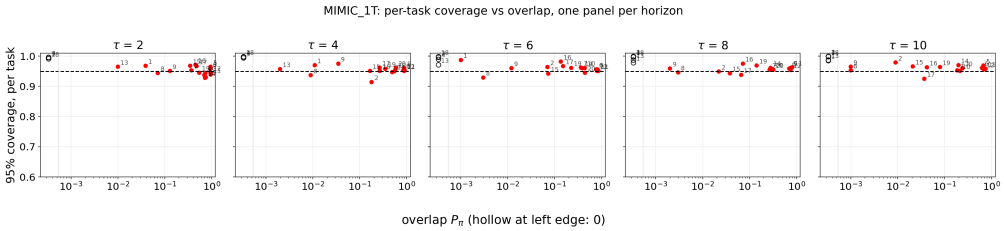
and the logit of the behavior propensity by γU at every step,

$$A_t^l \sim \text{Bernoulli}\left(\sigma(\gamma_Y^l Y_{t-1}^{A,l} + \gamma_X^l f_Y^l(X_t) + b^l + \gamma U).\right), \tag{7}$$

[4]While these MC dropout intervals are the uncertainty-quantification procedure proposed by the original paper, they should not be interpreted as confidence intervals. Bootstrapping could instead be used to construct confidence intervals, but would require resampling the observed data and retraining IGC-Net on every resample, which is computationally prohibitive given its long training time.



(a) `mimic_y1`



(b) `mimic_1t`

Figure S.2: **Coverage against overlap** Overlap (x-axis) is measured as the proportion of observed trajectories that exactly match the target policy, and coverage (y-axis) is evaluated using nominal 95% intervals. Hollow points denote tasks with zero empirical overlap. Tasks with positive overlap generally achieve coverage above 0.9, whereas zero-overlap tasks exhibit the largest deviation from nominal coverage, in either direction.

Table S.5: **Context sample size under a static and a history-dependent target policy.** RMSE on the query split (size=1000), mean±SD over 20 tasks (↓ better).CT cannot handle dynamic policies and is not run on the history-dependent setting.

		`mimic_1t`			`mimic_1t_hdp1`	
Horizon	n context	**PICLE-PFN**	CT	IGC-Net	**PICLE-PFN**	IGC-Net
$\tau = 2$	100	**0.453±0.148**	0.581±0.142	0.600±0.099	**0.402±0.129**	0.520±0.102
	300	**0.380±0.086**	0.500±0.107	0.531±0.099	**0.348±0.106**	0.446±0.119
	1000	**0.375±0.121**	0.475±0.260	0.425±0.084	**0.326±0.100**	0.420±0.120
$\tau = 4$	100	**0.633±0.213**	0.747±0.175	0.833±0.166	**0.573±0.179**	0.656±0.167
	300	**0.532±0.129**	0.633±0.128	0.613±0.120	**0.504±0.154**	0.588±0.151
	1000	0.535±0.203	0.667±0.472	**0.517±0.116**	**0.475±0.142**	0.706±0.245

We fix $\lambda = 0.5$ and vary the confounding strength over $\gamma \in \{1, 2, 4\}$, using the original evaluation without unobserved confounding ($\gamma = \lambda = 0$) as the reference.

E Comparison with CausalLongPFN

Concurrent work, CausalLongPFN (Zare et al., 2026), also uses a PFN to estimate CAPO. However, their method only supports static treatment sequences, whereas PICLE-PFN supports both static and dynamic treatment policies. More importantly, their approach, in general, do not recover the multi-step CAPO.

For a prediction horizon greater than one, CausalLongPFN recursively applies a one-step outcome predictor. At each step from the start to the end time step, it inserts the predicted outcome mean into the next outcome history while leaving future time-varying covariates masked. This plug-in rollout does not, in general, identify the multi-step CAPO. To see this, consider a simplified setting where the only time-varying confounder is the intermediate outcome itself, yielding the trajectory $O = (B, A_1, Y_2, A_2, Y_3)$. Support a target treatment sequence is (a_1, a_2). At the second step, CausalLongPFN's one-step predictor models the conditional distribution of Y_3 given

Table S.6: Our PFN vs. CausalLongPFN on `cancer`: normalized RMSE (mean ± std over 20 tasks) and inference runtime per task (1000 context / 1000 query trajectories, one A100).

	Normalized RMSE ↓		Runtime (s) ↓	
Horizon τ	PICLE-PFN	CausalLongPFN	PICLE-PFN	CausalLongPFN
1	0.275 ± 0.026	$\mathbf{0.195} \pm 0.020$	**0.039**	6.67
2	0.308 ± 0.030	$\mathbf{0.278} \pm 0.025$	**0.051**	12.49
3	$\mathbf{0.298} \pm 0.037$	0.306 ± 0.026	**0.065**	18.80
4	$\mathbf{0.264} \pm 0.038$	0.298 ± 0.030	**0.085**	24.75
5	$\mathbf{0.216} \pm 0.032$	0.278 ± 0.035	**0.100**	31.18

(B, A_1, Y_2, A_2). Suppose the one-step predictor perfectly recovers the true conditional mean:

$$g(b, a_1, y_2, a_2) = \mathbb{E}[Y_3 \mid B = b, A_1 = a_1, Y_2 = y_2, A_2 = a_2].$$

The target CAPO, $\mathbb{E}[Y_3^{a_1,a_2} \mid B = b]$, satisfies

$$\mathbb{E}[Y_3^{a_1,a_2} \mid B = b] = \mathbb{E}[g(b, a_1, Y_2, a_2) \mid B = b, A_1 = a_1].$$

The critical flaw in their one-step mean rollout approach is that, it inserts the first-step predicted mean $\widehat{y}_2 = \mathbb{E}[Y_2 \mid B = b, A_1 = a_1]$ and returns $g(b, a_1, \widehat{y}_2, a_2)$. In general,

$$\mathbb{E}[g(b, a_1, Y_2, a_2) \mid B = b, A_1 = a_1] \neq g(b, a_1, \mathbb{E}[Y_2 \mid B = b, A_1 = a_1], a_2).$$

Thus, even with exact one-step conditional means, mean plug-in rollout does not generally recover the multi-step CAPO.

Additionally, the released CausalLongPFN checkpoint supports at most 10 time-varying covariates and a maximum prediction horizon of five. Our MIMIC benchmark contains 27 time-varying covariates and is therefore incompatible with its input interface. Consequently, among our benchmarks, we can evaluate CausalLongPFN only on `cancer` and restrict the comparison to $\tau_{\max} = 5$. This benchmark is less sensitive to the flaw in their one-step mean rollout approach, because it only has one time-varying confounder, which is the outcome itself in the intermediate step. Table S.6 shows that CausalLongPFN achieves lower RMSE at $\tau \in \{1, 2\}$, whereas PICLE-PFN achieves lower RMSE at longer horizons. This pattern is consistent with the limitation of mean plug-in rollout discussed above.

CausalLongPFN also requires substantially longer inference time at every evaluated horizon. Two things of its method contribute to this cost. First, multi-step prediction requires multiple forward pass because CausalLongPFN needs sequential one-step model evaluations, whereas PICLE-PFN estimates all supported horizons in a single forward pass. Second, its context tokens attend to the query token, making their representations query-dependent. Consequently, the full context set is re-encoded for each query trajectory at every rollout step, rather than being encoded once and reused across queries.

F Complexity

Let N be the number of trajectories (context and query, processed jointly), T the maximum length, $\tau_{\max} \leq T$ the maximum prediction horizon, and $d = \max\{d_L, d_S\} > d_A$ the covariate dimension. Feature pooling self-attends among the covariates of each time step, costing $O(NTd^2)$. In the observed stream, causal attention and row attention cost $O(NT^2)$ and $O(TN^2)$. Each trajectory carries $K = T\tau_{\max} - \tau_{\max}(\tau_{\max} - 1)/2 = \Theta(T\tau_{\max})$ readout tokens, one per (start, endpoint) window; their cross-attention to the observed stream costs $O(NKT)$ and their row attention $O(KN^2)$. In total,

$$O\big(NTd^2 + NT^2 + TN^2 + NT^2\tau_{\max} + T\tau_{\max}N^2\big) = O\big(NTd^2 + T\tau_{\max}N^2\big) \quad \text{for } T \ll N,$$

which is quadratic in the number of trajectories, linear in the sequence length for a fixed horizon cap, and at most $O(NTd^2 + T^2N^2)$ for the full triangle $\tau_{\max} = T$. Since row attention uses only the N_c context trajectories as keys, its exact cost is $O(T\tau_{\max}NN_c)$, i.e. linear in the number of queries for a fixed context.