Competing Event Models: Next Event Prediction Under Interventions

Abstract

Modeling interventions that include treatment times is an important task in many domains such as healthcare, finance, and others. Scaling up models for these interventional distributions is challenging, since popular architectures, e.g. transformers that predict the next event in a sequence, do not naturally support alterting times of specific treatments. We develop *competing event models*, an autoregressive generative approach in which estimating interventions in both *when* and what treatments are applied is made simple. The key element in our solution draws from the competing risks literature and models the timing of each type of event, e.g. a treatment or an observation, given that its the next to occur. This design allows straightforward treatment timing interventions via next-token prediction, admits a simple likelihood-based objective, and yields valid effect estimates under standard assumptions. We evaluate on a simulated benchmark for effect estimation of sequential treatments.

1. Introduction

Autoregressive models are the de-facto solution for generation of sequential data in language (Radford et al., 2018), genetic sequences (Rives et al., 2021), medical records (Steinberg et al., 2021) and more. Modeling interventions using such architectures can be applied to tasks like off-policy evaluation and policy optimization. Consider modeling medical records and observing the data of a patient suffering from heart failure. A cardiologist may wish to assess the risk of worsening conditions, e.g. experiencing decompensation, or side-effects, under different treatment policies/regimens. Such causal inference questions are important in medicine (Joshi et al., 2025; van Amsterdam et al., 2022; Chen et al., 2021), and generating potential trajectories under regimens of interest is an attractive way to answer them.

In many scenarios, the timing of treatment is a crucial aspect of treatment regimens, or *policies*. For example, we would like to know how the trajectory would change if we treated a patient a month later, or perhaps a week earlier than planned. More elaborate questions may include more complex policies like “What if in the future we will treat the patient whenever cholesterol levels are above a certain threshold?”. However, we claim here that common autoregressive models of the observational distribution, e.g. transformers trained for next event/token prediction, do not naturally lend themselves to sampling from interventional distributions that modify timing of a specific treatment because these next event prediction approaches explicitly model the time of *any* next event. In this work, we take a step towards estimating such interventions by augmenting autoregressive models to predict potential event times, as explained below. Our contributions are as follows.

- We develop competing event model (CEMs), autoregressive models where each event has a type, e.g. a treatment or observation event. To determine the type of the next event, a CEM samples a potential next event time for each type, then assigns the one that received the minimal time among the possible types. We show that such a procedure samples correctly from the observational distribution while allowing simple interventions on treatments, (e.g. what drug and dosage to give) and their timings (when to administer it). We specify conditions when these interventions produce correct causal estimates.
- To learn CEMs from observed data, we provide a maximum likelihood method and implement it with a decoder-only transformer architecture. We demonstrate the method in an off-policy evaluation problem for a tumor growth simulator.

In Section 2, we review the formalism of intervening on temporal point processes and define the problem. The discussion in this section leads us to develop CEMs and review related methods in Section 3. We then experiment with their implementation on a tumor growth simulation in Section 4.

2. Modeling Interventions in Point Processes

The problem setting, described below, follows a notation close to that in Upadhyay et al. (2018); Wald et al. (2025) and identifiability conditions from Røysland et al. (2022). In Section 2.2 we discuss autoregressive models of the observational distribution, such as transformers trained for next token prediction, and how we may use them to sample from interventional distributions.

. **AUTHORERR: Missing \icmlcorrespondingauthor.**

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

2.1. Problem Setting

Data are generated by marked point processes with an underlying multivariate counting process $\{N^e\}_{e \in \mathcal{E}}$ on the time interval $[0, T]$, where the set $\mathcal{E}$ denotes the possible types of events. We further assume some standard technical conditions that $N^e(t)$ is almost surely finite for any $t \in [0, T]$ and the filtration is the σ-algebra generated by random variables $N^e(t)$ and their marks, see Aalen et al. (2008) for an introduction. The random variables for marks are denoted by Z, and a trajectory sampled from the process is denoted by $\mathcal{H} = \{(t_0, e_0, \mathbf{z}_0), (t_1, e_1, \mathbf{z}_1), \dots, (t_n, e_n, \mathbf{z}_n)\}$, where $\mathcal{H}_t = \{(t_k, e_k, \mathbf{z}_k) \in \mathcal{H} : t_k \leq t\}$ are events up to time t.

Definition 2.1. P_{obs} is a multivariate marked point process, supported on the time interval $[0, T]$ for some $T > 0$, with observed components $\{N^e\}_{e \in \mathcal{E}}$ where $\mathcal{E}$ is a finite set. We also allow unobserved components $\{N^u\}_{u \in \mathcal{U}}$, whose jumps $\mathcal{H}^u$ are omitted from the trajectories $\mathcal{H}$. We assume intensities $\lambda_{\text{obs}}(t \mid \mathcal{H}_t, \mathcal{H}_t^u) = \mathbb{E}[dN(t) \mid \mathcal{H}_t, \mathcal{H}_t^u]$ exist, and let the marks take on values in space $\mathcal{Z}$. For $\mathcal{H}_t \in \text{supp}(P_{\text{obs}})$, we denote the density functions of the next event time and type by $P_{\text{obs}}(T_{\text{next}}, E_{\text{next}} \mid \mathcal{H}_t)$, and the mark conditioned on these by $P_{\text{obs}}(Z_{\text{next}} | \mathcal{H}_t, T_{\text{next}} = t_{\text{next}}, E_{\text{next}} = e_{\text{next}})$.

For example, when modeling a trajectory of events in a medical record, each type $e \in \mathcal{E}$ may correspond to an ICD code, and Z may represent additional details of the event. If the type e corresponds to the administration of a drug, then the respective $\mathbf{z}$ can encode additional details such as the dose or more specific features of the drug. An intervention will replace one or more of the intensity functions and mark distributions, while keeping all others fixed, *the unobserved components included*. For simplicity, we focus on interventions on one intensity process. The technical novelty in our solution regards such interventions, and generalizing to interventions on several components or mark distributions is straightforward.

Definition 2.2. An intervention on the timing of $e \in \mathcal{E}$ replaces $\lambda^e_{\text{obs}}(t \mid \mathcal{H}_t, \mathcal{H}_t^u)$ with an intensity $\lambda^e(t \mid \mathcal{H}_t)$ while keeping all components of P_{obs} fixed. We denote the resulting interventional distribution by P.

Having defined the interventions of interest, we next specify the conditions on the unobserved processes that we require for identifiability. They are a special case of the eliminability assumption formalized in Røysland et al. (2022). These are rather strong assumption and relaxing them is an interesting research question, yet they are commonly used (see (Røysland, 2011; Schulam & Saria, 2017; Vanderschueren et al., 2023; Hess & Feuerriegel, 2025; Wald et al., 2025) for somewhat similar assumptions in related settings) to guarantee identifiability of P from P_{obs}.

Lemma 2.3 (corolloary of Røysland et al. (2022))**.** *Let P_{obs} an observed distribution and consider an intervention on the timing of $e \in \mathcal{E}$. Assume that the unobserved processes can be divided into $\lambda^u = [\lambda^{u_1}, \lambda^{u_2}]$ such that the following two conditions hold: (i) $\lambda^e(t \mid \mathcal{H}_t, \mathcal{H}_t^u) = \lambda^e(t \mid \mathcal{H}_t, \mathcal{H}_t^{u_2})$, and (ii) $\lambda^{\mathcal{E}\setminus e}(t \mid \mathcal{H}_t, \mathcal{H}_t^u) = \lambda^{\mathcal{E}\setminus e}(t \mid \mathcal{H}_t)$ and $\lambda^{u_2}(t \mid \mathcal{H}_t, \mathcal{H}_t^u) = \lambda^{u_1}(t \mid \mathcal{H}_t)$. Furthermore, assume that $\lambda^e \ll \lambda^e_{\text{obs}}$, where $\ll$ denotes absolute continuity. Let $P_{\text{obs}}|_{\mathcal{E}}$ be the marginal distribution over observed variables. The interventional distribution obtained by intervening on timing of $e \in \mathcal{E}$ in $P_{\text{obs}}|_{\mathcal{E}}$ is P as defined in Definition 2.2.*

To conclude this problem introduction, we summarize the intuition behind the formal statement above. Since the learner does not observe the events $\mathcal{H}^u$, the training set $\{\mathcal{H}_i\}_{i=1}^m$ it recevies is a sample from $P_{\text{obs}}|_{\mathcal{E}}$. Therefore, a distribution that can be calculated from the training data is the one obtained by plugging λ_e into $P_{\text{obs}}|_{\mathcal{E}}$. The distribution we would like to estimate is the one where λ_e is plugged into P_{obs} (i.e. where unobserved processes are not ignored), and then marginal over the observed variables is taken. Assumptions (i) and (ii) intuitively mean that the past of an unobserved process can only affect the future of λ^e or $\lambda^{\mathcal{E}\setminus e}$, but not both (i.e. unobserved histories do not confound future events). These conditions ensure that the distribution we can calculate coincides with the desired one. This assumption is summarized graphically by the local independence graph (Didelez, 2008) in Figure 1. The overlap assumption in (iii) ensures that plugging λ^e into a model of P_{obs} does not result in trajectories that go out-of-support w.r.t P_{obs}.

$$\begin{tikzcd} U_2 \arrow[r] & N^e \arrow[r, bend left] & N^{\mathcal{E}\setminus e} \arrow[l, bend left] & U_1 \arrow[l] \end{tikzcd}$$

Figure 1: The assumed local independence graph. Intuitively, an edge $a \to b$ means the past of proccess a affects the future intensity of b.

Having formalized the estimation problem of interest, we turn to solve the algorithmic problem: how to learn autoregressive models that allow easy interventions on timings?

2.2. Interventions in Next-Event Models

Our goal in the rest of this work is to provide a simple "next-event" (or next-token) prediction architecture that supports interventions on the timing of each type $e \in \mathcal{E}$, that is, on $\lambda^e(t|\mathcal{H}_t)$, or on its mark distribution, $P(\boldsymbol{Z}_{\text{next}}|\mathcal{H}_t, T_{\text{next}}, E_{\text{next}})$. This reflects interventions that are of interest in real-world applications. Returning to our example of modeling patient trajectories, if e is a type that corresponds to prescribing a drug, then we may be able to intervene both on the timing on the event and its mark which includes details about the treatment. For other events, e.g. when e corresponds to a lab test, we may be able to control the timing, but not the mark if it holds the results of the test. Hence, separating these objects can be helpful.

Parameterizing next-event distributions. When fitting neural networks to model point processes, a crucial architectural choice is which mathematical object to fit. Common choices are to let the parameters θ of a neural network specify the intensity function $\lambda_{\text{obs}}(t \mid \mathcal{H}_t; \theta)$, or the distribution of the next jump time, $P_{\text{obs}}(T_{\text{next}} \mid \mathcal{H}_t; \theta)$. Both choices (as well as other functions) can express the joint distribution, and there is a one-to-one correspondence between them (Shchur et al., 2021). We focus on the latter, as it has some attractive properties, and it is a common choice in recent transformer-based models (McDermott et al., 2023; Song et al., 2024; Steinberg et al., 2023; Yang et al., 2023; Labach et al., 2023; Hill et al., 2023; Pang et al., 2024; Renc et al., 2024). For instance, Renc et al. (2024) train a next-token prediction model that has tokens for medical events in the MIMIC dataset and additional tokens corresponding to times between events (1-2 hours, 2-6 hours etc.). Mark distributions are also parameterized $P_\theta(\boldsymbol{Z}_{\text{next}} | \mathcal{H}_t, T_{\text{next}}, E_{\text{next}}; \theta)$.

Challenges in intervening on $P(T_{\text{next}} \mid \mathcal{H}_t; \theta)$. Turning to interventions on timings of $e \in \mathcal{E}$, which going forward *we refer to as the treatment*, we note below that obtaining a sample from the interventional $P(T_{\text{next}} \mid \mathcal{H}_t)$ by sampling $P_{\text{obs}}(T_{\text{next}} \mid \mathcal{H}_t)$ requires a non-trivial transformation. Consider two interventions of interest:

1. We are given data from a population different than the one observed in P_{obs}. We would like plug-in the treatment timing policy of that population into our model of P_{obs}.

2. We would like to set the next treatment at some fixed time $t + \delta$, or prevent it from happening altogether.

The crux of the challenge in performing these interventions on a model of $P(T_{\text{next}} \mid \mathcal{H}_t; \theta)$ is that T_{next} is not the next event time of a specific type, but the next event time of *any* type. For the first intervention, if we are given data or a closed form intensity $\lambda^e(t \mid \mathcal{H}_t)$, there is no set of parameters in $P(T_{\text{next}} \mid \mathcal{H}_t; \theta)$ that controls treatment timing alone and can be tuned by the provided target population data. Nor there are outputs where we can plug-in a given intensity λ^e. The second type of intervention is challenging as well. Say that we draw $t_{\text{next}} \sim P_{\text{obs}}(T_{\text{next}} \mid \mathcal{H}_t; \theta)$, where $t < t + \delta$, and our model of P_{obs} assigns a probability of 0.3 for the event being a treatment (i.e. type e). To intervene, we will somehow need to set this probability to 0. If we do that, then how should we distribute the probability between other types? Also, the intervention should affect the distribution of T_{next}, how should we calculate that? While the desired quantity is a functional of P_{obs} and in principle can be calculated based on our model, the required procedure would far exceed simple uses of autoregressive models, like sampling or directly intervening on next tokens. This motivates us to define CEMs and their training procedure in what follows.

3. Competing Event Models

Recall that our goal is to learn a next-token prediction model where replacing the intensity λ^e_{obs} is straightforward. To this end, the key is to introduce distributions $p_{\tilde{e}}(T_{\text{next}} \mid \mathcal{H}_t; \theta)$ for all $\tilde{e} \in \mathcal{E}$, which are the distributions of the next event of type $\tilde{e}$, conditioned on no other event occurring beforehand. These objects are common in the survival analysis and competing risks literature (Crowder, 2012; Kalbfleisch & Prentice, 2002; Tsiatis, 1975; Lee et al., 2018), where we wish to fit a distribution for the survival time of a patient, but in the observed data some patients are lost to censoring. That is, they drop out of the experiment after a certain time. Identification of the correct survival distribution is possible under similar conditions to the ones laid out in Lemma 2.3. Competing Event Models takes inspiration from this literature and extends the principle to sequence data, and interventions on timing. Let us formalize this.

Definition 3.1. Let P_{obs} a multivariate marked process with intensities $\{\lambda^{\tilde{e}}_{\text{obs}}\}_{\tilde{e} \in \mathcal{E}}$, $e \in \mathcal{E}$ one type of event in the process, and $\mathcal{H}_t$ events up to time t. Denoting by $\Delta N(t, t+\delta) = 0$ the event where no jumps occur in the interval $(t, t+\delta)$, we define the following intensity for any $\delta \in (0, T - t]$,

$$\tilde{\lambda}^e_{\text{obs}}(t + \delta | \mathcal{H}_t) = \lambda^e_{\text{obs}}(t + \delta | \mathcal{H}_t, \Delta N(t, t+\delta) = 0).$$

When considering this intensity, we will denote the random variable for the first jump time after t by T_e, the probability density for this jump time by $\tilde{p}_e(T_e | \mathcal{H}_t)$, and the CDF of $\tilde{p}_e(T_e | \mathcal{H}_t)$ by $F_e[t + \delta | \mathcal{H}_t]$.

As suggested in the definition, there is a one-to-one correspondence between $\tilde{p}_e$ and λ^e_{obs}, hence we will implement treatment timing interventions by changing $\tilde{p}_e$ while keeping $\tilde{p}_{\tilde{e}}$ fixed for all $\tilde{e} \in \mathcal{E} \setminus e$. The key components that remain to be specified, is how to form a joint distribution from $\{\tilde{p}_{\tilde{e}}\}_{\tilde{e} \in \mathcal{E}}$ and sample from it; how to fit the distributions from data; and how to perform interventions.

The lemma below (see Appendix A for proof) suggests sampling times independently from all the $\tilde{p}_{\tilde{e}}$ and setting the next event type and time according to the earliest one. It also gives the expression for the likelihood, from which we derive the training algorithm. The sampling and training procedures are summarized in Algorithm 1, including the loss to fit mark distributions.

Lemma 3.2. *Let* $\mathcal{H}_t \in \text{supp}(P_{\text{obs}})$*, consider the random variable* $T_{\text{next}} = \min_{\tilde{e} \in \mathcal{E}} T_{\tilde{e}}$ *and* e *the minimizing argument, where* $\tilde{p}(\widetilde{T}_1, \ldots, \widetilde{T}_{|\mathcal{E}|} | \mathcal{H}_t) = \prod_{\tilde{e} \in \mathcal{E}} \tilde{p}_{\tilde{e}}(\widetilde{T}_{\tilde{e}} \mid \mathcal{H}_t)$*. Then* $\tilde{p}(T_{\text{next}} = \tilde{t}, E_{\text{next}} = e | \mathcal{H}_t) = p_{\text{obs}}(T_{\text{next}} = \tilde{t}, E_{\text{next}} = e | \mathcal{H}_t)$*. The likelihood of an observed trajectory* $\mathcal{H} = \{(t_j, e_j, \mathbf{z}_j)\}_{j=1}^n$ *under this model is given by*

$$\sum_{j \in [n]} \left(\log \tilde{p}_{e_j}\left(t_j | \mathcal{H}_{t_{j-1}}\right) + \sum_{e \neq e_j} \log\left(1 - F_e[t_j | \mathcal{H}_{t_{j-1}}]\right) \right)$$

Algorithm 1 CEMs Learning and Sampling

```
Training $p(\mathcal{H};\theta)$
Input: Trajectories $\{\mathcal{H}_i\}_{i=1}^N$, learning rate $\eta$
Initialize $\boldsymbol{\theta}$ randomly
for $D$ rounds do
    Draw batch $B$ and for each $\mathcal{H}_i \in B$ calculate
    $$\begin{aligned}\mathcal{L}(\mathcal{H}_i;\theta) \leftarrow &\sum_{(t_j,e_j,\mathbf{z}_j)\in\mathcal{H}_i} \log p_{e_j}(t_j|\mathcal{H}_{t_{j-1}};\theta)\\ &+\sum_{e\neq e_i}\log(1-F_e[t_j|\mathcal{H}_{t_{j-1}};\theta])\\ &+\log p(\mathbf{z}|\mathcal{H}_{t_{j-1}},T_j=t_j,E_j=e_j;\theta)\end{aligned}$$
    Update $\theta \leftarrow \theta + \eta|B|^{-1}\sum_{\mathcal{H}_i\in B}\nabla_\theta\mathcal{L}(\mathcal{H}_i;\theta)$
end for
Return $p(\cdot;\theta)$

Sampling $p(\cdot|\mathcal{H}_t;\boldsymbol{\theta})$ with intervention $\tilde{p}_{\tilde{e}}(T_{\text{next}} \mid \mathcal{H}_t)$
Sample $T_e \sim p_e(T_{\text{next}} \mid \mathcal{H}_t;\boldsymbol{\theta})$ for $e \neq \tilde{e}$
Sample $T_{\tilde{e}} \sim \tilde{p}_{\tilde{e}}(T_{\text{next}} \mid \mathcal{H}_t)$
Set $e_{\text{next}} \leftarrow \arg\min_e T_e, t_{\text{next}} \leftarrow T_{e_{\text{next}}}$
Sample $\mathbf{z}_{\text{next}} \sim p(\mathbf{z} \mid \mathcal{H}_t, T_{\text{next}} = t_{\text{next}}, E_{\text{next}} = e_{\text{next}})$
Return $\mathcal{H}_t \cup (t_{\text{next}}, e_{\text{next}}, \mathbf{z})$
```

Implementation considerations and interventions. The rather flexible formulation of CEMs allows various implementation choices. The timeline may be discretized, where events are sparse and occur irregularly, or it may be continuous and parameterized by, e.g., a mixture density network (Bishop, 1994). There is also flexibility in choosing what to consider as types of events $\mathcal{E}$ and what as marks $\mathcal{Z}$, according to the type of interventions we consider. The training objective maximizes the likelihood of the time t_j for the type e_j of the j-th event, while maximizing the probability of all times after t_j for the types $e \neq e_j$. In our implementation the timeline is discretized, so calculating $F_e[t_i \mid \mathcal{H}_{t_i-1}]$ is simple. Yet if one uses more complex functions that are difficult to integrate, the following identity (see Appendix A for derivation) can be useful to obtain an unbiased stochastic estimator of the gradient:

$$\nabla_\theta \log 1 - F_e[t_i \mid \mathcal{H}_{t_{i-1}};\theta] = \mathop{\mathbb{E}}_{t\sim p_e(\cdot|T_e>t_i;\theta)}\left[\nabla_\theta \log p_e(t|\mathcal{H}_{t_{i-1}};\theta)\right]$$

The resulting estimator is rather intuitive: Sample a next time for $e \neq e_i$ from the estimate of $\tilde{p}_e$, conditioned this time is later than t, and maximize the likelihood of this time.

Finally, we discuss how to apply interventions on a fitted model $P_{\text{obs}}(\mathcal{H};\theta)$, like those suggested in Section 2.2. Manually setting the potential time T_e to some value, like $t+\delta$, and following the min-time sampling procedure, is a simple way to estimate queries such as "what if we treat at time $t+\delta$". To finetune according to a reward/loss function, we may tune parameters θ_e that only affect the distribution $\tilde{p}_e(\cdot;\theta)$ (in our implementation, there is a shared representation, and a separate linear prediction head for each $\tilde{e} \in \mathcal{E}$). Replacing $\tilde{p}_e(\cdot;\theta)$ with a closed-form alternative is also straightforward, and we do that in our experiment next.

$(\gamma,\beta)_{\text{obs}}$	ERM / MC	FQE	EDQ	CEM
		$(\gamma,\beta)_{\text{int}} = (6, 0.75)$		
$(6, 0.75)$	0.0325 ± 0.0002	0.0461 ± 0.0006	0.0373 ± 0.0007	0.0305 ± 0.0044
$(10, 0.5)$	0.063 ± 0.0026	0.0750 ± 0.0210	0.050 ± 0.0029	0.0345 ± 0.0026

Figure 2: NRMSE for tumor growth simulation. Top row evaluates prediction on P_{obs}, the bottom evaluates prediction under an intervention where treatment distribution $(\gamma,\beta)_{\text{obs}} = (10, 0.5)$ is replaced by $(\gamma,\beta)_{\text{int}} = (6, 0.75)$.

4. Experiments

As a proof-of-concept for CEMs, we use a pharmacokinetic model for tumor growth. This setting from Bica et al. (2020) was adapted to study the irregular time sampling we are interested in (Seedat et al., 2022; Vanderschueren et al., 2023; Wald et al., 2025). We defer most details on the simulator to Appendix B and briefly describe the task here. The simulator works in discrete time $t \in [T]$ where $T = 20$, and irregular event times are induced by the feature being unobserved at most times. We have $\mathcal{E} = \{\text{x}, \text{a}\}$, where the mark $z_t \in \mathbb{R}_+$ for x is the tumor volume and is observed with probability $\sigma((\bar{z}_{t-d:t-1}/d_{\max}) - 1.5)$, where $\bar{z}_{t-d:t-1}$ is the average tumor volume over the last d timesteps, and $d_{\max}$ is the maximum considered volume. For a the mark denotes type of treatment: radiotherapy, chemotherapy, or combined therapy. We intervene on the treatment assignment mechanism, determined by two parameters (γ, β), where each type of treatment is assigned with probability $\sigma(\gamma(x_{\text{last}} - \beta) + t - t_{\text{last}})$. Here, x_{last} is the last observed volume and t_{last} is the time of the last treatment. β controls how often treatments are applied, while γ controls the dependence of treatment assignment on tumor volume.

We implement CEMs by modifying the GPT-2 architecture, and also use this to implement baselines: ERM/MC that predicts the tumor growth from observed data, disregarding the target treatment policy, and two off-policy evaluation methods based on dynamic programming, FQE (Le et al., 2019) and EDQ (Wald et al., 2025). We relegate additional details on the simulator, baselines/related literature and architecture to Appendix B. To evaluate the methods, we use them to predict the tumor volume at time $T = 20$, and report the normalized mean squared error in Figure 2. CEMs outperform the baselines on prediction under the intervention, we further expand our analysis of the results in Appendix B.

References

Aalen, O., Borgan, O., and Gjessing, H. *Survival and event history analysis: a process point of view*. Springer Science & Business Media, 2008.

Bica, I., Alaa, A. M., Jordon, J., and van der Schaar, M. Estimating counterfactual treatment outcomes over time through adversarially balanced representations. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJg866NFvB.

Bishop, C. M. Mixture density networks. 1994.

Chen, I. Y., Joshi, S., Ghassemi, M., and Ranganath, R. Probabilistic machine learning for healthcare. *Annual review of biomedical data science*, 4(1):393–415, 2021.

Crowder, M. J. *Multivariate survival analysis and competing risks*. CRC Press, 2012.

Didelez, V. Graphical models for marked point processes based on local independence. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(1): 245–264, 2008.

Geng, C., Paganetti, H., and Grassberger, C. Prediction of treatment response for combined chemo-and radiation therapy for non-small cell lung cancer patients using a bio-mathematical model. *Scientific reports*, 7(1):13542, 2017.

Hess, K. and Feuerriegel, S. Stabilized neural prediction of potential outcomes in continuous time. In *The Thirteenth International Conference on Learning Representations*, 2025.

Hill, B. L., Emami, M., Nori, V. S., Cordova-Palomera, A., Tillman, R. E., and Halperin, E. Chiron: A generative foundation model for structured sequential medical data. In *Deep Generative Models for Health Workshop NeurIPS 2023*, 2023.

Joshi, S., Urteaga, I., van Amsterdam, W. A., Hripcsak, G., Elias, P., Recht, B., Elhadad, N., Fackler, J., Sendak, M. P., Wiens, J., et al. Ai as an intervention: improving clinical outcomes relies on a causal approach to ai development and validation. *Journal of the American Medical Informatics Association*, pp. ocae301, 2025.

Kalbfleisch, J. D. and Prentice, R. L. *The statistical analysis of failure time data*. John Wiley & Sons, 2002.

Labach, A., Pokhrel, A., Huang, X. S., Zuberi, S., Yi, S. E., Volkovs, M., Poutanen, T., and Krishnan, R. G. Duett: dual event time transformer for electronic health records. In *Machine Learning for Healthcare Conference*, pp. 403–422. PMLR, 2023.

Le, H., Voloshin, C., and Yue, Y. Batch policy learning under constraints. In *International Conference on Machine Learning*, pp. 3703–3712. PMLR, 2019.

Lee, C., Zame, W., Yoon, J., and Van Der Schaar, M. Deephit: A deep learning approach to survival analysis with competing risks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Li, M. M., Li, K., Ektefaie, Y., Messica, S., and Zitnik, M. Controllable sequence editing for counterfactual generation. *arXiv preprint arXiv:2502.03569*, 2025.

McDermott, M., Nestor, B., Argaw, P., and Kohane, I. S. Event stream gpt: a data pre-processing and modeling library for generative, pre-trained transformers over continuous-time sequences of complex events. *Advances in Neural Information Processing Systems*, 36:24322–24334, 2023.

Melnychuk, V., Frauen, D., and Feuerriegel, S. Causal transformer for estimating counterfactual outcomes. In *International Conference on Machine Learning*, pp. 15293–15329. PMLR, 2022.

Pang, C., Jiang, X., Pavinkurve, N. P., Kalluri, K. S., Minto, E. L., Patterson, J., Zhang, L., Hripcsak, G., Gürsoy, G., Elhadad, N., et al. Cehr-gpt: Generating electronic health records with chronological patient timelines. *arXiv preprint arXiv:2402.04400*, 2024.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Renc, P., Jia, Y., Samir, A. E., Was, J., Li, Q., Bates, D. W., and Sitek, A. Zero shot health trajectory prediction using transformer. *NPJ Digital Medicine*, 7(1):256, 2024.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

Røysland, K. A martingale approach to continuous-time marginal structural models. *Bernoulli*, 17(3):895 – 915, 2011.

Røysland, K., Ryalen, P., Nygaard, M., and Didelez, V. Graphical criteria for the identification of marginal causal effects in continuous-time survival and event-history analyses. *arXiv preprint arXiv:2202.02311*, 2022.

Schulam, P. and Saria, S. Reliable decision support using counterfactual models. *Advances in neural information processing systems*, 30, 2017.

Seedat, N., Imrie, F., Bellot, A., Qian, Z., and van der Schaar, M. Continuous-time modeling of counterfactual outcomes using neural controlled differential equations. *arXiv preprint arXiv:2206.08311*, 2022.

Shchur, O., Türkmen, A. C., Januschowski, T., and Günnemann, S. Neural temporal point processes: A review. *arXiv preprint arXiv:2104.03528*, 2021.

Song, Z., Lu, Q., Zhu, H., Buckeridge, D., and Li, Y. Trajgpt: Irregular time-series representation learning for health trajectory analysis. *arXiv preprint arXiv:2410.02133*, 2024.

Steinberg, E., Jung, K., Fries, J. A., Corbin, C. K., Pfohl, S. R., and Shah, N. H. Language models are an effective representation learning technique for electronic health record data. *Journal of biomedical informatics*, 113: 103637, 2021.

Steinberg, E., Fries, J., Xu, Y., and Shah, N. Motor: a time-to-event foundation model for structured medical records. *arXiv preprint arXiv:2301.03150*, 2023.

Tsiatis, A. A nonidentifiability aspect of the problem of competing risks. *Proceedings of the National Academy of Sciences*, 72(1):20–22, 1975.

Upadhyay, U., De, A., and Gomez Rodriguez, M. Deep reinforcement learning of marked temporal point processes. *Advances in neural information processing systems*, 31, 2018.

van Amsterdam, W. A., de Jong, P. A., Verhoeff, J. J., Leiner, T., and Ranganath, R. Decision making in cancer: Causal questions require causal answers. *arXiv preprint arXiv:2209.07397*, 2022.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Vanderschueren, T., Curth, A., Verbeke, W., and van der Schaar, M. Accounting for informative sampling when learning to forecast treatment outcomes over time. *arXiv preprint arXiv:2306.04255*, 2023.

Wald, Y., Goldstein, M., Efroni, Y., van Amsterdam, W. A., and Ranganath, R. Time after time: Scalable effect estimation for interventions on when and what to do. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yang, Z., Mitra, A., Liu, W., Berlowitz, D., and Yu, H. Transformehr: transformer-based encoder-decoder generative model to enhance prediction of disease outcomes using electronic health records. *Nature communications*, 14(1):7857, 2023.

A. Proof and Additional Details

Let us restate and prove the statement on the likelihood and sampling procedure of CEMs.

Lemma A.1. *Let $\mathcal{H}_t \in \mathrm{supp}(P_{\text{obs}})$, consider the random variable $T_{\text{next}} = \min_{\tilde{e}\in\mathcal{E}} T_{\tilde{e}}$ and e the minimizing argument, where $\tilde{p}(\widetilde{T}_1, \dots, \widetilde{T}_{|\mathcal{E}|}|\mathcal{H}_t) = \prod_{\tilde{e}\in\mathcal{E}} \tilde{p}_{\tilde{e}}(\widetilde{T}_{\tilde{e}} \mid \mathcal{H}_t)$. Then $\tilde{p}(T_{\text{next}} = \tilde{t}, E_{\text{next}} = e|\mathcal{H}_t) = p_{\text{obs}}(T_{\text{next}} = \tilde{t}, E_{\text{next}} = e|\mathcal{H}_t)$. The likelihood of an observed trajectory $\mathcal{H} = \{(t_j, e_j, \mathbf{z}_j)\}_{j=1}^n$ under this model is given by*

$$\sum_{j\in[n]} \left(\log \tilde{p}_{e_j}\left(t_j|\mathcal{H}_{t_{j-1}}\right) + \sum_{e\neq e_j} \log\left(1 - F_e[t_j|\mathcal{H}_{t_{j-1}}]\right) \right)$$

Proof. Let us show that the log-likelihood of a trajectory $\mathcal{H}$, when written in terms of the next jump-time distributions $\{\tilde{p}_j\}_{j=1}^k$ is

$$\sum_{j\in[n]} \left(\log \tilde{p}_{e_j}\left(t_j|\mathcal{H}_{t_{j-1}}\right) + \sum_{e\neq e_i} \log\left(1 - F_e[t_j|\mathcal{H}_{t_{j-1}}]\right) \right)$$

To show this we rewrite the terms above using intensities λ instead of next time densities, and observe that it coincides with the familiar log-likelihood of $\mathcal{H}$ from the point processes literature (Aalen et al., 2008),

$$\sum_{j\in[n]} \log \lambda^{e_j}(t_j|\mathcal{H}_{t_j-}) - \int_0^T \sum_{e\in\mathcal{E}} \lambda^e(t|\mathcal{H}_{t-})dt.$$

Here $\mathcal{H}_{t-}$ is the history of events up until time t, but not including time t. The equivalence follows easily by observing that in trajectory $\mathcal{H}$ the process does not jump at any $t \in (t_{j-1}, t_j)$, so it holds that $\lambda(t|\mathcal{H}_t) = \lambda(t|\mathcal{H}_{t_{j-1}}, \Delta N(t_{j-1}, t) = 0)$. Hence we get

$$\begin{aligned}
\tilde{p}_e(t_j|\mathcal{H}_{t_{j-1}}) &= \tilde{\lambda}^e(t_j|\mathcal{H}_{t_{j-1}}) \exp\left\{-\int_{t_{j-1}}^{t_j} \tilde{\lambda}^e(t|\mathcal{H}_{t_{j-1}})dt\right\} \\
&= \lambda^e(t_j|\mathcal{H}_{t_{j-1}}, \Delta N(t_{j-1}, t_j) = 0) \exp\left\{-\int_{t_{j-1}}^{t_j} \lambda^e(t|\mathcal{H}_{t_{j-1}}, \Delta N(t_j, t) = 0)dt\right\} \\
&= \lambda^e(t_i|\mathcal{H}_{t_j-}) \exp\left\{-\int_{t_{j-1}}^{t_j} \lambda^e(t|\mathcal{H}_{t-})dt\right\} \\
F_e[t_j|\mathcal{H}_{t_{j-1}}] &= 1 - \exp\left\{-\int_{t_{j-1}}^{t_j} \tilde{\lambda}^e(t|\mathcal{H}_{t_{j-1}})dt\right\} \\
&= 1 - \exp\left\{-\int_{t_{j-1}}^{t_j} \lambda^e(t|\mathcal{H}_{t_{j-1}}, \Delta N(t_j, t) = 0)dt\right\} \\
&= 1 - \exp\left\{-\int_{t_{j-1}}^{t_j} \lambda^e(t|\mathcal{H}_{t-})dt\right\}
\end{aligned}$$

Putting these together, we obtain

$$\begin{aligned}
&\sum_{j\in[n]} \left(\log \tilde{p}_{e_j}(t_j|\mathcal{H}_{t_{j-1}}) + \sum_{e\neq e_i} \log(1 - F_e[t_j|\mathcal{H}_{t_{j-1}}]) \right) \\
&= \sum_{j\in[n]} \log \lambda^{e_i}(t_j|\mathcal{H}_{t_j-}) - \int_{t_{j-1}}^{t_j} \lambda^{e_i}(t|\mathcal{H}_{t-})dt - \sum_{e\neq e_j} \int_{t_{j-1}}^{t_j} \lambda^e(t|\mathcal{H}_t-) \\
&= \sum_{j\in[n]} \log \lambda^{e_i}(t_j|\mathcal{H}_{t_j-}) - \int_0^T \sum_{e\in\mathcal{E}} \lambda^e(t|\mathcal{H}_{t-})dt.
\end{aligned}$$

This indeed coincides with the likelihood in terms of $\lambda^e(t \mid \mathcal{H}_t-)$, which concludes the proof of the claim on the log-likelihood.

Next, to show that min-time sampling returns a correct sample from P_{obs}, we again show that the next-time density coincides with the familiar expression written with densities,

$$P_{\text{obs}}(T_{\text{next}} = \tilde{t}, E = e \mid \mathcal{H}_t) = \lambda^e(\tilde{t} \mid \mathcal{H}_{\tilde{t}-}) \exp\left(-\int_t^{\tilde{t}} \sum_{\tilde{e}\in\mathcal{E}} \lambda^{\tilde{e}}(s|\mathcal{H}_{s-})ds\right).$$

We start by spelling our $\mathcal{H}_{s-}$ using the event $\Delta N(t,s) = 0$, as the expression above conditions on events where there are no jumps in the interval $(t,\tilde{t})$. We begin by rewriting the probabilities of a next event time and type, under min-time sampling, in terms of densities and CDFs. The first equality in the following holds due to independence, while in the second we switch to writing with the intensities $\tilde{\lambda}$ we defined, then we combine the exponents, followed by spelling out the conditioning sets and switching $\tilde{\lambda}$ to λ, and finally we note that the observed history $\mathcal{H}_t$ coincides with the events $\Delta N(t,s) = 0$ for $s \leq \tilde{t}$ since no jumps occur in this interval.

$$\begin{aligned}
\tilde{p}(T_{\text{next}} = \tilde{t}, E_{\text{next}} = e \mid \mathcal{H}_t) &= \tilde{p}_e(\widetilde{T}_e = \tilde{t} \mid \mathcal{H}_t) \prod_{\tilde{e}\in\mathcal{E}\setminus e} 1 - F_{\tilde{e}}[\tilde{t} \mid \mathcal{H}_t] \\
&= \tilde{\lambda}^e(\tilde{t} \mid \mathcal{H}_t) \exp\left(-\int_t^{\tilde{t}} \tilde{\lambda}^e(s \mid \mathcal{H}_t)ds\right) \prod_{\tilde{e}\in\mathcal{E}\setminus e} \exp\left(-\int_t^{\tilde{t}} \tilde{\lambda}^{\tilde{e}}(s \mid \mathcal{H}_t)ds\right) \\
&= \tilde{\lambda}^e(\tilde{t} \mid \mathcal{H}_t)\exp\left(-\sum_{\tilde{e}\in\mathcal{E}} \int_t^{\tilde{t}} \tilde{\lambda}^{\tilde{e}}(s \mid \mathcal{H}_t)ds\right) \\
&= \lambda^e(\tilde{t} \mid \mathcal{H}_t, \Delta N(t,\tilde{t}) = 0)\exp\left(-\sum_{\tilde{e}\in\mathcal{E}} \int_t^{\tilde{t}} \lambda^{\tilde{e}}(s \mid \mathcal{H}_t, \Delta N(t,s) = 0)ds\right) \\
&= \lambda^e(\tilde{t} \mid \mathcal{H}_{\tilde{t}-})\exp\left(-\sum_{\tilde{e}\in\mathcal{E}} \int_t^{\tilde{t}} \lambda^{\tilde{e}}(s \mid \mathcal{H}_{s-})ds\right).
\end{aligned}$$

This concludes the proof as the right hand side coincides with $P_{\text{obs}}(T_{\text{next}} = \tilde{t}, E_{\text{next}} = e \mid \mathcal{H}_t)$, so the desired identity is obtained. $\square$

For completeness, we also spell out the derivation of the identity from the main text, from which we may form a stochastic estimator of the gradient for the CDF,

$$\begin{aligned}
\nabla_\theta \log 1 - F_e[t_i \mid \mathcal{H}_{t_{i-1}};\theta] =& \nabla_\theta \log \int_{t_i}^T \tilde{p}_e(T_{\text{next}} = t \mid \mathcal{H}_{t_{i-1}};\theta) = \\
& \frac{1}{\int_{t_i}^T \tilde{p}_e(T_{\text{next}} = t \mid \mathcal{H}_{t_{t-1}};\theta)dt} \int_{t_i}^T \nabla_\theta \tilde{p}_e(T_{\text{next}} = t' \mid \mathcal{H}_{t_{i-1}};\theta)dt' = \\
& \int_{t_i}^T \frac{\tilde{p}_e(T_{\text{next}} = t' \mid \mathcal{H}_{t_{i-1}};\theta)}{\int_{t_i}^T \tilde{p}_e(T_{\text{next}} = t \mid \mathcal{H}_{t_{t-1}};\theta)dt} \nabla_\theta \log \tilde{p}_e(T_{\text{next}} = t' \mid \mathcal{H}_{t_{i-1}};\theta)dt' = \\
& \mathop{\mathbb{E}}_{t'\sim p_e(\cdot|T_e>t_i;\theta)} \left[\nabla_\theta \log p_e(t'|\mathcal{H}_{t_{i-1}};\theta)\right]
\end{aligned}$$

B. Details on Simulations and Baselines

Cancer simulator: The tumor growth simulation we use is adapted from Bica et al. (2020); Seedat et al. (2022); Melnychuk et al. (2022) and is based on the work of Geng et al. (2017). Tumor volumes $V(t)$ are simulated as finite differences from the following differential equation,

$$\frac{dV(t)}{dt} = \left(\underbrace{\rho \log\left(\frac{K}{V(t)}\right)}_{\text{Tumor growth}} - \underbrace{\beta_c C(t)}_{\text{Chemotherapy}} - \underbrace{\left(\alpha_r d(t) + \beta_r d(t)^2\right)}_{\text{Radiotherapy}} + \underbrace{e_t}_{\text{Noise}}\right) V(t).$$

Here $C(t)$ is the chemotherapy concentration, $d(t)$ represents the level of radiothearpy. $\rho, K, \beta_c, \alpha_r, \beta_r$ are effect parameters drawn for each patient from a prior distribution described in Geng et al. (2017), and $e_t \sim \mathcal{N}(0, 0.0001)$ is a noise term. To create irregularly sampled observations of the tumor volume, at each time step we draw a value from a Bernoulli distribution to decide whether the trajectory contains the tumor volume at this time step or not. The success probability is a function of the average tumor volume over the most recent 15 volumes (both observed and unobserved). If we denote a missing value by $\emptyset$ and the observation at timestep t by X_t (which equals $\emptyset$ if there is no sample at this timestep and $V(t)$ otherwise), then sampling times are drawn according to the following probabilities:

$$p(V_t \neq \emptyset | \mathcal{H}_t) = \sigma\left(\frac{\bar{V}_{t-15:t}}{V_{\max}} - 1.5\right)$$

The policies we use to decide on treatments draw binary decisions of whether or not to apply chemotherapy and radiotherapy at each timestep. Denoting these decisions by random variables C_t and d_t, they are drawn according to $P(R_t = 1|\mathcal{H}_t) = \sigma\left(\gamma \cdot (v_{\text{last}} - \beta) + t - t_{\text{last}}\right)$, where v_{last} is the last observed volume before time t and t_{last} is the last time that treatment was applied before t. The same probabilities are applied for d_t.

we conjecture that this is because they are simpler and faster to train than the dynamic programming baselines, where FQE suffers from noisy gradients due to time discretization, while EDQ is better but still requires many epochs to train

Architecture: To implement all the methods we use the GPT-2 architecture. Each token is a concatenation of embeddings of time t_i, value $\mathbf{z}_i$ and event type $e_i \in \{A, X, Y, \Delta T\}$. The event types A, X, Y correspond to actions, features, and outcomes, while ΔT is introduced to represent time differences between two events (the time difference can also be 0). For CEMs we include prediction heads for the time difference to next event, for each event type (i.e. predicting the potential time $\tilde{T}_e$). After each event token of type that is not Δ, we place a token of type Δ with a value of this drawn next time difference. Since we discretize time in our implementation, the time prediction heads output a probability distribution over possible categorical time differences and are trained with a cross-entropy loss. We also introduce a prediction head for the marks of each type except Δ, and the mark of the next event is taken as the prediction of that head in case the corresponding type is the one with the earliest time.

Baselines: We choose the baselines Monte-Carlo/Empirical Risk Minimization as this is the simplest baseline, that demonstrates the effect of distribution shift on the performance of a naïve learner. This baseline simply regresses to the observed label, and thus does not take into account that at test time we will intervene on the treatment policy.

FQE (Le et al., 2019) is a dynamic programming algorithm that is arguably the most common algorithm for policy evaluation in discrete time. Since the problem we experiment with is an off-policy evaluation problems, it is an appropriate baseline to examine the effect of using an algorithm that is not tailored for irregularly sampled times.

EDQ (Wald et al., 2025) is a recent algorithm proposed as a dynamic programing algorithm that is an alternative to FQE, and is more suitable for irregularly sampled times. Since finely discretized times may harm the propagation of information backward in time in discrete-time FQE, EDQ does dynamic programming based on the earliest disagreement time, which may be much later than oen timestep ahead, and thus enable better flow of the error backwards in time.

For both EDQ and FQE above we also include a target network (Van Hasselt et al., 2016), and update it with soft-Q updates. Empirically, we should note that it seems like FQE and EDQ are underperforming CEMs due to long training times (FQE suffers from this more than EDQ). Sppeding up the implementations of these algorithm might be possible, and they might achieve improved performace, though we leave that for future exploration.

Finally, we wish to mention several relevant works regarding causal inference on data with irregularly sampled times, as there are a few reccent works on the topic. Seedat et al. (2022) give an architecture that is more suitable for irreguarly sampled data, but does not intervene on treatment timing, Vanderschueren et al. (2023) give a reweighting technique to simulate non-informative sampling times. However, they do not learn generative models of such data, nor intervene on them, and the method cannot evaluate dynamic policies (where treatments and times are dependent on past observations). Hess & Feuerriegel (2025) give a thorough treatment and neural model for reweighting intensities to make them non-informative, and using stabilized weights to improve sample complexity. They also do not learn an intervenable generative model, and to the best of our understanding do not support dynamic policies. Finally, a recent preprint (Li et al., 2025) proposes an architecture to answer time-related counterfactual questions, yet the code is not available and the paper does not formalize a causal inference question. It is interesting to explore whether the methods and estimands we develop are connected to those in Li et al. (2025).