When to intervene: learning optimal intervention policies for critical events

Anonymous Author(s)
Affiliation
Address
email

Abstract

Providing timely interventions before the onset of a critical event, such as a system failure is an important problem in many industrial settings. Before the onset of the critical event, usually, systems exhibit behavioral changes which often manifest as stochastic co-variate observations which may be leveraged to trigger intervention. In this paper, for the first time, we formulate the problem of finding an optimally timed intervention (OTI) policy as minimizing the expected residual time to event, subject to a constraint on the probability of missing the event. Existing machine learning approaches to intervention on critical events focus on predicting event occurrence within a pre-defined window (a classification problem) or predicting time-to-event (a regression problem). Interventions are then triggered by setting model thresholds. These are heuristic-driven, lacking guarantees regarding optimality. To model the evolution of system behavior, we introduce the concept of a hazard rate process. We show that the OTI problem is equivalent to an optimal stopping problem on the associated hazard rate process. This key link has not been explored in literature. Under Markovian assumptions on the hazard rate process, we show that an OTI policy at any time is completely determined by the conditional hazard rate function at that time. We proceed to analytically characterize the optimal stopping policy, enabling practical survival-model-based critical event intervention algorithms. Further, we show that our framework includes, as a special case, the important class of neural hazard rate processes generated by recurrent neural networks (RNNs). To model neural hazard rate processes we propose a dynamic deep recurrent survival analysis (DDRSA) architecture, introducing an RNN encoder into the static DRSA setting. Finally, we demonstrate RNN-based OTI policies with practical experiments and show that they outperform popular intervention methods.

1 Introduction

We consider a problem that involves the monitoring of observations from a discrete-time stochastic process with the goal of intervening to prevent the occurrence of a critical event that will happen in the future. A critical event is missed (loosely termed a false negative) when intervention is not triggered before the critical event occurs. The general objective is the design of algorithms that can be used to trigger a timely intervention (the objective may be to maximize useful life, minimize overall domain specific costs e.t.c.), subject to false negative constraints.

This problem has significant practical significance in a number of fields. Critical events could be the failure of a machine or key component, customer churn, loan default, or even death. Intervening on such events ahead of time, in an optimal manner is of great significance to businesses and society, due to the potential of mitigating the impact of the occurrences of these events.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

1.1 Interventions on critical events

The monitored stochastic process is represented by a discrete, temporal sequence of random variables with the random variable X_j representing a d-dimensional real-valued observation at time step j. Let the set $\mathcal{X}_j = \{X_s : 1 \leq s \leq j\}$ represent the observation history until time step j. The critical event is assumed to occur at a random time-step $L < L_{max}$[1]. An intervention at time step j is considered early if the intervention occurs at a time-step $j < L$. We consider present and future interventions (at $j = L$ and beyond) too late by definition.

We consider a data-driven sequential decision rules (intervention policies) that are defined as follows:

Definition 1.1 (Intervention policies)**.** An intervention policy is a sequence of functions $\boldsymbol{\varphi} = (\varphi_1, \varphi_2, \cdots \varphi_j, \cdots)$, where $\varphi_j : \mathbb{R}^{d \times j} \mapsto \{0, 1\}$ maps the history of observations at a time step j to a decision of whether to intervene ($\varphi_j(\mathcal{X}_j) = 1$) or not ($\varphi_j(\mathcal{X}_j) = 0$).

The intervention policies we consider in this paper are terminal, i.e: once a decision to intervene is triggered, the policy stops. Note that we can also allow for policy reactivation on the same subject after a critical event. The sequence of observations starting at such policy reactivations are simply treated as independent realizations of the process. So if we recover from the occurence of a first critical event and re-activate the policy to continue to monitor for a second second critical event, these cases will be considered as independent trials.

Definition 1.2 (policy stopping time)**.** Intervention on a monitored process happens at a random time step $\tau := \min\{j \geq 1 : \varphi_j(\mathcal{X}_j) = 1\}$. We call this time the stopping time [2] of the policy.

1.2 Optimally timed intervention (OTI) on critical events

We now formally state the optimally timed intervention (OTI) problem for the first time. We seek to find intervention policy (or equivalently, the stopping time) that minimizes expected time-to-event $\mathbb{E}[(L - \tau)^+]$ subject to the constraint that the probability of missing the critical event is at most $\alpha \in (0, 1]$:

$$\tau^* = \arg \inf_{\tau \in \mathcal{M}_\alpha} \mathbb{E}[(L - \tau)^+] \text{ where: } \mathcal{M}_\alpha = \{1 \leq \tau < L_{max} : \mathbb{P}(L \leq \tau) \leq \alpha\} \tag{1}$$

The expected time-to-event, also called expected residual life or mean residual life (MRL)) is a quantity that is well studied in literature with application in multiple domains [1, 14].

Note that there exists a multiplier C_α, such that the OTI problem may be expressed in the following equivalent, unconstrained form:

$$\tau^* = \arg \inf_{\tau} \mathbb{E}[(L - \tau)^+] + C_\alpha \mathbb{P}(L \leq \tau) \text{ for some } C_\alpha > 0 \qquad \text{(OTI problem)}$$

The objective simply encodes the tradeoff between expected residual life and the probability of missing the critical event. C_α can directly be interpreted as the risk (cost) of missing the critical event. Changing C_α allows one to vary the risk tradeoff between the miss-rate and expected time-to-event. Note that when we miss the critical event i.e: $\tau \geq L$, expected residual life is zero, so $\mathbb{E}[(L - \tau)^+] = 0$.

A related objective also appears in optimal stopping formulations for Bayesian quickest change point detection problem [37], also called the Shirayaev problem [29], where the objective is to monitor a stochastic process for the quickest possible detection of a change point or stochastic disorder. Here the tradeoff is between stopping too early (proabability of a false alarm) and the expected time delay in detecting the change point, after it has occured.

1.3 Related work

Window based critical event prediction: Intervention is a binary decision. At each time-step a decision must be made to intervene or not. This is often modeled directly as a binary classification

[1] This upper bound assumption does not pose practical limitations due to our survival analysis approach. We may treat unobserved critical events as censored. See Section 4.1 for details

[2] a stopping time in probability theory is random a variable τ such that $\mathbb{1}(\tau = n)$ is a function of $\mathcal{X}_n$. So we can determine if $\tau = j$ or not by only considering $\mathcal{X}_j$ [29].

problem where we look for event occurrence within a specified future window. In window based intervention (WBI), time-steps where an event is present within a lookahead period are labeled as positive class examples and the rest are labeled as negative examples. A classifier may then be learnt to predict the probability of the positive class. An optimal threshold on this probability is then tuned over a validation set that measures costs of triggered interventions directly to choose an optimal trigger threshold. WBI policies are standard practice in predictive maintenance[33, 2, 22, 2, 38, 40, 13].

Time-to-Event prediction (Survival analysis): Classical survival analysis deals with predicting the time to an event (ex: death, machine failure) given a vector of observations (co-variates) obtained from a subject in the presence of incomplete (censored) observations [23]. The goal is to estimate functions such as the hazard rate (rate of death as a function of time) or survival function (probability of survival past a given time) and understand the impact of co-variates on these functions. Classical parametric model-based approaches [10, 41] have been extended in recent work using machine learning approaches [42, 18, 5, 30, 20]. In some cases, the impact of a co-variate on the survival function may vary over time (for instance, the impact of smoking may increase with age), resulting in approaches that extend survival analysis to model the time-varying effects of covariates [35, 3]. These have deep learning counterparts [31, 12] that use recurrent neural networks to model the survival function.

Recent work has applied modern deep learning approaches to longitudinal data which are sequential observations taken over time for each subject. [16, 19, 17]. Corresponding work also exists in the area of machine failure prediction [26]. The relevant papers to our work are the recurrent neural network approaches [19, 26] that operate on longitudinal data to predict the probability distribution of failure time given previous history of co-variates for any subject, at each observation time. While these methods are mostly complementary to the primary goal of this paper where we focus on obtaining optimal intervention policies, we do propose improvements in to these methods in Section 4.1 to overcome the distributional assumptions in [26] and difficulty with long sequences [19]. In practice, intervention policies based on survival analysis are typically just thresholds on the predicted time-to-event.

Optimal stopping problems: Optimal stopping problems [39, 9, 4] are a subclass of stochastic control problems with only two possible actions:stop and continue. The goal is to stop taking observations when the current reward (or cost) exceeds (or falls below) expected future reward (or cost). Optimal stopping techniques have been applied to several problems such as sequential hypothesis testing [39, 11, 7], early change point detection [29, 37, 36, 32] and early classification [25]. Optimal stopping policies for machine maintenance have been studied in reliability theory [6, 24], where the goal is to find the best stopping time to replace/maintain a machine based on condition. However, the approaches in such cases are limited to simple probabilistic models of machine degradation with a single or few co-variates. General model-free reinforcement learning approaches to the optimal stopping problem are generally feasible in situations where we have a lot of data or in a model-based setting where the underlying randomness can efficiently be simulated and the model can be used to draw Monte-Carlo samples [8]. This is not an option in most critical event prediction settings when the critical event data is often scarce and no simple model exists for the evolution of the co-variates.

1.4 Outline of contributions

The OTI problem is hard since system evolution towards a critical event is complex with typically scarce occurences of critical events. Current modeling approaches are challenged either because they make significant overly simplifying assumptions on the underlying process or because scarce events precludes powerful model-free approaches.

To our knowledge, we are the first to frame the OTI problem and link it to both survival analysis and optimal stopping theory. This allows, for the first time, the ability to model complex system evolution towards scarce critical events with the ability to intervene optimally.

- A process under observation may exhibit degradation/disorder as it evolves towards a critical event. We capture this feature by introducing the concept of a hazard rate process, which is a stochastic process modeling the evolution of conditional hazard rates as new observations are taken. A key contribution of our work is to formalize the OTI problem in equation (OTI problem) and show that the OTI problem is equivalent to an optimal stopping problem on the

underlying hazard rate process (Proposition 2.1). This allows us to tap into optimal stopping theory [39, 9, 4] to unlock the structure of OTI policies and also leverage computational methods of estimating/learning hazard rate functions. These are key to our development of a general machine learning approach to the OTI problem.

- Under Markovian assumptions on the hazard rate process, we show that the optimal intervention policy decision rule at each time step can be analytically determined by the hazard rate function at that time step (Theorem 3.1). Further we show that the optimal policy is reduces to a variable threshold on the expected time-to-event(Corollary 3.1.1), showing that static threshold policies are sub-optimal in general.
- We show that our theory applies to an important class of hazard rate processes generated by recurrent neural networks (Section 4). We term these processes Neural hazard rate processes. This allows us to apply survival analysis methods to learn hazard rate processes and analytically compute optimal intervention policies for critical events. We adapt the state of the art deep dynamic survival analysis (DRSA) work [31] to our setting. Unlike DRSA, our dynamic DRSA (DDRSA) setting requires modeling dynamic hazard rate functions. We do this by adapting the DRSA architecture to our dynamic covariate evolution setting using an encoder-decoder seq2seq architecture.

2 OTI as an optimal stopping problem on a hazard rate process

In this section we establish a link between the OTI problem, survival analysis and optimal stopping theory. We model the degradation of the monitored stochastic process towards the critical event by the corresponding evolution of underlying hazard rate functions conditioned on observation history. To this end, we will use the following definitions of a conditional *hazard rate function* $h_j(\cdot)$, and *survival function* $S_j(\cdot)$:

$$h_j(k) := \mathbb{P}(L = j + k | \mathcal{X}_j, L \geq j) \qquad S_j(k) := \mathbb{P}(L > j + k | \mathcal{X}_j, L \geq j) \tag{2}$$

Note that we have omitted the implied conditioning on the covariates, so that $h_j(k|\mathcal{X}_j)$ is written as $h_j(k)$ for convenience. Also, we have $h_j(-1) := 0$ and $S_j(-1) := 1$ reflecting the fact that the critical event has not occured until time step j. We now define the concept of a hazard rate processes:

Definition 2.1 (Hazard Rate Process)**.** Consider the random vector constructed from the conditional hazard rate function $h_j(\cdot)$ given by $\mathbf{H}_j = [h_j(0), h_j(1) \cdots h_j(L_{max} - 1)]^T$. We call the sequence of tuples $\{(\mathcal{X}_j, \mathbf{H}_j)\}$ a hazard rate process.

Thus, hazard rate processes model the relation between the evolution of co-variates and onset of the critical event.

2.1 Optimal intervention on a hazard rate process

Proposition 2.1 (Optimal intervention on a hazard rate process)**.** Given a hazard rate process $\{(\mathcal{X}_j, \mathbf{H}_j)\}$, the intervention risk at a stopping time τ is given by:

$$R^H_\tau(\mathbf{H}_\tau) = \sum_{k=0}^{L_{max}-1} \prod_{m=0}^{k} (1 - h_\tau(m)) + C_\alpha h_\tau(0) \tag{3}$$

In this setting, the optimal critical event intervention problem of equation (OTI problem) reduces to the following optimal stopping problem[3] on the corresponding hazard rate processes:

$$\tau^* = \arg \inf_{1 \leq \tau < L_{max}} \mathbb{E}\left[R^H_\tau(\mathbf{H}_\tau)\right] \tag{4}$$

Proof. We only give a proof sketch here. The proof of the theorem is given in Appendix A. Starting with terms in the OTI problem: for expected residual life we may write $(L - \tau)^+ = \sum_{k \geq 0} \mathbb{1}(L > \tau + k)$, and then use the law of iterated expectations, $\mathbb{E}[(L - \tau)^+] = \mathbb{E}[\sum_{k \geq 0} \mathbb{E}[\mathbb{1}(L > \tau + k)|\mathcal{X}_\tau]]$. We see that this expression is $\mathbb{E}[\sum_{k \geq 0} S_\tau(k)]$ and convert to hazard rates. Similarly, probability of missing the event $\mathbb{P}(L \leq \tau) = \mathbb{E}[\mathbb{E}[\mathbb{1}(L \leq \tau)|\mathcal{X}_\tau]]$, reduces to $\mathbb{E}[h_\tau(0)]$. □

[3] Problems of the form $\inf_\tau \mathbb{E}\left[R_\tau(Y_\tau)\right]$ where $R(\cdot)$ is a cost/risk function, and $\{Y_\tau\}$ is a stochastic process

Note that the formulation of equations (3) and (4) exposes the dependence of the OTI objective on the co-variates, since $h_\tau(.)$ depends on the covariates. This was implicit in the original formulation.

3 Analytic solution to the OTI problem for Markov hazard rate processes

In order to solve the OTI optimal stopping problem we make the assumption that the hazard rate processes we deal with satisfy the Markov property:

Definition 3.1 (Markov hazard rate process)**.** Consider the hazard rate process $\{(\mathcal{X}_j, \mathbf{H}_j)\}$. We call it a Markov hazard rate process, if we have:

$$\mathbb{P}(X_{j+1}|\mathcal{X}_j) = \mathbb{P}(X_{j+1}|X_j) \;\; \forall j \tag{5}$$

With this assumption in place we can state the central result of this paper:

Theorem 3.1 (OTI policy for a Markov hazard rate process)**.** *Given a Markov hazard rate process $\{(\mathcal{X}_j, \mathbf{H}_j)\}$ and associated derived risk process $\{R^H(\mathbf{H}_j)\}$, the optimal intervention policy that solves the problem posed in Proposition 2.1 is a policy $\boldsymbol{\varphi}^*$ that calls for intervention at stopping time τ^* such that:*

$$\tau^* = \min\{1 \leq j < L_{max} : \varphi_j^*(\mathbf{H}_j) = 1\} \textit{ with } \varphi_j^*(\mathbf{H}_j) = \mathbb{1}(R_\tau^H(\mathbf{H}_j) \leq V_j(\mathbf{H}_j))$$

for value functions $V_j(\mathbf{H}_j)$ given by:

$$V_j(\mathbf{H}_j) = \min_{0 \leq l < L_{max}-j} \left\{ \sum_{k=0}^{L_{max}-l-1} \prod_{m=0}^{k} (1 - h_j(l+m)) + C_\alpha \sum_{m=0}^{l} \left[\prod_{k=0}^{m-1} (1 - h_j(k)) \right] h_j(m) \right\} \tag{6}$$

Proof. We only give a proof sketch here. The complete proof of the theorem is given in Appendix A. The key is to notice that under the Markov hazard rate process assumption, and the form of of equations (3) and (4), that risk R_j depends only on X_j so we have an optimal stopping problem that has a so-called Markov Representation [29]. The optimal policy has a standard form given by $\varphi_j^*(R_j) = \mathbb{1}(R_j \leq V_j(X_j)$ with $V_j(X_j) = \inf_{l \geq 0} \mathbb{E}[R_{j+l}|\mathcal{X}_j]$. So we need only compute $\mathbb{E}[R_{j+l}|\mathcal{X}_j]$, where the expectated future risk over the possible occurences of the critical event for a rule that stops at $j+l$. Reasoning, using the hazard rates $h_j(k)$, we calculate $\mathbb{E}[R_{j+l}|\mathcal{X}_j]$ analytically giving rise to the final form for V_j. □

This theorem tells us that in the case of a hazard rate processes, the optimal policy may be constructed explicitly with knowledge of only the conditional hazard rate functions at that time. The *value* functions $V_j(\cdot)$ gives the minimum expected cost of continuing to take observations, having obtained a conditional hazard rate function ($\mathbf{H}_j$) at time j. It is clear that the the optimal policy is a time-varying threshold on the risk process in general. Also note that putting $l = 0$ in the value function recovers the instantaneous risk at time step j as the first term over which the minimum is computed in the value function. The value function gives the best stopping risk over all stopping decisions in the present and future. The following Corollary allows a further important simplification of the optimal policy:

Corollary 3.1.1 (OTI policy as a time varying threshold on time-to-event)**.** *Let T_j be the expected time to event at time j. The optimal stopping time and optimal OTI policy of Theorem 3.1 may be simplified to the following form:*

$$\tau^* = \min\{1 \leq j < L_{max} : \varphi_j^*(\mathbf{H}_j) = 1\} \textit{ with } \varphi_j^*(\mathbf{H}_j) = \mathbb{1}(T_j \leq V_j^{'}(\mathbf{H}_j))$$

where

$$V_j^{'}(\mathbf{H}_j) = \min_{0 \leq l < L_{max}-j} \left\{ \sum_{k=0}^{L_{max}-l-1} \prod_{m=0}^{k} (1 - h_j(l+m)) + C_\alpha \sum_{m=1}^{l} \left[\prod_{k=0}^{m-1} (1 - h_j(k)) \right] h_j(m) \right\} \tag{7}$$

with T_j given by:

$$T_j = \sum_{k=0}^{L_{max}} \prod_{m=0}^{k} (1 - h_j(m)) = \sum_{k=0}^{L_{max}} S_j(k) \tag{8}$$

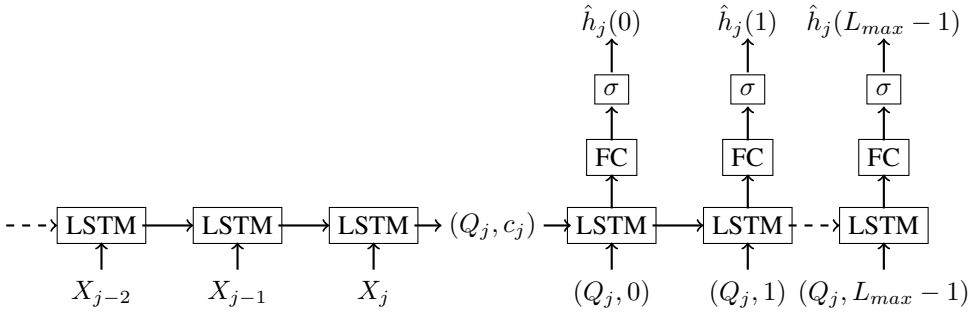


Figure 1: Architecture diagram for Deep Dynamic Survival Analysis (DDRSA).

The proof is given in Appendix A. This Corollary clearly shows that static threshold TTE policies are sub-optimal in general. The value function contain two terms. The first term, which is the risk of stopping early before the critical event which intuitively gets smaller as j progresses toward the critical event. The second term is the risk of missing the critical event that depends on the cost of missing the event. It is high where the onset of the critical event is expected.

4 Neural hazard rate processes

While it is true that one may adapt any method (in theory) that is used for Survival Analysis to estimate the condtitional hazard rate functions used in our work, RNNs are a good fit for being able to model complex patterns in multi-variate time-series data and summarizing observation history.

The previous theory applied to Markov hazard rate processes. We show that this is not a significant restriction. In fact, the class of hazard rate processes that can be modeled by recurrent neural networks (RNNs) also satisfy our Markovian assumptions. To see this, note that RNNs may be viewed as a feature transformation, mapping $\mathcal{X}_j$ to features $\mathcal{Z}_j = \{Z_s : 1 \leq s \leq j\}$ where Z_j is the hidden state vector of the RNN at time step j. If we output hazard rate vectors as functions of the hidden state we generate a *neural* hazard rate process given $\{(\mathcal{Z}_j, \mathbf{H}_j)\}$. ue to the recurrent dynamics of the RNN, the final state encode all history, therefore $\mathbb{P}(Z_{j+1}|\mathcal{Z}_j) = \mathbb{P}(Z_{j+1}|Z_j)$ and the process is Markovian.

4.1 Dynamic deep recurrent survival analysis (DDRSA)

Since the OTI theory developed in the previous section directly applies to neural neural hazard rate processes, we may fit a neural hazard rate process model to data and then apply the optimal OTI policy to intervene. Thus, our approach requires estimates of conditional hazard rates, that evolve over time (hazard rate processes).

Ren et. al. [31] developed Deep Recurrent Survival Analysis (DRSA), a deep recurrent neural architecture that uses and RNN to map a static covariate vector to a single conditonal hazard rate function (so it is not a hazard rate process that evolves over time). Unlike DRSA, our Dynamic DRSA (DDRSA) setting requires modeling dynamic survival and hazard rate functions. We do this by adapting the DRSA architecture to our dynamic covariate evolution setting. While there are other attempts at modeling the dynamic evolution of survival functions [19, 26] they either make distributional assumptions [26] or have difficulty with long sequences due to large softmax layer [19].

Figure 1 shows our DDRSA architecture which is a classic encoder-decoder architecture. DDRSA uses an encoder RNN that maps co-variate history $\mathcal{X}_j$ to an single encoder hidden state vector Q_j. This vector is then passed into a decoder which has a standard DRSA architecture, producing decoder hidden states Z_{jk} and corresponding conditional hazard rates $h_j(k) = \sigma(D(Z_{jk}))$. DDRSA can be seen as a generative model for a Markov hazard rate process.

During training, each training sample, randomly selected from one of the time-series at time step j includes a lookback window of co-variates $X_{j-K} \cdots X_j$ that are fed into the encoder input. The encoder's final hidden state is replicated and provided to the decoder as input to each time step along with a linear time-step encoding feature. We leverage the DRSA loss of Ren et. al. [31] which is

composed of the following loss components for each sample:

$$l_z = \prod_{k=0}^{l-1} \left[1 - \hat{h}_j(k)\right] \hat{h}_j(l) \ , \ l_u = \left[1 - \prod_{k=0}^{L_{max}-1} (1 - \hat{h}_j(k))\right] \ , \ l_c = \prod_{k=0}^{L_{max}-1} (1 - \hat{h}_j(k)) \quad (9)$$

l_z is the likelihood of the critical event occurance at at time $j + l$, assuming that the event is not censored. l_u is the likelihood of event occurance (a.k.a. event rate) at any future time in the decoder output, given that the sample is not censored. l_c gives the likelihood of a censored sample, when the event does not occur at a future time over the decoder outputs. The final loss for a sample is given by :

$$L_f = -\lambda \log l_z - (1 - \lambda) \left[(1 - c) \log l_u + c \log l_c\right] \quad (10)$$

where λ is a tradeoff fraction and c is the censoring indicator, with $c = 1$ for a censored sample and $c = 0$ for uncensored. Losses averaged over a mini-batch of samples are used to learn the survival model. The sequential DRSA loss allows effective supervision at each time-step allowing it to be applied to model long survival functions without resorting to restrictive parametric distributions.

During inference, we estimate, at any time-index j $h_j(0), h_j(1), \cdots h_j(L_{max} - 1)$ as decoder outputs by passing values of co-variates $X_{j-K} \cdots X_j$ into the DDRSA architecture encoder. Then, we calculate expected time to event using equation (8) and the value function using equation (7). Intervention is triggered at time j if expected time to event falls below the value function.

Note that the model architecture is such that we can generate a long sequence of hazard rates. During training time we may restrict the decoder time-steps to a fixed value ($\leq L_{max}$) and consider events outside this as censored. Such handling of long sequences is a key practical advantage of using the survival formulation.

5 Experiments

The goal of this section is twofold:

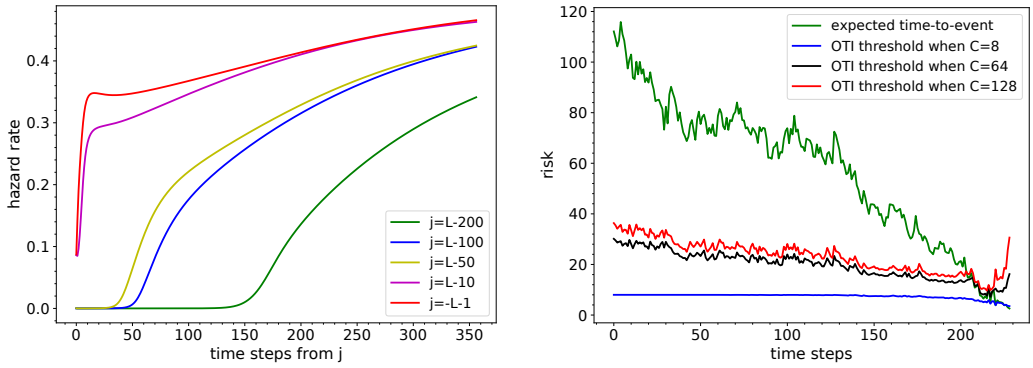
1. To complement the theory developed in the earlier sections with examples to validate and build intuition into survival and hazard rate processes how the optimal policy works to produce interventions. (Section 5.1)
2. To demonstrate that the theory does indeed translate into an effective practical methodology embodied by the DDRSA-RNN that can model hazard rate processes and generate interventions using the OTI policy. We call this combined approach the OTI-DDRSA-RNN. We show that OTI-DDRSA-RNN outperforms static threshold policies that are either window based (WBI-RNN) or time-to-event based (TTE-RNN)(Section 5.2).

Experiments are performed on two real-world datasets to validate our goals:

Turbofan Engine Failure Data [28, 34]: This dataset from NASA provides train and test data to predict Turbofan engine failures. The training set has simulated multivariate time-series observations from 21 sensors on a fleet of 218 engines. Each engine operates normally at the start and develops a fault at some point resulting in eventual failure. The test set consists of 218 partial engine time-series with remaining useful life (in engine cycles) indicated in each case. We could not use the test data as provided since it is missing part of every time series close to failure. We standardized the sensor observations to prepare them for input into our models.

Azure Predictive Maintenance Guide Data[27]: This is a dataset from a guide provided by Microsoft as a case study in failure prediction. The data comes from multiple sources including time-series of voltage, rotation, pressure and vibration measurements collected from 100 machines in real time averaged over every hour, error logs, machine information (type, age etc.) and 720 failure records. We adopt the same pre-processing, feature generation as given in the guide. Also, we downsample the data to take one sample every 15 hours.

For both datasets, we train on 70% of randomly selected co-variate time-series sequences and hold out 30% of the sequences for testing. From the training set a further 30% of the sequences are set aside as validation data to tune model parameters and policy thresholds. This process is repeated to produce 5 random splits. Model selection details and parameter settings are discussed in Appendix B.



(a) Progression of conditional hazard rates

(b) OTI policy in action

Figure 2: (a) Example of typical evolution of estimated conditional hazard rates estimated by DDRSA with progression in time step j towards the critical event. (b)The OTI policy in action. The OTI policy calls for intervention when the estimated time-to-event falls at or below a dynamically estimated policy threshold. As the cost C of missing the critical event increases, interventions are triggered earlier. The policy thresholds naturally adapt to the behavior of an individual sequence.

5.1 Discussion of OTI interventions on real data

We fit the DDRSA-RNN on the covariate time-series leading upto failure and predict on a single test set sequence. This produces, at each time step j of the sequence, an estimated conditional hazard rate vector $\mathbf{H}_j$ (representing the conditional hazard rates into the future), which gives us a probabilistic estimate of the onset of the critical event at each time step, taking into account co-variate history upto that time step. Figure 2(a) shows the evolution of estimated conditional hazard rate functions. We see that the onset of the critical event is clearly reflected in the progression of the estimated vectors $\mathbf{H}_j$.

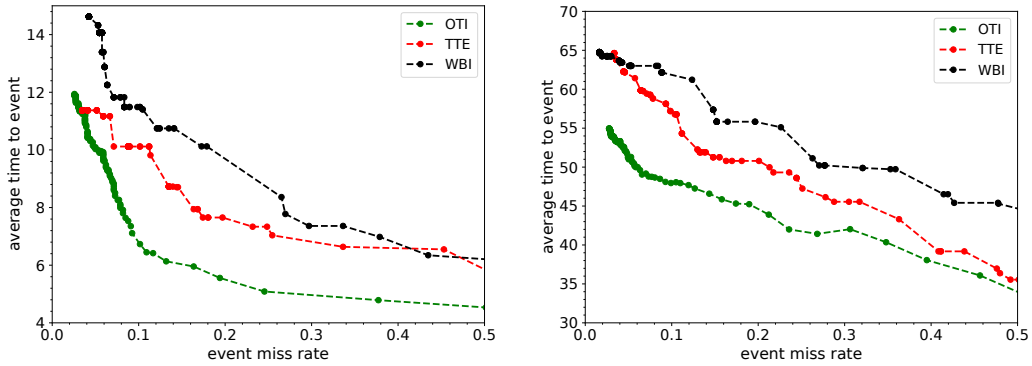
Next, we demonstrate the working of the OTI-policy, which was shown in Section 3 to be an adaptive threshold on the estimated time to event. Figure 2(b) plots the estimated time-to-event at each time step, estimated from $\mathbf{H}_j$ using equation (8). We also plot the adaptive threshold $V_j^{'}(\mathbf{H}_j)$, calculated using equation (7) of Corollary 3.1.1. An intervention is triggered when our time-to-event estimate falls on or below the adaptive threshold. Consider what happens as the cost of missing the critical event is varied. First, note that since time-to-event is independent of C, only the threshold changes. We see that increasing the cost of missing the critical event increases the threshold causing earlier intervention as we would expect.

5.2 Performance Evaluation

In this section we will evaluate the performance of the proposed OTI-DDRSA-RNN vs. two static threshold policy baselines: A window based intervention RNN (WBI-RNN) and a time-to-event based intervention RNN (TTE-RNN). All RNNs share the same encoder architecture which is an LSTM with 128 step lookback and hidden state dimension of 16 units.

WBI-RNN architecture: The WBI-RNN has the same encoder as the DDRSA-RNN with the final encoder output Q_j in Figure 1 passed to a dense (fully connected) layer to produce a single output probability per training sample. We train the model with a binary cross-entropy loss between the probability output and the binary state of event in a future window W of time steps. A threshold is used on this probability to trigger interventions once the probability is greater than the threshold. The window size and threshold are selected by cross-validation (see Apendix B).

TTE-RNN architecture: The TTE-RNN uses the DDRSA architecture shown in Figure 1 and leart with the DRSA loss as discussed above. However, interventions are trigerred with a static threshold that is set via cross-validation similar to the WBI-RNN.



(a) NASA turbofan data: policy tradeoff plot

(b) Azure PM guide data: policy tradeoff plot

Figure 3: Performance comparison of OTI-DDRSA-RNN, WBI-RNN and TTE-RNN policies on the NASA Turbofan (a) and Azure PM (b) datasets. The plots show the tradeoff between the probability of missing an event and the expected time-to-event when $C_\alpha \in \{8, 10, \cdots 256\}$.

WBI and TTE thresholds are tuned (individually, for 125 different settings of $C_\alpha \in \{8, 10, \cdots 256\}$) on each validation set using an empirical intervention policy risk:

$$R_{emp}(\varphi) = \frac{1}{N}\sum_{i=1}^{N}(L_i - \tau_i)\mathbb{1}(L_i > \tau_i) + C_\alpha \mathbb{1}(L_i \leq \tau_i) \tag{11}$$

where $(L = L_i, \tau = \tau_i)$ are realized outcomes for the ith series. All methods were trained, validated and tested using 10 random train-validation-test splits. Model parameter initialization seeds are randomized across trials. Results reported are medians over the 10 test splits.

Computational setup and inference time: All experiments were run on a Tensorbook Laptop with 32GB RAM and having a single NVIDIA GeForce GTX 1070 with Max-Q GPU. Training on a single dataset for 10 folds and 125 cost points took around 2 hours. DDRSA inference time per time-step time was around 12 milliseconds. The corresponding OTI policy evaluation time per time-step was around 0.5 milliseconds.

Figure 3 summarizes the evaluation results in the form of policy tradeoff curves. Each point represents a median over trials of the average time to event and event miss-rate, calculated for a particular setting of C_α (the corresponding standard error reported in Appendix B). Regarding the tradeoff plots, a lower frontier is preferred, since it implies a lower average time-to-event for a given miss-rate and a lower miss-rate for a given average time-to-event. We see that the OTI-DDRSA-RNN outperforms the static threshold policies on both datasets over a wide range of miss-rates.

Limitations: The quality of the OTI policy depends on the accuracy of the underlying survival analysis. Note that our approach requires fitting a survival model for $h_j(k)$ at each time step over several values of k. It is possible that some of these are better estimated than others, while in contrast a threshold algorithm can be tuned for a specific C_α on a validation set. What we expect with OTI, however is a broad range of miss-rates at which it will outperform. Being able to weight the survival estimation for enhanced policy performance for a specific C_α is an interesting next step. Also, there is no direct way of setting C_α to guaruntee a particular α. We are forced to try a range of values for C_α and pick the values with most acceptable tradeoffs.

References

[1] *Mean Residual Life — Concepts and Applications in Reliability Analysis*, pages 109–138. Springer New York, New York, NY, 2006.

[2] Azure ai guide for predictive maintenance solutions. 2018. https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/cortana-analytics-playbook-predictive-maintenance.

[3] Odd Aalen, Ørnulf Borgan, and Hakon Gjessing. *Survival and Event History Analysis: A Process Point of View*. Springer, NY, 2008.

[4] A.B. Aries and A.N. Shiryaev. *Optimal Stopping Rules*. Stochastic Modelling and Applied Probability. Springer Berlin Heidelberg, 2007.

[5] Anand Avati, Kenneth Jung, Stephanie Harman, Lance Downing, Andrew Y. Ng, and Nigam H. Shah. Improving palliative care with deep learning. *BMC Med. Inf. & Decision Making*, 18(S-4):55–64, 2018.

[6] T. Aven and U. Jensen. *Stochastic Models in Reliability*. Stochastic Modelling and Applied Probability. Springer New York, 2013.

[7] Jay Bartroff, Tze Leung Lai, and Mei-Chiung Shih. Sequential experimentation in clinical trials: Design and analysis. 2012.

[8] Sebastian Becker, Patrick Cheridito, and Arnulf Jentzen. Deep optimal stopping. *J. Mach. Learn. Res.*, 20:74:1–74:25, 2019.

[9] Y.S. Chow, H. Robbins, and D. Siegmund. *Great expectations: the theory of optimal stopping*. Houghton Mifflin, 1971.

[10] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187–220, 1972.

[11] T.S. Ferguson and T.S. Ferguson. *Mathematical Statistics: A Decision Theoretic Approach*. Probability and mathematical statistics. Academic Press, 1967.

[12] Eleonora Giunchiglia, Anton Nemchenko, and Mihaela van der Schaar. Rnn-surv: A deep recurrent model for survival analysis. In *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 23–32. Springer International Publishing, 2018.

[13] Ioana Giurgiu, Jácint Szabó, Dorothea Wiesmann, and John Bird. Predicting DRAM reliability in the field with machine learning. In *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Industrial Track, Las Vegas, NV, USA, December 11 - 15, 2017*, pages 15–21, 2017.

[14] Frank Guess and Frank Proschan. Mean residual life: Theory and applications. In *Quality Control and Reliability*, volume 7 of *Handbook of Statistics*, pages 215–224. Elsevier, 1988.

[15] Graeme Hickey, Pete Philipson, Andrea Jorgensen, and Ruwanthi Kolamunnage-Donà. Joint modelling of time-to-event and multivariate longitudinal outcomes: Recent developments and issues. *BMC Medical Research Methodology*, 16, 09 2016.

[16] Daniel Jarrett, Jinsung Yoon, and Mihaela van der Schaar. Match-net: Dynamic prediction in survival analysis using convolutional neural networks. 2018.

[17] Daniel Jarrett, Jinsung Yoon, and Mihaela van der Schaar. Dynamic prediction in clinical survival analysis using temporal convolutional networks. *IEEE Journal of Biomedical and Health Informatics*, 24:424–436, 2019.

[18] Jared Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. Deep survival: A deep cox proportional hazards network. *CoRR*, abs/1606.00931, 2016.

[19] Changhee Lee, Jinsung Yoon, and Mihaela van der Schaar. Dynamic-deephit: A deep learning approach for dynamic survival analysis with competing risks based on longitudinal data. *IEEE Transactions on Biomedical Engineering*, 67:122–133, 2019.

[20] Changhee Lee, William Zame, Jinsung Yoon, and Mihaela van der Schaar. Deephit: A deep learning approach to survival analysis with competing risks. 2018.

[21] Mei-Ling Ting Lee, George Alex Whitmore, and Bernard A. Rosner. Threshold regression for survival data with time-varying covariates. *Statistics in medicine*, 29 7-8:896–905, 2010.

[22] You-Luen Lee, Da-Cheng Juan, Xuan-An Tseng, Yu-Ting Chen, and Shih-Chieh Chang. Dc-prophet: Predicting catastrophic machine failures in datacenters. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part III*, pages 64–76, 2017.

[23] G Mackenzie, John D. Kalbfleisch, and Ross L. Prentice. *The Statistical Analysis of Failure Time Data*. 1980.

[24] V. Makis and Xiaoyue Jiang. Optimal replacement under partial observations. *Math. Oper. Res.*, 28:382–394, 05 2003.

[25] C. Martinez, G. Perrin, E. Ramasso, and M. Rombaut. A deep reinforcement learning approach for early classification of time series. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 2030–2034, 2018.

[26] Egil Martinsson. Wtte-rnn: Weibull time to event recurrent neural network. Master's thesis, Department of Computer Science and Engineering (Chalmers), Chalmers University of Technology, 2017. 103.

[27] Microsoft. Dataset from azure modeling guide for predictive maintainence. `https://github.com/microsoft/AMLWorkshop`, 2017.

[28] NASA. Phm08 challenge data set. `https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/`, 2008.

[29] H.V. Poor and O. Hadjiliadis. *Quickest Detection*. Cambridge University Press, 2008.

[30] Rajesh Ranganath, Adler Perotte, Noémie Elhadad, and David Blei. Deep survival analysis. In Finale Doshi-Velez, Jim Fackler, David Kale, Byron Wallace, and Jenna Wiens, editors, *Proceedings of the 1st Machine Learning for Healthcare Conference*, volume 56 of *Proceedings of Machine Learning Research*, pages 101–114, Children's Hospital LA, Los Angeles, CA, USA, 18–19 Aug 2016. PMLR.

[31] Kan Ren, Jiarui Qin, Lei Zheng, Zhengyu Yang, Weinan Zhang, Lin Qiu, and Yong Yu. Deep recurrent survival analysis. volume 33, pages 4798–4805, 07 2019.

[32] S. W. Roberts. A comparison of some control chart procedures. *Technometrics*, 8(3):411–430, 1966.

[33] Felix Salfner, Maren Lenk, and Miroslaw Malek. A survey of online failure prediction methods. *ACM Comput. Surv.*, 42(3):10:1–10:42, 2010.

[34] A. Saxena, K. Goebel, D. Simon, and N. Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management*, pages 1–9, 2008.

[35] Thomas H. Scheike and Torben Martinussen. *Dynamic Regression models for survival data*. Springer, NY, 2006.

[36] A. Shiryaev. On optimum methods in quickest detection problems. *Theory of Probability & Its Applications*, 8(1):22–46, 1963.

[37] A.N. Shiryaev. *Stochastic Disorder Problems*. Probability theory and stochastic modelling. Springer International Publishing, 2019.

[38] Stephan Spiegel, Fabian Mueller, Dorothea Weismann, and John Bird. Cost-sensitive learning for predictive maintenance. *CoRR*, abs/1809.10979, 2018.

[39] A. Wald. *Sequential Analysis*. Wiley Series in Probability and Mathematical Statistics: Probability and Mathematical Statistics. J. Wiley & sons, Incorporated, 1947.

[40] J. Wang, C. Li, S. Han, S. Sarkar, and X. Zhou. Predictive maintenance based on event-log analysis: A case study. *IBM Journal of Research and Development*, 61(1):11, 2017.

[41] L. J. Wei. The accelerated failure time model: A useful alternative to the cox regression model in survival analysis. *Statistics in Medicine*, 11(14-15):1871–1879, 1992.

[42] Chun-Nam Yu, Russell Greiner, Hsiu-Chin Lin, and Vickie Baracos. Learning patient-specific cancer survival distributions as a sequence of dependent regressors. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 1845–1853, 2011.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? **[TODO]**YES
   (b) Did you describe the limitations of your work? **[TODO]**YES
   (c) Did you discuss any potential negative societal impacts of your work? **[TODO]**No obvious negative societal impact
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? **[TODO]**YES
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? **[TODO]**YES
   (b) Did you include complete proofs of all theoretical results? **[TODO]**YES, in supplemental material
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? **[TODO]**YES
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? **[TODO]**YES
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? **[TODO]**YES, included in supplemental material
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? **[TODO]**YES, included in supplemental material
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? **[TODO]**YES
   (b) Did you mention the license of the assets? **[TODO]**YES, links to data provided in the paper have license information
   (c) Did you include any new assets either in the supplemental material or as a URL? **[TODO]**NO
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? **[TODO]**Not applicable. Only free license resources used
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? **[TODO]**Not applicable
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? **[TODO]**Not applicable
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? **[TODO]**Not applicable
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? **[TODO]**Not applicable