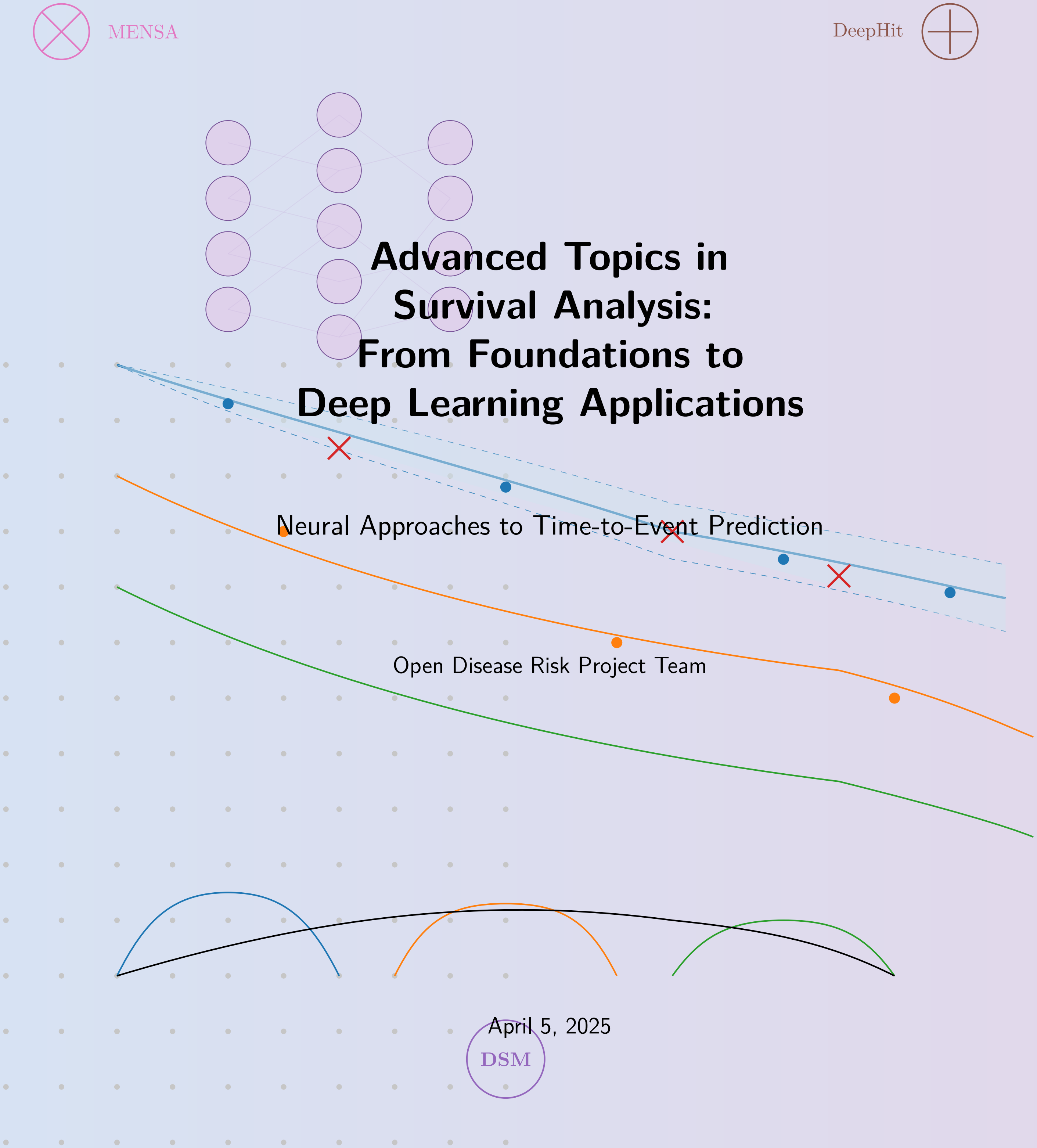


Advanced Topics in Survival Analysis: From Foundations to Deep Learning Applications

Neural Approaches to Time-to-Event Prediction

Open Disease Risk Project Team

April 5, 2025

Contents

Preface

This book provides a comprehensive introduction to survival analysis with a special focus on deep learning approaches. It transforms the slide deck format into an educational text with in-depth explanations, mathematical foundations, and practical implementation details.

Chapter 1

Introduction

1.1 What is Survival Analysis?

Survival analysis comprises a collection of statistical methods designed for analyzing **time-to-event data** - data where the primary interest is in the time until a specific event occurs. Unlike standard regression or classification problems, survival analysis specifically addresses scenarios where we want to answer "how long until" questions. These methods have widespread applications across multiple disciplines:

- **Medical research:** How long will a patient survive following diagnosis? What factors affect progression-free survival?
- **Engineering:** When will a machine component fail? How does maintenance affect lifetime?
- **Economics:** How long will a customer maintain a subscription? What influences customer retention?
- **Sociology:** When will someone find employment after completing education? What factors affect time to re-employment?

Key Terminology

Different fields often use different terminology for the same survival analysis concepts:

- **Medicine:** Survival analysis, time-to-event analysis
- **Engineering:** Reliability analysis, failure time analysis
- **Economics:** Duration analysis, event history analysis
- **Sociology:** Event history analysis, transition analysis

Despite these different names, the underlying statistical principles remain the same.

1.2 Distinctive Features of Survival Data

Survival data possesses several unique characteristics that distinguish it from other types of data and necessitate specialized analytical methods:

1.2.1 Censoring

The most distinctive feature of survival data is **censoring**, which occurs when we do not observe the exact time of the event for some subjects. Censoring introduces a fundamental challenge: we have incomplete information, but that incomplete information still provides valuable insights that we need to incorporate into our analysis.

Censoring

Censoring occurs when the event of interest is not observed within the study period or observation window. Instead of the exact event time, we only know that the event either:

- Has not yet occurred by a certain time (right censoring)
- Occurred before a certain time (left censoring)
- Occurred within a specific time interval (interval censoring)

Right Censoring

Right censoring is the most common type of censoring in survival analysis. It occurs when a subject exits the study before experiencing the event of interest.

Examples of Right Censoring

- A patient is still alive at the end of a clinical trial's follow-up period
- A study participant withdraws from the study before experiencing the event
- A customer is still subscribed to a service when data collection ends
- A machine component is still functioning at the end of an observation period

In each case, we only know that the subject survived *at least* until the censoring time, but we don't know the exact event time.

Figure 1.1 illustrates right censoring in a hypothetical clinical study. Subjects 1 and 2 experience the event of interest (e.g., disease progression), while Subject 3 is right-censored at the end of the study period.

Left Censoring

Left censoring occurs when the event of interest happens before the first observation time, but we don't know exactly when it occurred.

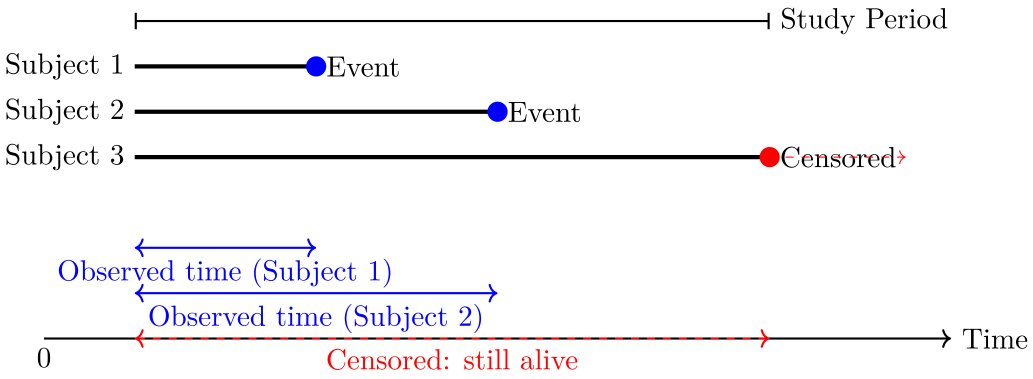


Figure 1.1: Illustration of right censoring in a clinical study. Subjects 1 and 2 experience the event of interest, while Subject 3 is right-censored at the end of the study period. We know Subject 3 survived at least until time 8, but we don't know their exact event time.

Examples of Left Censoring

- A patient already has the disease when first examined, but we don't know when it developed
- Environmental contamination detected at the start of monitoring, but we don't know when it began
- A customer had already canceled their subscription before the study began
- A component is discovered to be already failed at the first inspection

In these cases, we only know that the event occurred *sometime before* the first observation.

Figure 1.2 shows left censoring in a disease monitoring study. Subject 1 is found to have the disease at the study entry, representing left censoring.

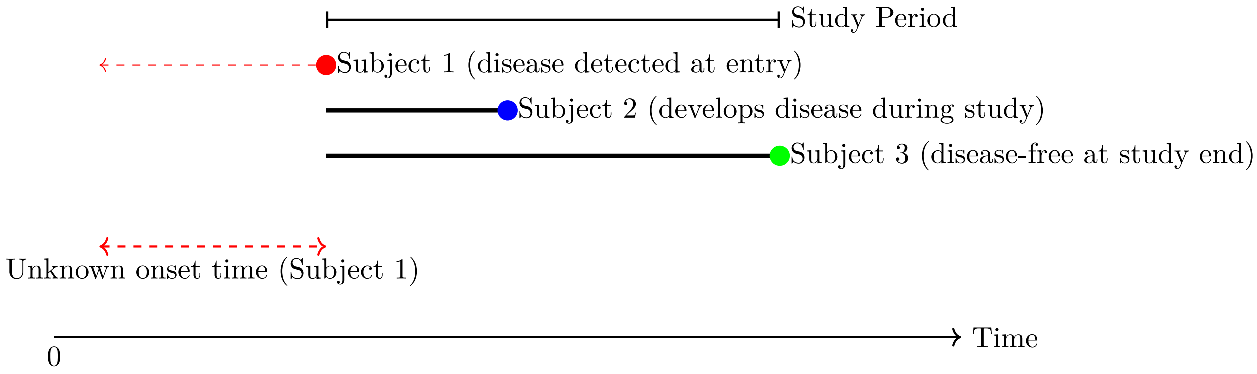

Study Period
Subject 1 (disease detected at entry)
Subject 2 (develops disease during study)
Subject 3 (disease-free at study end)
Unknown onset time (Subject 1)
Time
0

Figure 1.2: Illustration of left censoring. Subject 1 already has the disease when entering the study, so we only know the disease onset occurred sometime before the study entry point.

Interval Censoring

Interval censoring occurs when we know the event occurred within a specific time interval, but not the exact time.

Examples of Interval Censoring

- A tumor is detected during a regular screening visit, but was not present at the previous visit
- Infection is detected during a follow-up visit, but was not present at the previous visit
- A machine is found to be malfunctioning during a scheduled inspection, but was working at the last inspection
- A crack in a structure is discovered during routine examination, but was not present at the previous examination

We know the event occurred within the interval between observations, but not the exact time.

Figure 1.3 illustrates interval censoring in a periodic screening context. The condition is detected at Visit 3, but was not present at Visit 2, so the exact onset time is interval-censored between these visits.

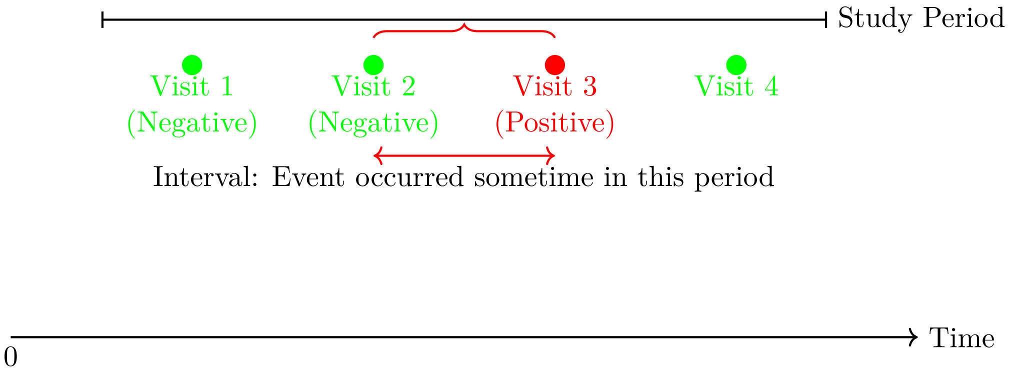


Figure 1.3: Illustration of interval censoring. The condition is negative at Visit 2 but positive at Visit 3, so we know the onset occurred within this time interval, but not the exact time.

1.2.2 Time-Varying Effects and Covariates

In survival analysis, the relationship between predictors and outcomes can change over time:

- **Time-varying effects:** The impact of a covariate on survival may change over time. For example, a treatment might be most effective in the short term but diminish in effectiveness over time.
- **Time-varying covariates:** The values of predictors may change over the course of observation. For instance, a patient's biomarker levels or medication dosages might change throughout a study.

1.2.3 Multiple Outcomes and Competing Risks

In many real-world scenarios, subjects may experience different types of events:

- **Competing risks:** Multiple possible event types where the occurrence of one event precludes the observation of others. For example, patients might die from cancer, heart disease, or other causes, but once one cause of death occurs, the others cannot be observed.
- **Recurrent events:** The same event can occur multiple times for a subject, such as hospital readmissions or equipment failures.
- **Multi-state processes:** Subjects can transition through various states over time, like different stages of disease progression.

1.3 Core Questions in Survival Analysis

Survival analysis addresses several fundamental questions:

1.3.1 Survival Probability Estimation

What is the probability that a subject will survive (not experience the event) beyond a specific time point? This is typically represented by the *survival function*, denoted as $S(t)$, which gives the probability of surviving beyond time t.

Survival Function

$$S(t) = P(T > t) \tag{1.1}$$

where T is the time-to-event random variable. The survival function has the following properties:

- $S(0) = 1$ (all subjects are event-free at the start)
- $\lim_{t\to\infty} S(t) = 0$ (eventually all subjects experience the event)
- $S(t)$ is monotonically decreasing (survival probability cannot increase with time)

Figure 1.4 shows a typical survival curve, which illustrates how the probability of surviving beyond time t decreases over time.

1.3.2 Risk Assessment: Hazard Function

What is the instantaneous risk of experiencing the event at a specific time, given survival up to that time? This is represented by the *hazard function*, denoted as $h(t)$.

Hazard Function

$$h(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t | T \geq t)}{\Delta t} \tag{1.2}$$

The hazard function represents the instantaneous rate of event occurrence at time t, conditional on having survived until time t.

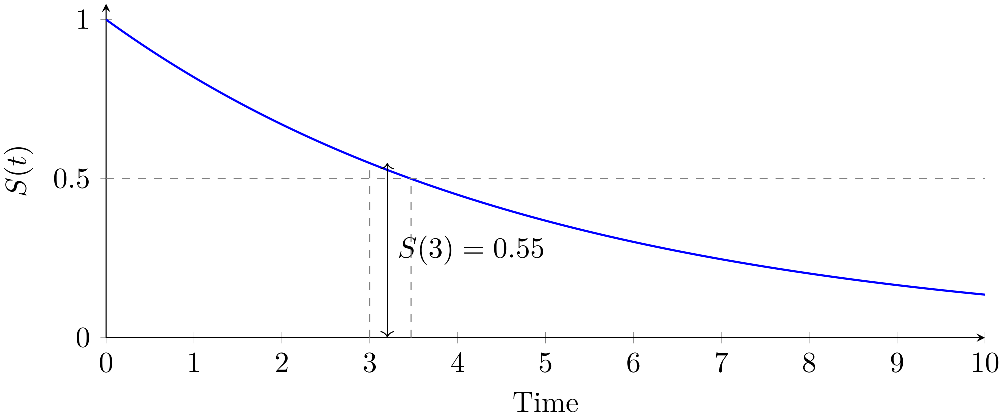


Figure 1.4: A typical survival function showing the probability of surviving beyond time t. The dashed lines indicate how to find the median survival time (where $S(t) = 0.5$).

Unlike the survival function, the hazard function can take many shapes, as illustrated in Figure 1.5.

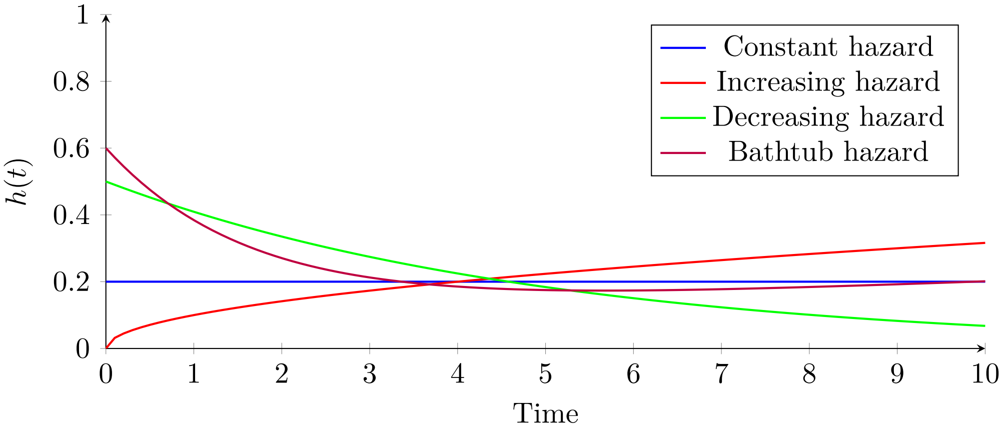


Figure 1.5: Different patterns of hazard functions. The constant hazard corresponds to the exponential distribution, increasing hazard might indicate aging or wear, decreasing hazard can represent early failures, and the bathtub pattern is common in reliability engineering.

1.3.3 Covariate Effects

How do various factors (covariates) affect survival probabilities and hazard rates? This is typically addressed through regression models that relate covariates to the hazard function or survival function.

Cox Proportional Hazards Model

$$h(t|\mathbf{X}) = h_0(t)\exp(\boldsymbol{\beta}^T\mathbf{X}) \tag{1.3}$$

where $h_0(t)$ is the baseline hazard function, $\mathbf{X}$ is the vector of covariates, and $\boldsymbol{\beta}$ represents the coefficients indicating how each covariate affects the hazard ratio.

1.3.4 Group Comparisons

Are there differences in survival experiences between groups? This is often assessed using non-parametric tests like the log-rank test or through regression modeling.

1.3.5 Advanced Questions

As the field progresses, particularly with the incorporation of machine learning techniques, additional questions are being addressed:

- How to model complex, non-linear relationships between covariates and outcomes?
- How to incorporate unstructured data (images, text, etc.) into survival models?
- How to handle competing risks and multi-state processes efficiently?
- How to provide uncertainty quantification for survival predictions?
- How to use survival predictions to guide personalized treatment decisions?

1.4 Traditional Approaches to Survival Analysis

Survival analysis has traditionally been approached through three main categories of methods: non-parametric, semi-parametric, and fully parametric. Each has its strengths and limitations.

1.4.1 Non-Parametric Methods

Non-parametric methods make no assumptions about the underlying distribution of survival times. They are useful for exploratory analysis and for comparing survival experiences between groups.

Kaplan-Meier Estimator

The Kaplan-Meier estimator is the most widely used non-parametric method for estimating the survival function from censored data.

Kaplan-Meier Estimator

$$\hat{S}(t) = \prod_{t_i \leq t} \left(1 - \frac{d_i}{n_i}\right) \tag{1.4}$$

where:

- t_i are the distinct event times observed in the data
- d_i is the number of events at time t_i
- n_i is the number of subjects at risk just before time t_i

The Kaplan-Meier estimator produces a step function that decreases at each event time, as shown in Figure 1.6.

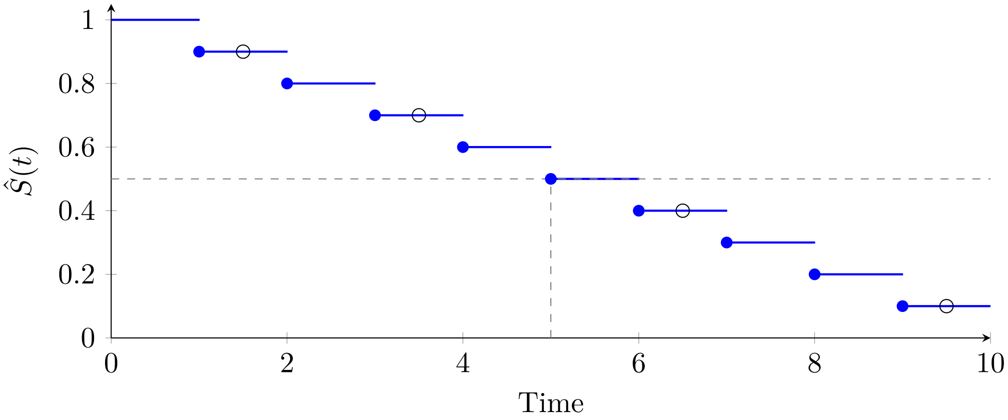


Figure 1.6: Kaplan-Meier survival curve. The step function drops at each event time, and censored observations are shown as open circles. The median survival time is 5 (where the curve crosses the 0.5 probability line).

Log-Rank Test

The Log-Rank test is a non-parametric statistical test used to compare survival distributions between two or more groups.

Log-Rank Test Statistic

$$\chi^2 = \frac{(O_1 - E_1)^2}{E_1} + \frac{(O_2 - E_2)^2}{E_2} + \cdots + \frac{(O_G - E_G)^2}{E_G} \tag{1.5}$$

where:

- O_g is the observed number of events in group g
- E_g is the expected number of events in group g under the null hypothesis
- G is the number of groups

The test statistic follows a chi-square distribution with $(G - 1)$ degrees of freedom under the null hypothesis of no difference between groups.

1.4.2 Semi-Parametric Methods

Semi-parametric methods make some assumptions about the relationship between covariates and hazard, but leave the baseline hazard function unspecified.

Cox Proportional Hazards Model

The Cox proportional hazards model is the most widely used regression model in survival analysis. It assumes that the effect of covariates is to multiply the hazard by a constant factor, but makes no assumptions about the shape of the baseline hazard.

Cox Proportional Hazards Model

$$h(t|\mathbf{X}) = h_0(t)\exp(\boldsymbol{\beta}^T\mathbf{X}) \tag{1.6}$$

where:

- $h_0(t)$ is the baseline hazard function (left unspecified)
- $\mathbf{X}$ is the vector of covariates
- $\boldsymbol{\beta}$ is the vector of regression coefficients

The key assumption is that the hazard ratio between any two individuals is constant over time (proportional hazards assumption).

The Cox model can be fit using partial likelihood, which allows estimation of $\boldsymbol{\beta}$ without having to specify $h_0(t)$. This makes it very flexible and robust.

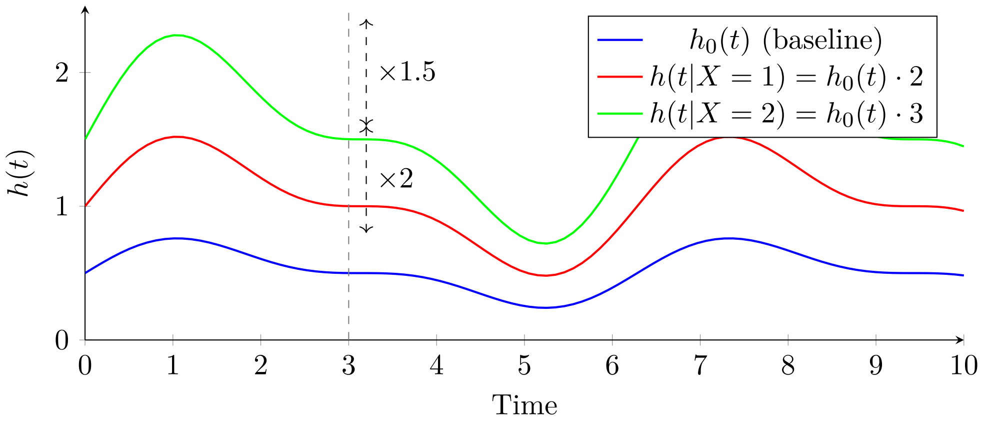


Figure 1.7: Illustration of the proportional hazards assumption in the Cox model. The hazard ratios between different covariate values remain constant over time, regardless of the shape of the baseline hazard.

Extensions of the Cox Model

Several extensions of the Cox model have been developed to address specific limitations:

- **Stratified Cox model:** Allows for different baseline hazards across strata while maintaining the same covariate effects.
- **Time-dependent Cox model:** Allows for time-varying covariates and time-varying effects through terms like $X(t)$ and $\beta(t)$.
- **Frailty models:** Incorporate random effects to account for unobserved heterogeneity and clustering in the data.
- **Competing risks extensions:** Adaptations for handling competing events, such as cause-specific Cox models and Fine-Gray models.

1.4.3 Fully Parametric Methods

Parametric methods specify a full probability distribution for the survival times. These methods are more efficient when the distributional assumptions are correct and allow for direct modeling of the survival time rather than just the hazard.

Common Parametric Distributions

Several probability distributions are commonly used in parametric survival analysis:

Exponential Distribution The exponential distribution assumes a constant hazard rate, which corresponds to a memoryless process.

Exponential Distribution

$$h(t) = \lambda \tag{1.7}$$

$$S(t) = e^{-\lambda t} \tag{1.8}$$

$$f(t) = \lambda e^{-\lambda t} \tag{1.9}$$

where $\lambda > 0$ is the rate parameter.

Weibull Distribution The Weibull distribution allows for both increasing and decreasing hazard rates, depending on the shape parameter.

Weibull Distribution

$$h(t) = \alpha\lambda(\lambda t)^{\alpha-1} \tag{1.10}$$

$$S(t) = e^{-(\lambda t)^{\alpha}} \tag{1.11}$$

$$f(t) = \alpha\lambda(\lambda t)^{\alpha-1}e^{-(\lambda t)^{\alpha}} \tag{1.12}$$

where $\alpha > 0$ is the shape parameter and $\lambda > 0$ is the scale parameter.

- If $\alpha = 1$, the Weibull reduces to the exponential distribution
- If $\alpha > 1$, the hazard increases with time
- If $\alpha < 1$, the hazard decreases with time

Log-normal and Log-logistic Distributions These distributions can model non-monotonic hazard rates, where the hazard first increases and then decreases over time.

Generalized Gamma Distribution A flexible three-parameter family that includes many other distributions as special cases.

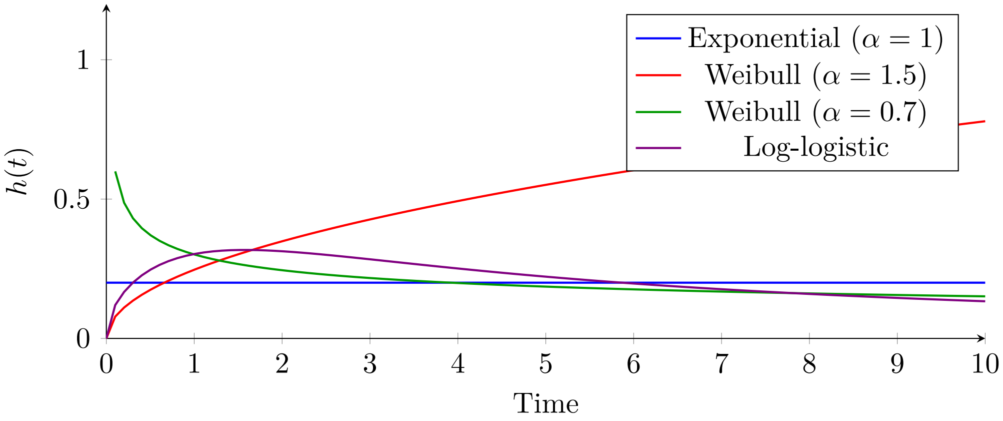


Figure 1.8: Hazard functions for different parametric survival distributions. Note how the shape parameter in the Weibull distribution controls whether the hazard increases or decreases with time, while the log-logistic allows for a non-monotonic hazard.

Parametric Regression Models

Parametric regression models relate covariates to the parameters of these distributions. For example, in a Weibull regression model:

Parametric Regression Model

$$h(t|\mathbf{X}) = \alpha\lambda(\mathbf{X})(\lambda(\mathbf{X})t)^{\alpha-1} \tag{1.13}$$

where $\lambda(\mathbf{X}) = \lambda_0 \exp(\boldsymbol{\beta}^T\mathbf{X})$, allowing covariates to affect the scale parameter.

1.5 Limitations of Traditional Methods

Despite their utility, traditional survival analysis methods face several important limitations that have motivated the development of more advanced approaches.

1.5.1 Modeling Constraints

Linear Relationship Assumptions

Traditional methods typically assume linear relationships between covariates and the log hazard or log survival time. This can be limiting when:

- The true relationships are non-linear
- Complex interactions exist between covariates
- The effect of a predictor varies substantially across its range

While transformations and interaction terms can address some of these issues, they require manual specification and prior knowledge of the functional form.

Restrictive Functional Forms

Parametric models are constrained by the chosen probability distribution:

- Limited flexibility in modeling complex hazard patterns
- Distribution selection can significantly affect results
- Multimodal hazard functions are difficult to capture

1.5.2 High-Dimensional Data Challenges

Traditional survival methods were not designed for modern high-dimensional data:

- Poor performance with large numbers of features
- Inability to directly handle unstructured data (images, text, signals)
- Need for extensive feature engineering
- Difficulty in capturing complex interactions automatically
- Inefficient with sparse or noisy features

1.5.3 Time-Varying Challenges

Handling time-varying aspects presents significant challenges:

- Complex implementation for time-varying covariates
- Special techniques needed for non-proportional hazards
- Limited ability to capture temporal patterns automatically
- Difficulty handling irregular time series

1.6 Modern Deep Learning Approaches

To address these limitations, researchers have developed a range of deep learning approaches for survival analysis, which fall into several categories.

1.6.1 Neural Cox Extensions

These methods maintain the semi-parametric nature of the Cox model but replace the linear predictor with neural networks.

Neural Cox Models

$$h(t|\mathbf{X}) = h_0(t)\exp(f_{\text{NN}}(\mathbf{X})) \tag{1.14}$$

where $f_{\text{NN}}(\mathbf{X})$ is a neural network function of the input features.

Examples include:

- DeepSurv: Uses feedforward neural networks with the partial likelihood
- Cox-Time: Extends to time-varying effects through neural networks

1.6.2 Discrete-Time Neural Survival Models

These approaches discretize time into intervals and predict the conditional probability of an event in each interval, often implementing this as a sequence prediction problem.

Discrete-Time Survival Models

$$P(t_k \leq T < t_{k+1} | T \geq t_k, \mathbf{X}) = f_{\text{NN},k}(\mathbf{X}) \tag{1.15}$$

where $f_{\text{NN},k}(\mathbf{X})$ is the output of a neural network for the k-th time interval.

Examples include:

- Nnet-survival: Uses multi-task logistic regression for each discrete time interval
- DeepHit: Predicts the probability mass function across discretized time intervals

1.6.3 Deep Parametric Survival Models

These models use neural networks to parameterize probability distributions, allowing for flexible modeling of complex survival patterns.

Deep Parametric Survival Models

$$S(t|\mathbf{X}) = S(t|\boldsymbol{\theta}_{\text{NN}}(\mathbf{X})) \tag{1.16}$$

where $\boldsymbol{\theta}_{\text{NN}}(\mathbf{X})$ are the distribution parameters learned by neural networks.

Examples include:

- Deep Survival Machines (DSM): Uses a mixture of parametric distributions
- Deep Weibull: Parameterizes Weibull distributions with neural networks

1.6.4 Representation-Focused Models

These approaches focus on learning powerful representations from complex input data:

- Survival transformers: Use attention mechanisms for feature interactions
- Recurrent neural networks: For temporal data and time-varying covariates
- Graph neural networks: For structured relationship data
- Multi-task survival learning: For joint prediction of related outcomes

1.7 Advantages of Deep Learning for Survival Analysis

Deep learning approaches offer several important advantages for survival analysis:

1.7.1 Feature Learning

Neural networks can automatically extract relevant features from raw data:

- Direct learning from unstructured data (images, text, signals)
- Transfer learning from pre-trained representations
- Reduced need for manual feature engineering
- Ability to handle high-dimensional inputs efficiently

1.7.2 Flexible Relationship Modeling

Deep learning models can capture complex relationships:

- Non-linear relationships between covariates and hazards
- Automatic interaction detection
- Handling of high-dimensional feature spaces
- Implicit regularization through network architecture

1.7.3 Beyond Proportional Hazards

Neural networks can model more complex hazard patterns:

- Non-proportional hazards without explicit specification
- Time-varying effects handled naturally
- Complex temporal patterns learned from data
- Multi-modal hazard distributions

1.7.4 Multi-Event Modeling

Deep learning facilitates joint prediction of multiple outcomes:

- Simultaneous prediction of multiple event types
- Capturing dependencies between different events
- Flexible competing risks modeling
- Information sharing across related outcomes

1.8 Course Roadmap

This book presents a comprehensive learning path from traditional survival analysis to advanced deep learning approaches, organized as follows:

1.8.1 Foundations (Chapters 1-3)

- Introduction to survival analysis concepts
- Statistical foundations and likelihood principles
- Traditional survival models and their limitations

1.8.2 Data Representation (Chapter 4)

- Handling survival data in deep learning
- Embedding techniques for categorical and continuous features
- Representation learning for survival analysis

1.8.3 Deep Survival Models (Chapters 5-7)

- Deep Survival Machines (DSM)
- Multi-Event Neural Survival Analysis (MENSA)
- Loss functions for survival prediction

1.8.4 Advanced Topics (Chapters 8-9)

- Numerical stability in survival models
- Expert knowledge integration
- Applications and case studies

Each chapter builds on the knowledge from previous chapters, providing a structured learning path from fundamental concepts to cutting-edge research in deep learning for survival analysis.

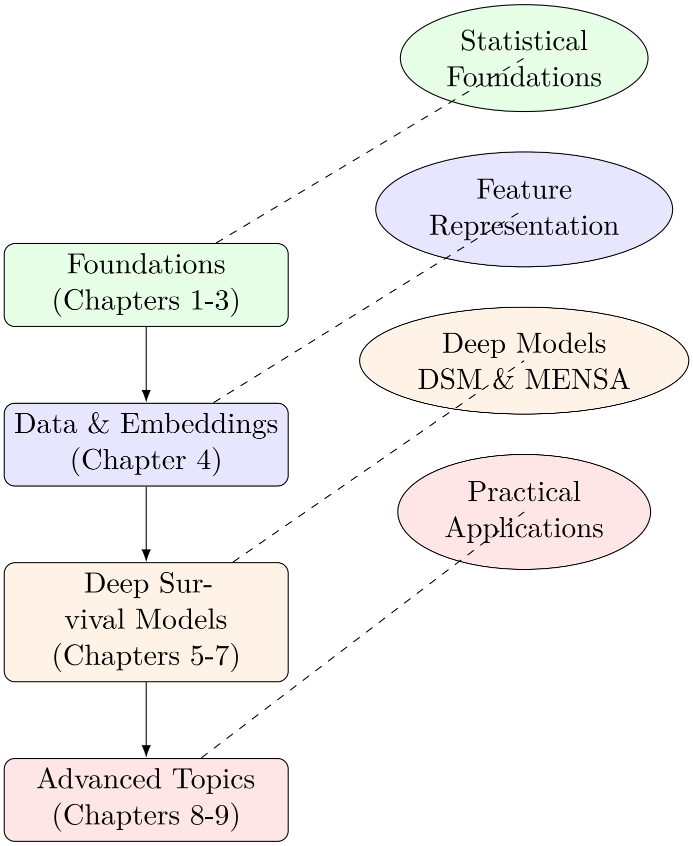

Statistical Foundations
Feature Representation
Deep Models DSM & MENSA
Practical Applications
Foundations (Chapters 1-3)
Data & Embeddings (Chapter 4)
Deep Survival Models (Chapters 5-7)
Advanced Topics (Chapters 8-9)

Figure 1.9: Course roadmap showing the progression from foundational concepts to advanced topics in deep learning for survival analysis.

Chapter 2

Motivation and Problem Setting

2.1 The Need for Time-to-Event Analysis

Many real-world scenarios require us to understand not just *if* an event will occur, but *when* it might happen. This timing aspect adds complexity to traditional predictive modeling approaches, but also provides valuable information that can lead to better decision-making and outcomes.

Time-to-Event Questions Across Domains

Time-to-event analysis answers crucial questions across numerous fields:

- **Healthcare:** When will a patient experience disease progression? How long will a treatment remain effective?
- **Engineering:** How long will a mechanical component function before failure? When should maintenance be scheduled?
- **Business:** When will a customer likely cancel their subscription? When might an employee leave the company?
- **Technology:** When will a system failure occur? What is the expected lifetime of a software feature?

Traditional predictive models typically focus on classification (will an event occur?) or regression (how much of something will occur?), but time-to-event analysis introduces a temporal dimension that requires specialized methodologies. Without proper handling of the time component, predictions can be severely biased and lead to incorrect conclusions.

2.2 Applications Across Domains

The need for time-to-event analysis spans virtually every field where timing information is critical for decision-making.

2.2.1 Healthcare Applications

In medicine, understanding the timing of events can directly impact treatment decisions and patient outcomes:

- **Patient survival analysis:** Estimating survival time after diagnosis or treatment
- **Disease progression:** Predicting time to progression or recurrence
- **Treatment efficacy:** Determining duration of treatment effectiveness
- **Adverse event timing:** Analyzing when side effects might occur
- **Hospital readmission:** Predicting when patients might return after discharge
- **Clinical trial analysis:** Comparing time-to-event outcomes between treatment groups

Time-to-event analysis in healthcare directly informs treatment planning, resource allocation, and the development of clinical guidelines.

2.2.2 Engineering and Reliability Applications

In engineering contexts, time-to-event analysis helps optimize maintenance schedules, warranty periods, and resource allocation:

- **Equipment failure prediction:** Estimating when machinery might fail
- **Component lifetime estimation:** Predicting the functional lifespan of parts
- **Maintenance optimization:** Scheduling preventive maintenance to minimize downtime
- **System reliability assessment:** Evaluating the reliability of complex systems over time
- **Warranty period modeling:** Setting appropriate warranty terms based on failure timing
- **Quality control:** Analyzing time-to-failure patterns to improve manufacturing

These applications help reduce costs, improve safety, and enhance product quality.

2.2.3 Business and Economic Applications

In business contexts, time-to-event analysis provides insights into customer behavior, employee retention, and financial risk:

- **Customer churn prediction:** Forecasting when customers might leave
- **Subscription lifecycle:** Modeling subscription duration patterns
- **Employee retention:** Predicting when employees might leave
- **Loan default timing:** Estimating when borrowers might default
- **Insurance claim occurrence:** Predicting timing of insurance claims
- **Project completion:** Modeling time-to-completion for projects

These applications help businesses optimize resource allocation, improve customer retention strategies, and manage financial risk.

2.2.4 Other Diverse Applications

Time-to-event analysis extends to many other fields:

- **Social sciences:** Modeling unemployment duration, marriage longevity
- **Technology:** Predicting software bug discovery rates, digital content lifespan
- **Environmental science:** Analyzing timing of extreme events, species extinction risks
- **Academia:** Studying citation timing, research impact evolution
- **Public policy:** Evaluating time-dependent effects of policy interventions

The ubiquity of time-to-event questions across domains highlights the critical importance of developing robust methodologies for modeling and analyzing such data.

2.3 Unique Characteristics of Time-to-Event Data

Time-to-event data presents several distinct challenges that require specialized analytical approaches. These characteristics make standard regression or classification methods inadequate without careful modification.

2.3.1 Incomplete Observations

Unlike conventional datasets where each observation has a complete set of values, time-to-event data frequently includes incomplete observationsa phenomenon known as censoring. Censoring occurs when the event of interest is not observed for some subjects, but we still have partial information about their event times.

In a clinical study, for example, patients may:

- Still be alive at the end of the study period
- Withdraw from the study before experiencing the event
- Be lost to follow-up for various reasons

These incomplete observations contain valuable information that must be properly incorporated into the analysis rather than discarded.

2.3.2 Variable Follow-up Durations

Subject follow-up periods typically vary considerably within a single dataset, with some subjects observed for short periods and others for much longer. This variability creates challenges for standardizing predictions and assessing model performance.

2.3.3 Time-Varying Covariates

Many predictors in time-to-event contexts change over the observation period. For example:

- A patient's biomarker values may fluctuate during treatment
- A machine's operational parameters may shift over time
- A customer's engagement metrics may evolve throughout the relationship

These dynamic covariates need special handling to accurately model their effects on the timing of events.

2.3.4 Multiple Competing Events

In many real-world scenarios, subjects may experience various mutually exclusive events:

- A patient might die from cancer, heart disease, or other causes
- A mechanical system might fail due to wear, electrical issues, or external damage
- A customer might churn for different reasons (price, service quality, competitor offers)

These competing events require models that can account for multiple possible outcomes and their interdependencies.

2.3.5 Informative Observation Patterns

In time-to-event data, the pattern of observations itself may contain information relevant to the outcome. For example, the frequency of patient visits might correlate with disease severity, or the timing of system inspections might relate to suspected issues. This informative missingness adds another layer of complexity to the analysis.

2.4 The Fundamental Challenge: Censoring

Censoringthe partial observation of event timesrepresents the most distinctive and challenging aspect of time-to-event data. Censoring occurs when we don't observe the exact time of the event for a subject, but we have some information about when it might have occurred.

A comprehensive discussion of censoring types, mechanisms, and their implications can be found in Chapter 4. Here, we highlight the key challenges that censoring introduces in time-to-event analysis.

Key Challenges with Censored Data

Censoring introduces several unique challenges:

- Standard statistical methods cannot be directly applied
- Naive approaches (ignoring censored observations or treating censoring times as event times) lead to biased results
- Different forms of censoring require different analytical approaches
- Assumptions about the censoring mechanism are critical for valid inference

2.5 Visualizing Survival Data

A comprehensive visual representation of survival data helps illustrate the unique challenges of time-to-event analysis. Figure 2.1 shows a typical survival dataset with varying follow-up times and both observed events and censored observations.

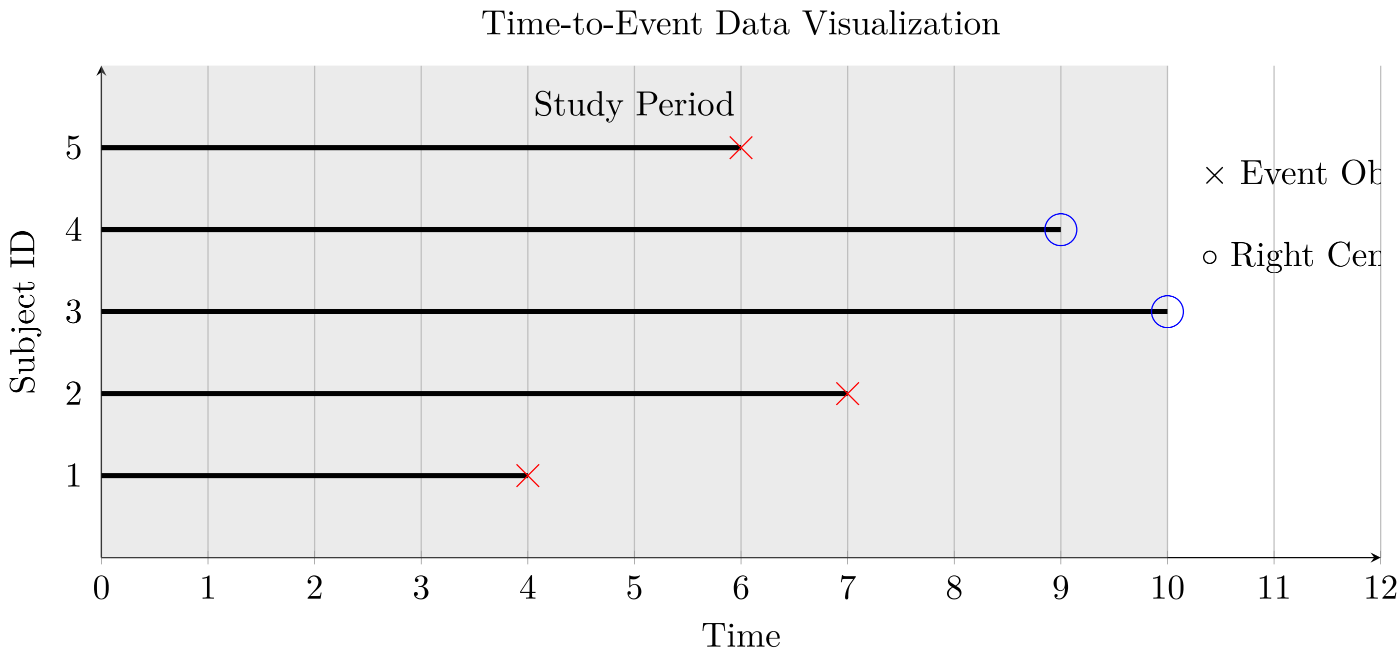


Figure 2.1: Visualization of time-to-event data for five subjects. Each horizontal line represents a subject's follow-up period. Red crosses indicate observed events, while blue circles represent right-censored observations. Note the varying follow-up durations and the mixture of event types.

This visualization highlights several key characteristics of survival data:

- **Variable follow-up durations:** Subjects are observed for different periods
- **Mixture of observed events and censoring:** Some subjects experience the event during observation, while others are censored
- **Administrative censoring:** Some subjects are censored at the study end
- **Early withdrawals:** Some subjects exit the study before its conclusion

2.6 Competing Risks: Another Challenge

Another significant challenge in time-to-event analysis is the presence of competing risksmultiple possible event types where the occurrence of one event precludes the observation of others. Chapter 4 provides a detailed discussion of competing risks, their analysis, and modeling approaches.

Competing risks fundamentally change how we approach survival analysis:

- **Event-specific hazards:** Each event type has its own hazard function

- **Cumulative incidence functions:** Standard Kaplan-Meier estimates become inappropriate; we need to use cumulative incidence functions that account for competing events
- **Interpretation changes:** Risk factors may have different effects on different event types
- **Modeling complexity:** Models must account for the interdependence between different event types

Ignoring competing risks can lead to substantial biases, particularly the overestimation of event probabilities when using standard survival methods.

2.7 Beyond Binary Events: Complex Time-to-Event Scenarios

Real-world time-to-event scenarios often extend beyond simple binary events (occurred/not occurred) to include more complex patterns:

2.7.1 Recurrent Events

Many events can occur multiple times for the same subject:

- Hospital readmissions
- Equipment failures and repairs
- Episodic disease flares
- Repeated customer purchases

Recurrent event analysis requires specialized methods that can model the correlation between events within the same subject and account for potential changes in risk patterns after each event.

2.7.2 Multi-State Processes

In multi-state processes, subjects can transition through various states over time:

- Disease progression through different stages
- Employee transitions between roles
- Customer journey through different product tiers
- System transitions between operational states

Multi-state models extend survival analysis to handle these complex transition patterns, with each transition potentially having its own risk factors and temporal patterns.

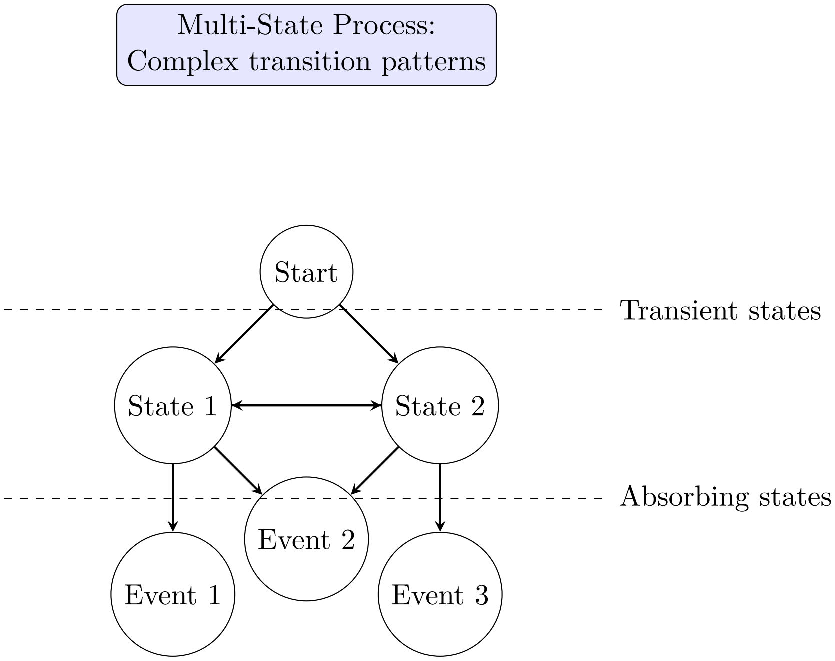


Figure 2.2: A multi-state process model. Subjects can transition between transient states and eventually reach absorbing states (terminal events). Each transition has its own hazard function that may depend on subject characteristics and history.

2.7.3 Joint Longitudinal and Time-to-Event Data

In many applications, we observe both:

- Longitudinal measurements of variables over time (e.g., biomarkers, system parameters)
- Time-to-event outcomes (e.g., failure, disease progression)

Joint models for longitudinal and time-to-event data enable:

- Incorporating time-varying measurements into event predictions
- Accounting for measurement error in longitudinal data
- Addressing informative dropout in longitudinal studies
- Providing dynamic, updated predictions as new measurements arrive

2.8 Summary

The unique characteristics of time-to-event dataparticularly censoring, variable follow-up durations, competing risks, and complex event patternsnecessitate specialized analytical approaches. Standard regression and classification methods are inadequate without proper adaptation to handle these distinctive features.

The next chapters will explore both traditional and modern approaches to addressing these challenges, beginning with the statistical foundations of survival analysis and progressing to advanced deep learning methods. By understanding these specialized techniques, we can extract meaningful insights from time-to-event data across diverse domains, leading to better predictions, improved decision-making, and ultimately better outcomes.

Chapter 3

Survival Analysis Foundations

3.1 Foundations of Survival Analysis

Survival analysis comprises a rich collection of statistical methods specifically designed to analyze time-to-event data (Kleinbaum and Klein, 2012). As introduced in previous chapters, these methods address scenarios where the primary interest is the timing of events rather than just whether they occur (Cox, 1972). This chapter provides a comprehensive overview of the mathematical foundations and core concepts that underpin both traditional and modern survival analysis approaches.

Chapter Overview

This chapter covers:

- Core mathematical functions in survival analysis and their relationships
- Types and properties of hazard functions
- Statistical handling of censored observations
- Likelihood construction for survival data
- Non-parametric estimation approaches
- Semi-parametric modeling with Cox proportional hazards
- Competing risks methodologies
- Limitations of classical methods

Understanding these foundations is crucial for properly interpreting results and developing innovative approaches to survival analysis, particularly when working with complex real-world datasets and modern machine learning techniques.

3.2 Mathematical Framework of Survival Analysis

Survival analysis is built upon a set of interrelated mathematical functions that provide different perspectives on time-to-event processes. Each function offers unique insights, but they are all mathematically connected, forming a coherent framework for analysis.

3.2.1 Core Functions and Their Definitions

Let T be a non-negative random variable representing the time until an event of interest occurs. The distribution of T can be characterized by several equivalent functions:

Survival Function

The **survival function**, denoted $S(t)$, represents the probability that the event has not occurred by time t:

$$S(t) = P(T > t) \tag{3.1}$$

Key properties:

- $S(0) = 1$ (all subjects event-free at the start)
- $\lim_{t\to\infty} S(t) = 0$ (eventually all subjects experience the event)
- $S(t)$ is monotonically decreasing (risk never decreases with time)
- $S(t)$ is right-continuous

The survival function provides a direct measure of the proportion of subjects who remain event-free over time, making it particularly useful for visualization and communicating results to non-specialists.

Cumulative Distribution Function

The **cumulative distribution function (CDF)**, denoted $F(t)$, represents the probability that the event has occurred by time t:

$$F(t) = P(T \le t) = 1 - S(t) \tag{3.2}$$

Key properties:

- $F(0) = 0$ (no events at the start)
- $\lim_{t\to\infty} F(t) = 1$ (eventually all subjects experience the event)
- $F(t)$ is monotonically increasing
- $F(t)$ is right-continuous

While the CDF is the standard representation for random variables in probability theory, survival analysis often focuses on the survival function instead, as it directly captures the concept of "surviving" beyond a specific time point.

Probability Density Function

The **probability density function (PDF)**, denoted $f(t)$, represents the instantaneous rate of event occurrence at exactly time t:

$$f(t) = \frac{d}{dt}F(t) = -\frac{d}{dt}S(t) \tag{3.3}$$

Key properties:

- $f(t) \geq 0$ for all t
- $\int_0^\infty f(t)dt = 1$
- $P(t_1 < T \leq t_2) = \int_{t_1}^{t_2} f(t)dt$

The PDF represents the unconditional instantaneous probability of the event occurring at time t. This differs from the hazard function, which considers the conditional probability given survival to time t.

Hazard Function

The **hazard function** (also called the hazard rate, intensity function, or failure rate), denoted $h(t)$, represents the instantaneous rate of event occurrence at time t, given survival up to time t:

$$h(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t | T \geq t)}{\Delta t} = \frac{f(t)}{S(t)} = -\frac{d}{dt}\log S(t) \tag{3.4}$$

Key properties:

- $h(t) \geq 0$ for all t
- $h(t)$ has no upper bound (can exceed 1)
- The shape of $h(t)$ reveals how risk changes over time
- Different from a probability (represents a rate)

The hazard function is perhaps the most informative representation for understanding the dynamics of risk over time. It answers the question: "Given that a subject has survived to time t, what is the instantaneous risk of experiencing the event?"

Cumulative Hazard Function

The **cumulative hazard function**, denoted $H(t)$, represents the accumulated risk up to time t:

$$H(t) = \int_0^t h(u)du = -\log S(t) \tag{3.5}$$

Key properties:

- $H(0) = 0$ (no accumulated risk at the start)
- $H(t)$ is monotonically increasing
- $H(t)$ has no upper bound
- $S(t) = \exp(-H(t))$

The cumulative hazard function is particularly useful for model estimation and comparison. It accumulates risk over time and has a direct relationship with the survival function.

3.2.2 Relationships Between Survival Functions

These core functions form an interconnected system where each function can be derived from the others. Figure 3.1 illustrates the key relationships between these functions.

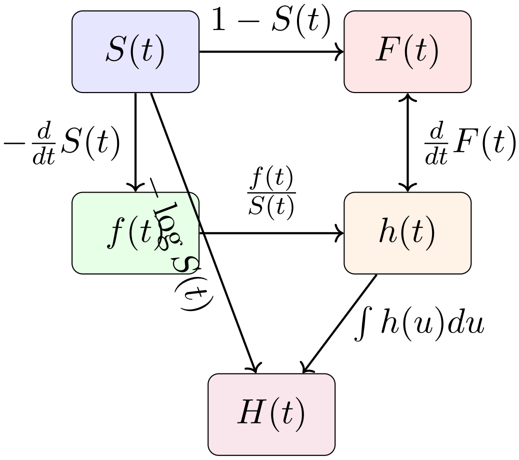


Figure 3.1: Relationships between the core functions in survival analysis. Each function can be derived from the others through the transformations shown on the arrows.

These mathematical relationships lead to several important practical formulas:

Key Mathematical Relationships

$$S(t) = \exp(-H(t)) = \exp\left(-\int_0^t h(u)du\right) \tag{3.6}$$

$$f(t) = h(t) \times S(t) \tag{3.7}$$

$$h(t) = \frac{f(t)}{S(t)} = -\frac{d}{dt}\log S(t) \tag{3.8}$$

Understanding these relationships is crucial for model development and implementation. For example:

- To derive a survival function from a specified hazard function, use $S(t) = \exp(-\int_0^t h(u)du)$
- To compute the density function when the hazard and survival are known, use $f(t) = h(t) \times S(t)$
- To estimate the hazard function from empirically derived survival estimates, use $h(t) = -\frac{d}{dt} \log S(t)$

The choice of which function to work with often depends on the specific application, computational considerations, and the specific modeling approach used.

3.3 Understanding Hazard Functions

The hazard function provides particularly valuable insights into the time-dependent risk profile. It helps identify the underlying mechanisms and patterns in time-to-event processes.

3.3.1 Interpretation of the Hazard Function

The hazard function $h(t)$ represents the instantaneous rate of failure at time t, given survival to time t. While it is not a probability (it can exceed 1), it can be interpreted as follows:

- For small time intervals Δt, the quantity $h(t)\Delta t$ approximates the probability of the event occurring in the interval $(t, t + \Delta t]$, given survival to time t
- Higher values of $h(t)$ indicate greater instantaneous risk
- The shape of $h(t)$ over time reveals how risk evolves throughout the process
- Comparing hazard functions between groups shows relative risk over time

Intuitive Interpretation

While mathematically defined as a conditional rate, the hazard function can be intuitively understood as:

- In medical contexts: The force of mortality or the propensity to fail at time t
- In engineering: The wear-out rate or proneness to failure at a specific point in a component's life
- In business: The customer's tendency to churn or cancel service at a given point in their lifecycle

The shape of the hazard function often provides clues about the underlying mechanisms driving the event process.

3.3.2 Common Hazard Patterns

Different processes exhibit characteristic hazard patterns that reflect their underlying mechanisms. Understanding these common patterns helps with model selection and interpretation.

Constant Hazard

The simplest hazard pattern is the constant hazard, where the instantaneous risk remains unchanged over time.

Constant Hazard Function

$$h(t) = \lambda \quad \text{for all } t \geq 0 \tag{3.9}$$

where $\lambda > 0$ is a constant rate parameter.
This leads to:

$$H(t) = \lambda t \tag{3.10}$$

$$S(t) = \exp(-\lambda t) \tag{3.11}$$

$$f(t) = \lambda \exp(-\lambda t) \tag{3.12}$$

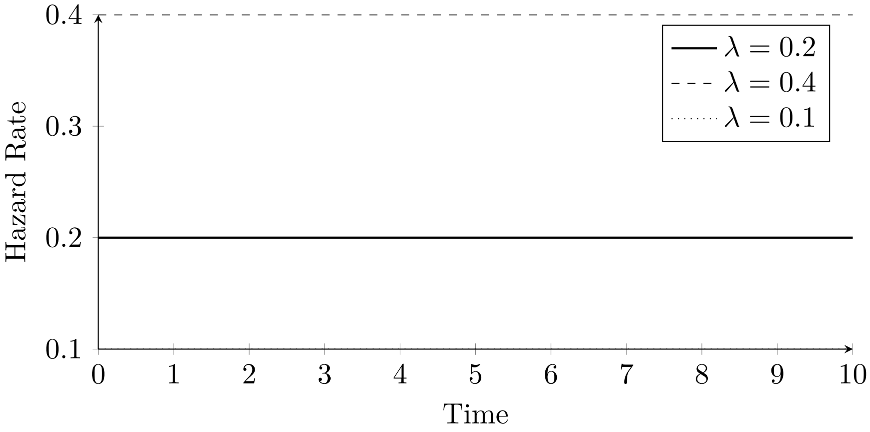


Figure 3.2: Constant hazard functions with different rate parameters λ. A constant hazard indicates that the instantaneous risk remains stable over time, regardless of how long the subject has survived.

A constant hazard results in an exponential distribution for the event times. This distribution has the "memoryless" property, meaning that the future risk is independent of the past survival history.

Applications of Constant Hazard Models

Constant hazard patterns are commonly observed in:

- Random electronic component failures during their useful life period
- Occurrence of accidents unrelated to age or wear
- Events driven by random environmental factors
- Radioactive decay processes
- Some chronic disease mortality in middle age ranges

The constant hazard assumption is often used as a baseline or null model against which to compare more complex hazard patterns.

Monotonically Increasing Hazard

Many real-world processes show increasing risk over time, reflecting aging, wear, or accumulating damage.

Weibull Increasing Hazard

A flexible model for increasing hazard is the Weibull distribution with shape parameter $k > 1$:

$$h(t) = \frac{k}{\lambda}\left(\frac{t}{\lambda}\right)^{k-1} \tag{3.13}$$

where $k > 1$ is the shape parameter and $\lambda > 0$ is the scale parameter.

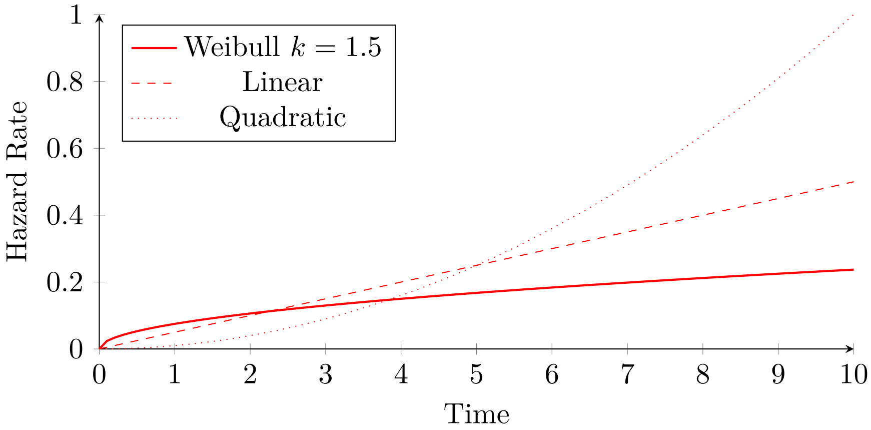


Figure 3.3: Monotonically increasing hazard functions. These patterns indicate that risk increases with time, which is characteristic of aging, wear-out, and progressive deterioration processes.

Increasing hazards are particularly common in:

- Aging processes in living organisms
- Mechanical wear and fatigue failures

- Progressive diseases where condition worsens over time
- Cancer recurrence risk after initial treatment
- Systems with cumulative damage

The rate of increase in the hazard reflects the speed of deterioration or aging in the underlying process.

Monotonically Decreasing Hazard

Some processes exhibit decreasing risk over time, often reflecting initial vulnerability followed by increasing stability.

Weibull Decreasing Hazard

The Weibull distribution with shape parameter $k < 1$ produces a decreasing hazard:

$$h(t) = \frac{k}{\lambda}\left(\frac{t}{\lambda}\right)^{k-1} \tag{3.14}$$

where $0 < k < 1$ is the shape parameter and $\lambda > 0$ is the scale parameter.

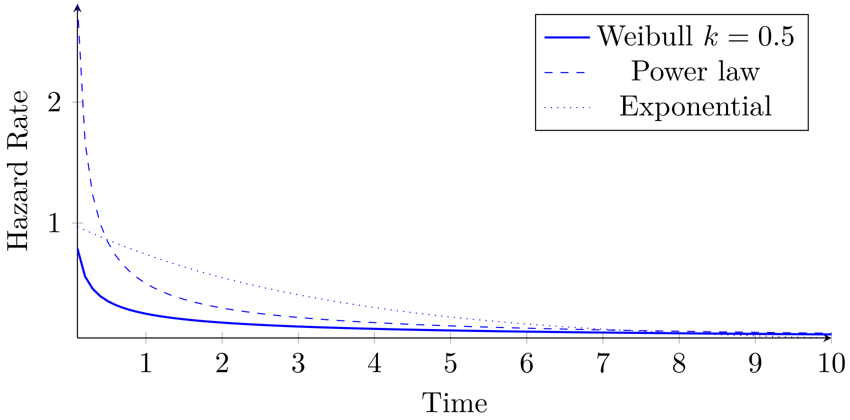


Figure 3.4: Monotonically decreasing hazard functions. These patterns show risk diminishing with time, which can indicate early vulnerability followed by strengthening or stabilization.

Decreasing hazards appear in:

- Infant mortality in humans and early failure in manufacturing
- Post-surgical infection risk, which is highest immediately after surgery
- Treatment efficacy that diminishes over time
- Immune response development after exposure
- Systems that strengthen or stabilize with time

Decreasing hazards often reflect selection processes where weaker subjects fail early, leaving a progressively more robust population.

Non-Monotonic Hazard Patterns

Many real-world processes exhibit more complex hazard patterns that combine elements of increasing and decreasing hazards.

Bathtub-Shaped Hazard The bathtub-shaped hazard curve starts high, decreases to a relatively stable level, and then increases again at later times.

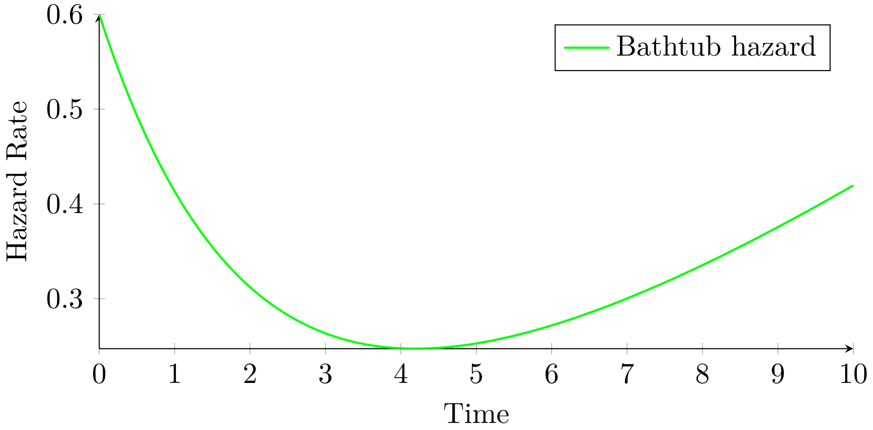


Figure 3.5: Bathtub-shaped hazard function showing high initial risk, followed by a period of relative stability, and ending with increasing risk. This pattern is common in human mortality and complex system lifecycles.

This pattern commonly appears in:

- Human mortality across the lifespan (infant mortality, stable adulthood, aging)
- Complete product lifecycles (early defects, useful life, wear-out)
- Systems with multiple failure modes operating at different time scales

Hump-Shaped Hazard Hump-shaped (or unimodal) hazards start low, increase to a peak, and then decrease again.

This pattern is common in:

- Post-surgical complications that peak during a critical recovery phase
- Treatment response patterns with a window of vulnerability
- Certain diseases with age-specific incidence peaks
- Log-normal failure distributions

3.3.3 Hazard Patterns in Real-World Applications

Recognizing hazard patterns in real data helps identify appropriate models and understand underlying mechanisms.

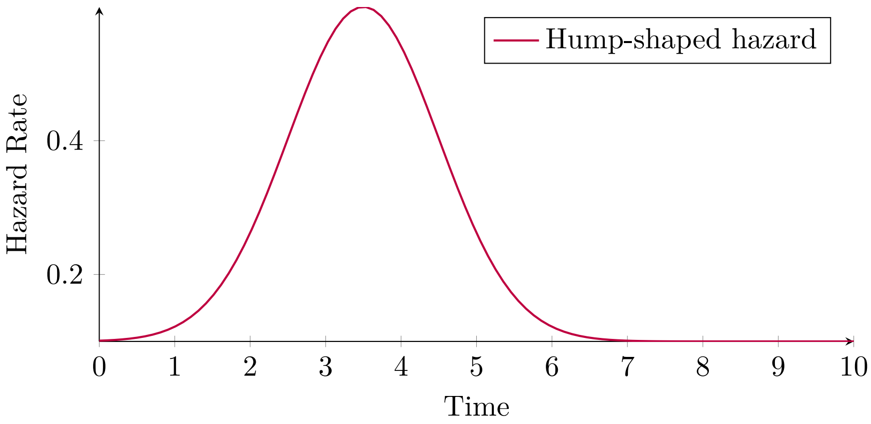


Figure 3.6: Hump-shaped hazard function showing risk that increases to a peak and then decreases. This pattern often occurs in scenarios with a critical time window of vulnerability.

Medical Applications

Different medical conditions exhibit characteristic hazard patterns:

- **Cancer progression:** Often shows increasing hazard as the disease advances and becomes more aggressive
- **Post-surgery recovery:** Typically shows high initial risk from complications, followed by declining risk
- **Infectious diseases:** May show hump-shaped hazards with peak risk during specific phases
- **Chronic conditions:** Often exhibit slowly increasing hazards reflecting progressive nature

Engineering Applications

Mechanical and electronic systems often follow well-established hazard patterns:

- **Electronic components:** Often follow the bathtub curve with three distinct phases:
  - Early failures due to manufacturing defects (decreasing hazard)
  - Random failures during useful life (constant hazard)
  - Wear-out failures in later life (increasing hazard)
- **Mechanical systems:** Frequently show increasing hazards due to wear, fatigue, or corrosion
- **Complex systems:** May exhibit mixture patterns reflecting multiple components and failure modes

Implications for Modeling

Understanding the hazard pattern of a process provides crucial guidance for model selection:

- Constant hazard processes are well-modeled by exponential distributions
- Monotonically changing hazards often fit well with Weibull distributions
- Non-monotonic hazards may require mixture models or more flexible parametric forms
- Complex patterns might benefit from non-parametric or semi-parametric approaches

Misspecification of the hazard pattern can lead to poor model fit and misleading conclusions.

3.4 Censoring in Survival Analysis

Censoringthe partial observation of event timesis a defining characteristic of survival data that requires specialized analytical methods. The detailed discussion of censoring, its types, and mechanisms can be found in Chapter 4.

3.4.1 Truncation vs. Censoring

It's important to distinguish between censoring and truncation, as they represent fundamentally different data limitations.

Truncation

Truncation occurs when subjects are systematically excluded from the sample based on their event times. Unlike censoring (which is partial observation), truncation means complete exclusion of certain subjects.

- **Left truncation:** Subjects are only included if their event time exceeds some threshold $(T > L)$
- **Right truncation:** Subjects are only included if their event time is less than some threshold $(T < R)$

Examples of truncation include:

- Studies of disease duration that recruit patients who already have the disease (left truncation)
- Studies based on death records, which only include those who have died (right truncation)
- Age-specific studies that only include subjects within certain age ranges

Truncation creates a biased sample and requires special methods to correct for the bias in estimation.

3.5 Likelihood Functions for Survival Data

The likelihood function serves as the foundation for parameter estimation in survival analysis. Due to censoring, the standard likelihood approach must be adapted to handle partially observed data.

3.5.1 General Principles of Likelihood Functions

In standard statistics, the likelihood function measures the consistency between observed data and a particular set of model parameters:

Standard Likelihood Function

For observations $X = \{x_1, x_2, \ldots, x_n\}$ and parameters θ:

$$\mathcal{L}(\theta|X) = p(X|\theta) = \prod_{i=1}^{n} p(x_i|\theta) \tag{3.15}$$

where $p(x_i|\theta)$ is the probability density function for observation x_i.

Maximum likelihood estimation (MLE) seeks the parameter values that maximize this likelihood:

$$\hat{\theta}_{MLE} = \arg\max_{\theta} \mathcal{L}(\theta|X) \tag{3.16}$$

For computational convenience, we often maximize the log-likelihood instead:

$$\hat{\theta}_{MLE} = \arg\max_{\theta} \log \mathcal{L}(\theta|X) = \arg\max_{\theta} \sum_{i=1}^{n} \log p(x_i|\theta) \tag{3.17}$$

3.5.2 Survival Likelihood with Censoring

In survival analysis, we observe a mixture of exact event times and censored observations. Each type contributes differently to the likelihood.

Survival Likelihood with Right Censoring

For a dataset with n subjects, each characterized by (y_i, δ_i) where y_i is the observed time and δ_i is the event indicator:

$$\mathcal{L}(\theta) = \prod_{i=1}^{n} [f(y_i|\theta)]^{\delta_i} [S(y_i|\theta)]^{1-\delta_i} \tag{3.18}$$

where:

- For observed events ($\delta_i = 1$): Contribution is $f(y_i|\theta)$, the density at the exact time
- For censored observations ($\delta_i = 0$): Contribution is $S(y_i|\theta)$, the probability of surviving beyond the censoring time

This formulation elegantly handles both types of observations by using the event indicator δ_i to select the appropriate term for each subject.

3.5.3 Alternative Formulations

Using the relationship $f(t) = h(t)S(t)$, we can rewrite the likelihood in terms of the hazard function:

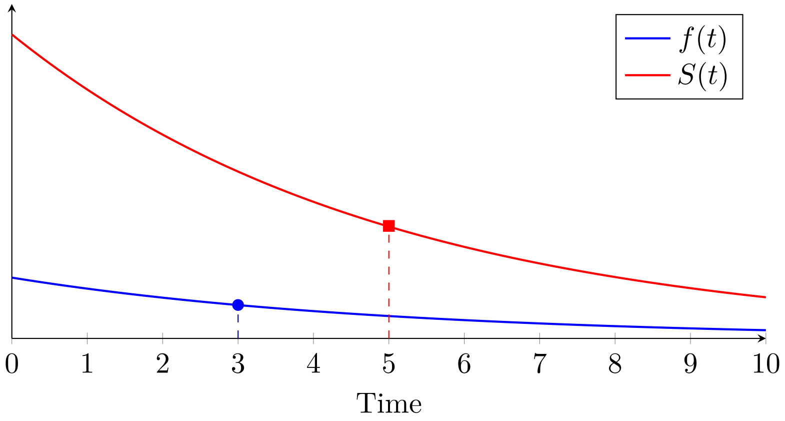


Figure 3.7: Likelihood contributions for different observation types. For observed events (blue), the contribution is the density $f(t)$ at the exact event time. For censored observations (red), the contribution is the survival function $S(t)$ at the censoring time.

Hazard-Based Likelihood

$$\mathcal{L}(\theta) = \prod_{i=1}^{n} [h(y_i|\theta)S(y_i|\theta)]^{\delta_i}[S(y_i|\theta)]^{1-\delta_i} \tag{3.19}$$

$$= \prod_{i=1}^{n} [h(y_i|\theta)]^{\delta_i} S(y_i|\theta) \tag{3.20}$$

Taking the logarithm and using $S(t) = \exp(-H(t))$, we obtain the log-likelihood:

Log-Likelihood for Survival Data

$$\log \mathcal{L}(\theta) = \sum_{i=1}^{n} \delta_i \log h(y_i|\theta) + \sum_{i=1}^{n} \log S(y_i|\theta) \tag{3.21}$$

$$= \sum_{i=1}^{n} \delta_i \log h(y_i|\theta) - \sum_{i=1}^{n} H(y_i|\theta) \tag{3.22}$$

This formulation is particularly useful for model development and implementation, as it separates the components related to observed events (first term) and the risk accumulated by all subjects (second term).

3.5.4 Handling Other Types of Censoring

While the likelihood for right-censored data is most common, other types of censoring require different likelihood formulations:

- **Left censoring:** For a left-censored observation at time c, the contribution is $F(c|\theta)$, the probability of the event occurring before time c
- **Interval censoring:** For an observation known to occur in the interval (L, R), the contribution is $F(R|\theta) - F(L|\theta)$, the probability of the event occurring within the interval

- **Mixed censoring:** When different observations have different censoring types, each contributes appropriately according to its type

General Likelihood with Multiple Censoring Types

For a dataset with right-censored, left-censored, and interval-censored observations:

$$\mathcal{L}(\theta) = \prod_{i \in \mathcal{R}} f(y_i|\theta) \prod_{j \in \mathcal{L}} F(y_j|\theta) \prod_{k \in \mathcal{I}} [F(R_k|\theta) - F(L_k|\theta)] \prod_{l \in \mathcal{C}} S(y_l|\theta) \tag{3.23}$$

where $\mathcal{R}$, $\mathcal{L}$, $\mathcal{I}$, and $\mathcal{C}$ represent the sets of exact, left-censored, interval-censored, and right-censored observations, respectively.

3.6 Non-Parametric Survival Estimation

Non-parametric methods make minimal assumptions about the underlying distribution of survival times, providing flexible descriptive analyses of survival patterns.

3.6.1 The Kaplan-Meier Estimator

The Kaplan-Meier estimator (also called the product-limit estimator) is the most widely used non-parametric approach for estimating the survival function from right-censored data.

Kaplan-Meier Estimator

For a dataset with observed times $t_1 < t_2 < \cdots < t_k$, the Kaplan-Meier estimator of the survival function is:

$$\hat{S}(t) = \prod_{i:t_i \leq t} \left(1 - \frac{d_i}{n_i}\right) \tag{3.24}$$

where:

- d_i is the number of events at time t_i
- n_i is the number of subjects at risk just before time t_i
- $\left(1 - \frac{d_i}{n_i}\right)$ represents the conditional probability of surviving through the interval containing t_i

The Kaplan-Meier estimator has several important properties:

- It is a step function that decreases only at observed event times
- It is the non-parametric maximum likelihood estimator for the survival function
- It properly accounts for right-censored observations
- It does not require specification of the underlying distribution

Confidence Intervals for Kaplan-Meier Estimates

Greenwood's formula provides a variance estimator for the Kaplan-Meier estimate:

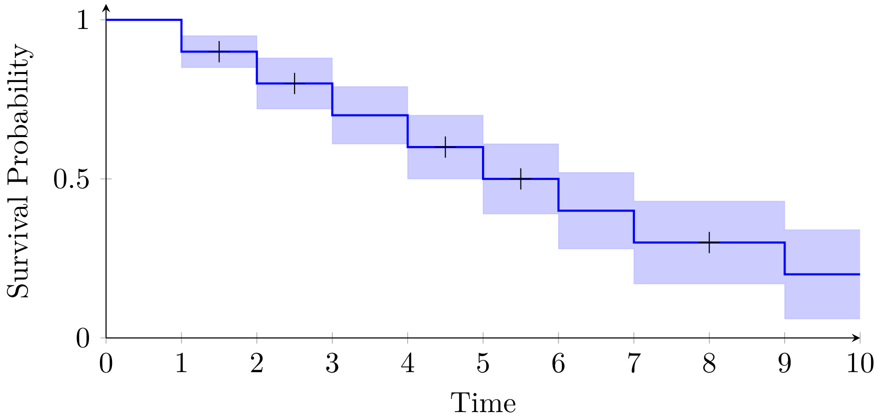


Figure 3.8: Kaplan-Meier estimate of the survival function with 95% confidence intervals. The step function decreases only at observed event times, while + marks indicate censored observations. The confidence intervals (shaded region) widen over time as information decreases.

Greenwood's Formula

$$\text{Var}[\hat{S}(t)] = [\hat{S}(t)]^2 \sum_{i:t_i \leq t} \frac{d_i}{n_i(n_i - d_i)} \tag{3.25}$$

A 95% confidence interval for the survival function at time t can be constructed as:

$$\hat{S}(t) \pm 1.96\sqrt{\text{Var}[\hat{S}(t)]} \tag{3.26}$$

Alternative formulations, such as the log-log transformation, are sometimes used to ensure the confidence limits remain within the range $[0, 1]$ and to improve the approximation for small samples.

3.6.2 Nelson-Aalen Estimator

While the Kaplan-Meier estimator focuses on the survival function, the Nelson-Aalen estimator provides a non-parametric estimate of the cumulative hazard function.

Nelson-Aalen Estimator

The Nelson-Aalen estimator of the cumulative hazard function is:

$$\hat{H}(t) = \sum_{i:t_i \leq t} \frac{d_i}{n_i} \tag{3.27}$$

where d_i and n_i are defined as in the Kaplan-Meier estimator.

The Nelson-Aalen estimator provides a basis for:

- Estimating the cumulative hazard directly
- Examining temporal patterns in hazard rates
- Checking model assumptions

- Deriving an alternative estimator of the survival function:

$$\hat{S}_{NA}(t) = \exp(-\hat{H}(t)) \tag{3.28}$$

3.6.3 Comparing Groups with Non-Parametric Methods

Non-parametric methods are particularly valuable for comparing survival between groups without assuming specific distributional forms.

Log-Rank Test

The log-rank test is the most widely used method for comparing two or more survival curves.

Log-Rank Test

The log-rank test compares observed versus expected events in each group under the null hypothesis of equal survival functions.
For G groups, the test statistic is:

$$\chi^2 = \sum_{j=1}^{G} \frac{(O_j - E_j)^2}{E_j} \tag{3.29}$$

where O_j is the observed number of events in group j, and E_j is the expected number under the null hypothesis.
Under the null hypothesis, this statistic follows a chi-square distribution with $G - 1$ degrees of freedom.

The log-rank test is most powerful for detecting differences when the hazard ratios between groups are constant over time (i.e., when the proportional hazards assumption holds).

Weighted Log-Rank Tests

Various weighted versions of the log-rank test give different emphasis to early or late differences in survival:

- **Gehan-Breslow test:** Weights by the number at risk, giving more emphasis to early differences

- **Tarone-Ware test:** Uses square root of the number at risk as weights

- **Peto-Peto test:** Weights by a modified survival estimate, reducing the influence of late differences where data are sparse

3.7 The Cox Proportional Hazards Model

The Cox proportional hazards model is the most widely used approach for analyzing the effect of covariates on survival time. It represents a semi-parametric approach that balances flexibility with the ability to perform covariate adjustment.

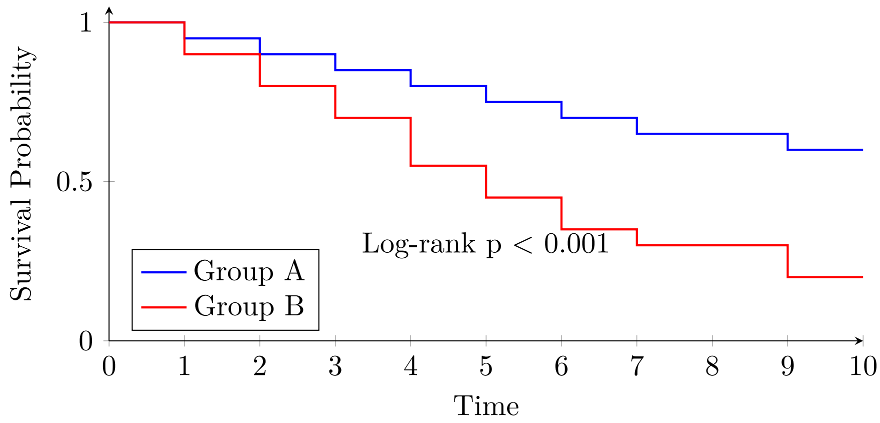


Figure 3.9: Kaplan-Meier curves comparing survival between two groups, with the log-rank test result indicating a significant difference.

3.7.1 Model Formulation

Cox Proportional Hazards Model

The Cox model specifies the hazard function for an individual with covariate vector X as:

$$h(t|X) = h_0(t) \exp(X\beta) \tag{3.30}$$

where:

- $h_0(t)$ is the baseline hazard function (left unspecified)
- X is the vector of covariates
- β is the vector of regression coefficients
- $\exp(X\beta)$ is the hazard ratio relative to baseline

This model has several notable features:

- It makes no assumptions about the shape of the baseline hazard $h_0(t)$
- The effect of covariates is multiplicative on the hazard scale
- The hazard ratio $\exp(X\beta)$ is constant over time (the proportional hazards assumption)
- The log hazard is a linear function of the covariates

3.7.2 Partial Likelihood Estimation

A key innovation of the Cox model is the partial likelihood approach, which allows estimation of β without specifying the baseline hazard $h_0(t)$.

Partial Likelihood

The partial likelihood for the Cox model is:

$$PL(\beta) = \prod_{i:\delta_i=1} \frac{\exp(X_i\beta)}{\sum_{j\in R(t_i)} \exp(X_j\beta)} \tag{3.31}$$

where:

- The product is over observed events (not censored observations)
- $R(t_i)$ is the risk set at time t_i (subjects still at risk just before t_i)
- The numerator represents the hazard for the subject who experienced the event
- The denominator sums the hazards across all subjects at risk at that time

The partial likelihood depends only on the order of events, not their exact timing, and eliminates the baseline hazard. Maximizing this partial likelihood yields consistent estimates of β under the proportional hazards assumption.

3.7.3 Interpretation of Parameters

The coefficients in the Cox model have a direct hazard ratio interpretation:

Interpreting Cox Model Coefficients

For a coefficient β_j corresponding to covariate X_j:

- $\exp(\beta_j)$ is the hazard ratio for a one-unit increase in X_j, holding other covariates constant
- $\beta_j > 0$ (or $\exp(\beta_j) > 1$) indicates increased hazard (worse survival)
- $\beta_j < 0$ (or $\exp(\beta_j) < 1$) indicates decreased hazard (better survival)
- $\beta_j = 0$ (or $\exp(\beta_j) = 1$) indicates no effect on hazard

Example: If $\beta_j = 0.693$ (so $\exp(\beta_j) = 2$), then a one-unit increase in X_j is associated with a doubling of the hazard rate.

3.7.4 Baseline Hazard Estimation

After estimating β, we can estimate the baseline hazard non-parametrically using the Breslow estimator:

Breslow Estimator

$$\hat{H}_0(t) = \sum_{i:t_i\leq t} \frac{d_i}{\sum_{j\in R(t_i)} \exp(X_j\hat{\beta})} \tag{3.32}$$

where d_i is the number of events at time t_i.

From this, we can derive the baseline survival function:

$$\hat{S}_0(t) = \exp(-\hat{H}_0(t)) \tag{3.33}$$

And the individual survival function for a subject with covariates X:

$$\hat{S}(t|X) = [\hat{S}_0(t)]^{\exp(X\hat{\beta})} \tag{3.34}$$

3.7.5 Extensions of the Cox Model

Several extensions address limitations of the basic Cox model:

Time-Dependent Covariates

The Cox model can accommodate covariates that change over time:

$$h(t|X(t)) = h_0(t)\exp(X(t)\beta) \tag{3.35}$$

Examples include:

- Biomarkers that change during follow-up
- Treatment status that changes over time
- Environmental exposures that vary temporally

Stratified Cox Model

When the proportional hazards assumption holds within groups but not across groups, the stratified Cox model allows different baseline hazards for each stratum:

$$h(t|X, s) = h_{0s}(t)\exp(X\beta) \tag{3.36}$$

where s indexes the stratum and $h_{0s}(t)$ is the stratum-specific baseline hazard.

Frailty Models

Frailty models account for unobserved heterogeneity and clustered data by introducing random effects:

$$h(t|X, Z) = h_0(t)\exp(X\beta + Z) \tag{3.37}$$

where Z represents unobserved frailty (random effect) that induces correlation within clusters.

3.8 Competing Risks Analysis

Competing risks occur when subjects can experience different types of events, where the occurrence of one event precludes observation of others. A detailed discussion of competing risks can be found in Chapter 4.

3.8.1 Framework and Terminology

In a competing risks setting, we observe:

- The time to the first event: $T = \min(T_1, T_2, \ldots, T_K)$
- The type of the first event: $J \in \{1, 2, \ldots, K\}$
- Possibly right-censoring: C

We observe (Y, δ, J) where:

- $Y = \min(T, C)$ is the observed time
- $\delta = I(T \leq C)$ indicates whether an event was observed
- J indicates the type of event (when $\delta = 1$)

3.8.2 Key Functions in Competing Risks

Cause-Specific Hazard

The cause-specific hazard for event type j is:

$$h_j(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t, J = j | T \geq t)}{\Delta t} \tag{3.38}$$

This represents the instantaneous risk of event type j at time t, given survival until time t.

Cumulative Incidence Function

The cumulative incidence function (CIF) for event type j is:

$$F_j(t) = P(T \leq t, J = j) = \int_0^t S(u) h_j(u) du \tag{3.39}$$

where $S(t) = \exp(-\sum_{k=1}^K \int_0^t h_k(u) du)$ is the overall survival function.
The CIF represents the probability of experiencing event type j by time t, accounting for the fact that other event types can occur.

3.8.3 Modeling Approaches for Competing Risks

Two main modeling approaches are used for competing risks:

Cause-Specific Hazards Approach

This approach models each cause-specific hazard separately, typically using Cox regression:

$$h_j(t|X) = h_{0j}(t) \exp(X\beta_j) \tag{3.40}$$

Each event type has its own baseline hazard and regression coefficients, reflecting potentially different risk factors. This approach is useful for understanding the effect of covariates on the instantaneous risk of each event type.

Fine-Gray Subdistribution Hazards Approach

The Fine-Gray approach models the subdistribution hazard directly:

$$\lambda_j(t|X) = \lambda_{0j}(t)\exp(X\gamma_j) \tag{3.41}$$

where the subdistribution hazard is defined as:

$$\lambda_j(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t, J = j | T \geq t \cup (T \leq t \cap J \neq j))}{\Delta t} \tag{3.42}$$

This approach models the CIF directly and is particularly useful for risk prediction and prognostic modeling.

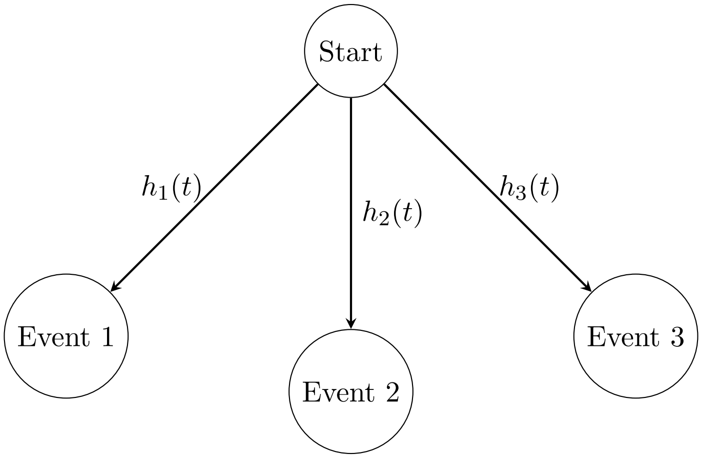


Figure 3.10: Example of a competing risks model. From the initial state, a subject can transition to one of several possible event states, each with its own hazard function.

3.9 Limitations of Classical Methods

While classical survival analysis methods have proven invaluable across many domains, they have significant limitations, particularly for complex data and modern applications.

3.9.1 Limitations of Parametric Models

Parametric models make strong assumptions about the underlying distribution of survival times:

- **Restrictive distributional assumptions:** Real-world data often do not conform to standard parametric families
- **Difficulty modeling complex hazard patterns:** Multi-modal or highly variable hazard shapes may not be well-approximated
- **Sensitivity to outliers:** Parameter estimates can be strongly influenced by extreme observations
- **Limited flexibility:** Fixed functional forms constrain how covariates affect survival

3.9.2 Limitations of the Cox Model

Despite its flexibility, the Cox model has several important limitations:

- **Proportional hazards assumption:** The hazard ratio between any two subjects is assumed constant over time, which is often violated
- **Linear functional form:** Covariates are assumed to have a log-linear effect on the hazard, which may not capture complex relationships
- **Limited ability to model interactions:** Interactions must be pre-specified and are typically limited to low-order terms
- **Challenges with high-dimensional data:** The traditional Cox model is not designed for settings where the number of predictors exceeds the number of observations

3.9.3 Limitations in Competing Risks Analysis

Traditional competing risks approaches have their own challenges:

- **Independence assumption:** Many methods implicitly assume independence between competing risks, which is often implausible
- **Difficulty modeling dependencies:** The joint distribution of event times is generally not identifiable from the observed data
- **Limited flexibility for risk-specific effects:** Covariates may have complex, non-linear effects that differ across event types
- **Challenges with time-dependent effects:** Time-varying effects are difficult to model in the competing risks framework

3.9.4 The Case for Advanced Methods

These limitations motivate the development of more flexible approaches, particularly those leveraging modern machine learning techniques:

- **Capturing non-linear relationships:** Methods that can automatically detect complex patterns in data
- **Handling high-dimensional data:** Approaches that work well with many predictors
- **Modeling interactions:** Techniques that can discover interactions without explicit specification
- **Relaxing distributional assumptions:** Methods that make fewer assumptions about the underlying data-generating process
- **Incorporating time-varying effects:** Approaches that naturally accommodate changes in effects over time

3.10 Modern Extensions of Classical Methods

Building on the foundation of classical survival analysis, modern approaches incorporate advanced statistical and machine learning techniques to address many of the limitations previously discussed.

3.10.1 Machine Learning Adaptations

Several machine learning methods have been adapted for survival analysis:

- **Random survival forests:** Ensemble methods that construct multiple survival trees and average their predictions, capturing non-linear effects and interactions
- **Survival support vector machines:** Adaptations of SVM methodology to handle censored data
- **Gradient boosting for survival:** Sequential construction of multiple weak learners to create a strong predictor
- **Neural network-based approaches:** Deep learning architectures specifically designed for time-to-event outcomes

3.10.2 Flexible Modeling Approaches

Several techniques enable more flexible modeling of survival relationships:

- **Spline-based methods:** Flexible modeling of baseline hazards and time-varying effects using spline functions
- **Penalized methods:** Regularization approaches for high-dimensional settings
- **Dynamic prediction:** Methods that update predictions as new information becomes available
- **Joint modeling:** Simultaneous modeling of longitudinal measurements and time-to-event outcomes

3.10.3 Causal Inference Methods

Causal inference approaches bring additional rigor to survival analysis in observational settings:

- **Marginal structural models:** Account for time-varying confounding through inverse probability weighting
- **G-computation:** Estimate causal effects through simulation of counterfactual outcomes
- **Instrumental variable methods for survival:** Address unmeasured confounding in survival settings

3.11 Transition to Deep Learning Approaches

The following chapters will build on these foundations to introduce modern deep learning approaches to survival analysis. These approaches combine the statistical rigor of traditional survival methods with the flexibility and power of neural networks, enabling:

- Automatic feature learning from complex, high-dimensional data
- Capture of intricate non-linear relationships without explicit specification
- Integration of multiple data modalities (structured, text, images)

- Flexible modeling of time-varying effects
- Handling of competing risks and complex event patterns

Looking Ahead

In the next chapter, we will examine Deep Survival Machines (DSM), a framework that combines neural networks with mixture distribution modeling to provide flexible, interpretable survival predictions. This approach addresses many of the limitations of classical methods while maintaining the probabilistic framework necessary for valid survival analysis.

3.12 Summary

This chapter has covered the fundamental mathematical and statistical foundations of survival analysis:

- The core functions that characterize time-to-event data: survival function, hazard function, and their relationships
- Common hazard patterns and their practical interpretations
- The statistical handling of censored observations through specialized likelihood functions
- Non-parametric approaches, particularly the Kaplan-Meier estimator
- The semi-parametric Cox proportional hazards model for covariate adjustment
- Competing risks methodology for scenarios with multiple event types
- Limitations of classical approaches and the motivation for modern methods

Understanding these foundations is essential for developing and applying advanced survival analysis methods. The subsequent chapters will build on this foundation to introduce deep learning approaches that retain the statistical validity of traditional methods while leveraging the flexibility and power of neural networks.

Chapter 4

Censoring and Competing Risks

4.1 Understanding Censoring

Censoring occurs when we have incomplete information about a subject's survival time. It is a fundamental concept in survival analysis as most real-world studies involve some form of censoring. Without accounting for censoring, survival estimates would be biased.

4.1.1 Types of Censoring

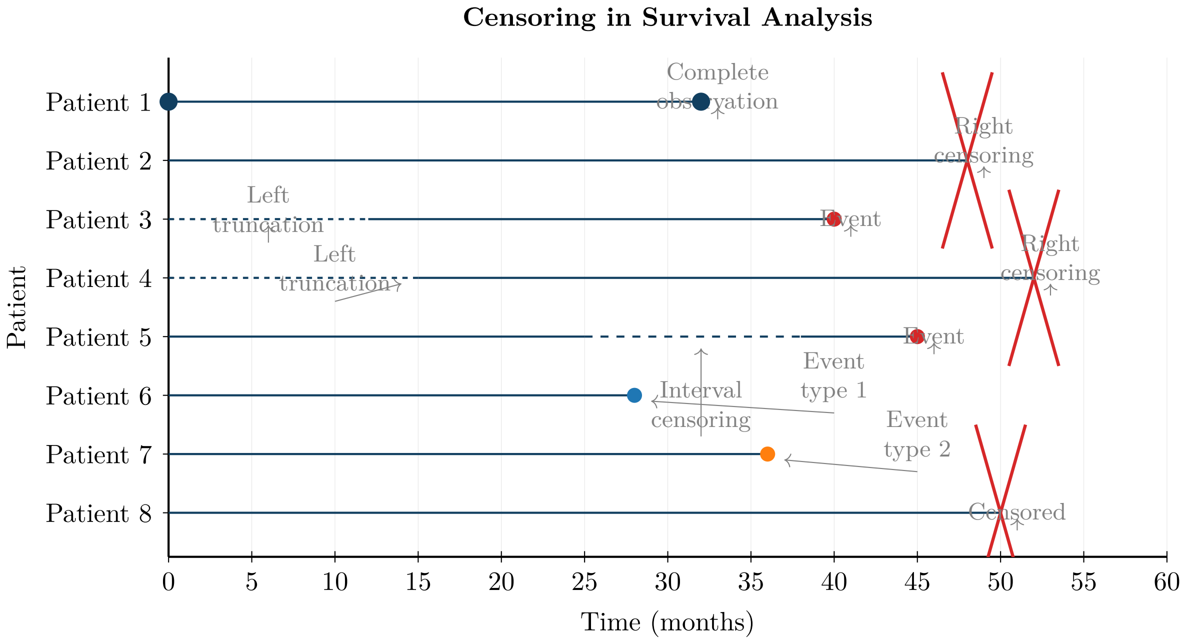


Figure 4.1: Different types of censoring in survival analysis and competing risks. The plot shows various scenarios including complete observation, right censoring, left truncation, interval censoring, and competing risks. Solid lines represent observed follow-up, dashed lines represent unobserved time, and markers indicate events or censoring.

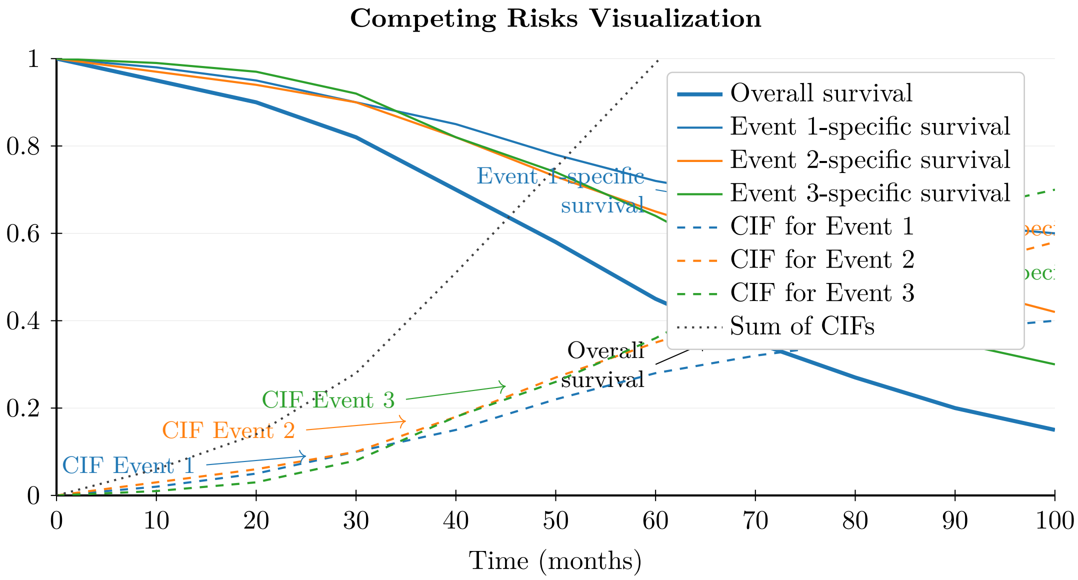


Figure 4.2: Visualization of competing risks: survival curves and cumulative incidence functions (CIFs). The overall survival (thick blue line) decreases more rapidly than any cause-specific survival curve. The dashed lines show the cumulative incidence functions for each competing event. Note that the sum of all CIFs (dotted line) can exceed 1.0 when viewing causes independently, illustrating why proper competing risks analysis is important.

Censoring Types

Censoring occurs when the event time is not precisely observed but is known to occur within a certain time range. The three main types of censoring are:

- **Right censoring:** The most common type, occurs when a subject has not experienced the event of interest by the end of the study, is lost to follow-up, or withdraws from the study. Right censoring gives us a lower bound on the true time-to-event.
- **Left censoring:** Occurs when the event is known to have occurred before the first observation time, but the exact time is unknown. We only know that $T < t_{first}$.
- **Interval censoring:** Occurs when we only know that the event occurred within a certain time interval, but not the exact time. This happens when subjects are assessed periodically, and the event is detected at a follow-up visit.

The Kaplan-Meier estimator and Cox proportional hazards model both handle right censoring. Left truncation can be accommodated by modifying the risk sets in these methods. Interval censoring requires specialized techniques like the Turnbull estimator or parametric models.

4.1.2 Time-to-Event Data Visualization

The following visualizations illustrate time-to-event data for individual subjects and specific examples of different censoring types:

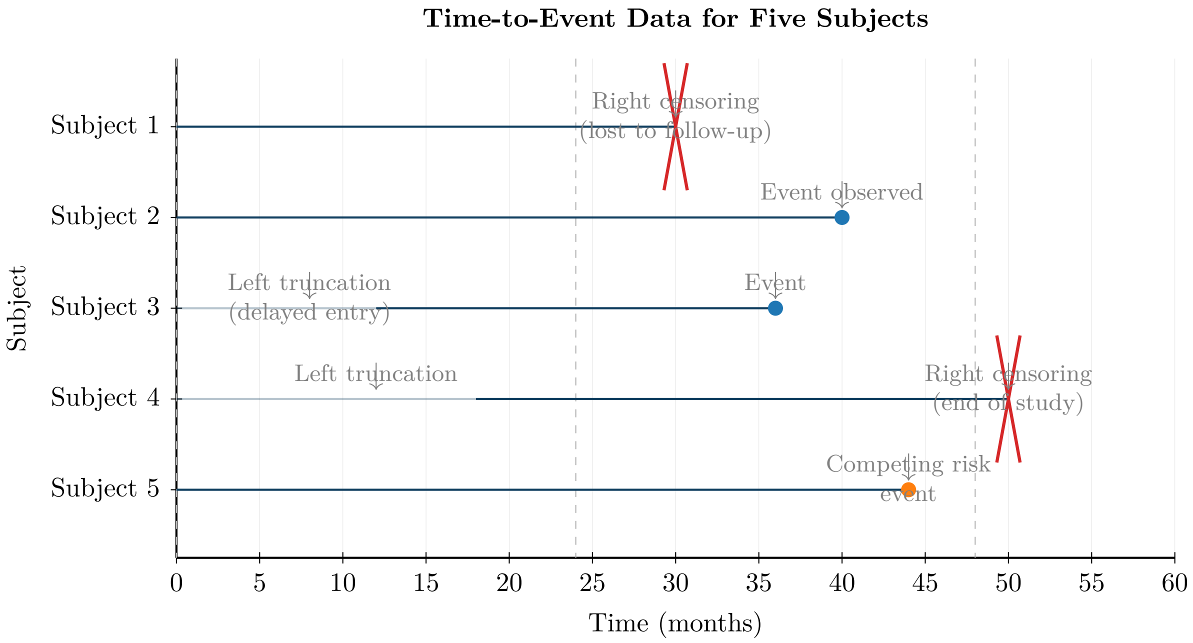


Figure 4.3: Time-to-event data for five subjects showing various censoring and event patterns. Subject 1 experiences right censoring due to loss to follow-up. Subject 2 has an observed event. Subject 3 enters the study late (left truncation) and experiences an event. Subject 4 enters late and is right-censored at the end of the study. Subject 5 experiences a competing risk event.

4.1.3 Clinical Study Visualization

The following visualization illustrates real-world patient follow-up patterns in a clinical trial setting:

4.1.4 Censoring Mechanisms and Their Implications

The mechanism that causes censoring has important implications for statistical modeling and the validity of results. Different types of censoring mechanisms create different analytical challenges.

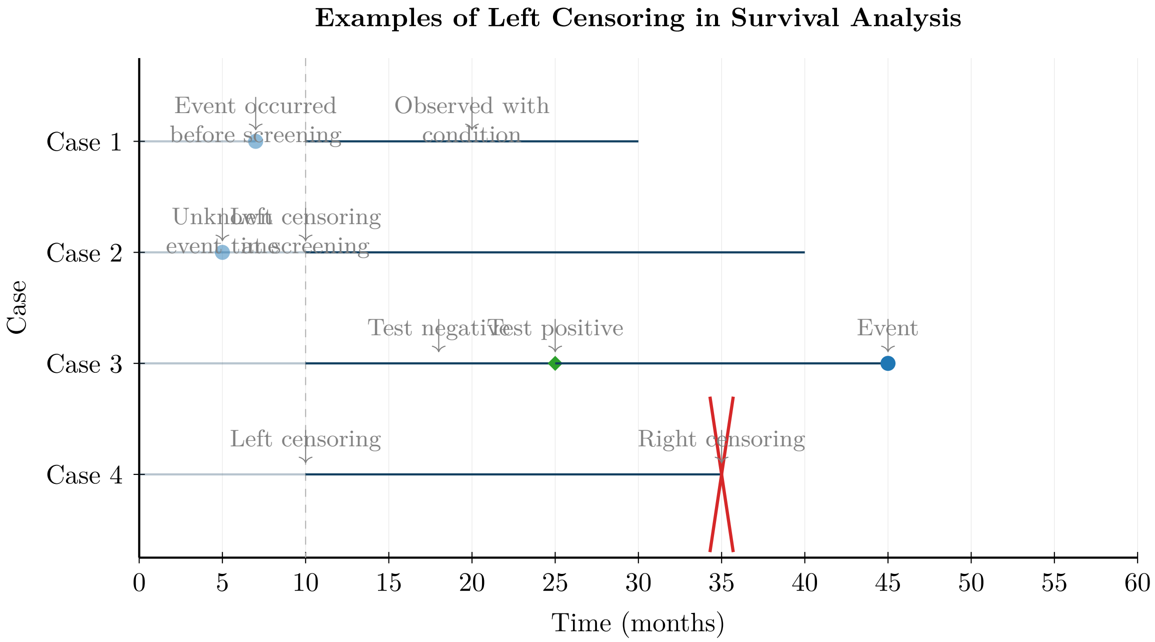


Figure 4.4: Examples of left censoring in survival analysis. Case 1 shows an event known to have occurred before screening, with the exact time unknown. Case 2 shows a condition detected at the initial screening (left censored). Case 3 shows delayed detection where a test initially misses the condition but later detects it. Case 4 shows both left and right censoring in the same subject. For left censoring, we only know that the event occurred before a certain time point, in contrast to right censoring where we know it occurred after.

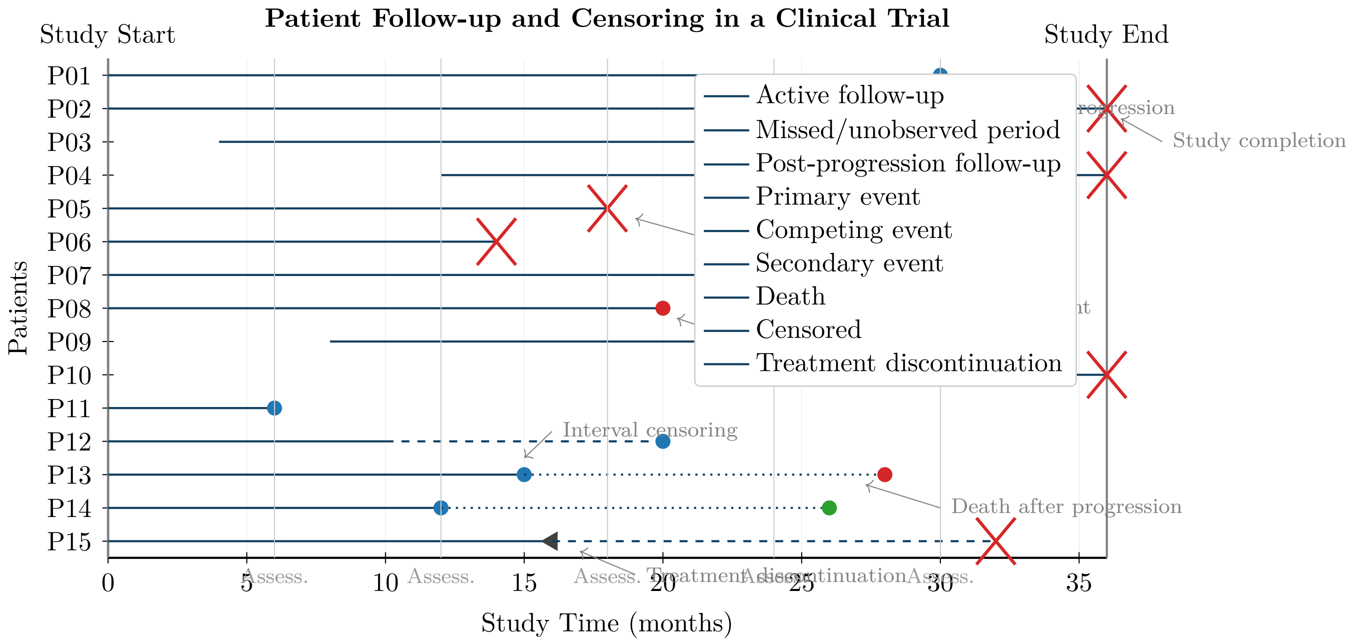


Figure 4.5: Visualization of patient follow-up in a 36-month clinical trial. The plot shows various scenarios including complete follow-up, early/late enrollment, loss to follow-up, treatment discontinuation, competing events, and interval censoring. Vertical lines represent scheduled assessment visits at 6-month intervals. This visual representation illustrates the complexity of longitudinal data in clinical studies and the various types of censoring that must be accounted for in survival analysis.

Censoring Mechanisms

Three types of censoring mechanisms are distinguished:

- **Missing Completely At Random (MCAR):** Censoring is independent of both observed and unobserved factors. Examples include administrative end of study or random equipment failure in monitoring devices.

$$P(C = c|T = t, X = x) = P(C = c) \tag{4.1}$$

- **Missing At Random (MAR):** Censoring depends on observed covariates but not on the event time itself. Examples include study withdrawal related to observed side effects.

$$P(C = c|T = t, X = x) = P(C = c|X = x) \tag{4.2}$$

- **Missing Not At Random (MNAR):** Censoring depends on the unobserved event time. Examples include patients dropping out because of health deterioration not captured in observations.

$$P(C = c|T = t, X = x) \neq P(C = c|X = x) \tag{4.3}$$

where T is the event time, C is the censoring time, and X represents covariates.

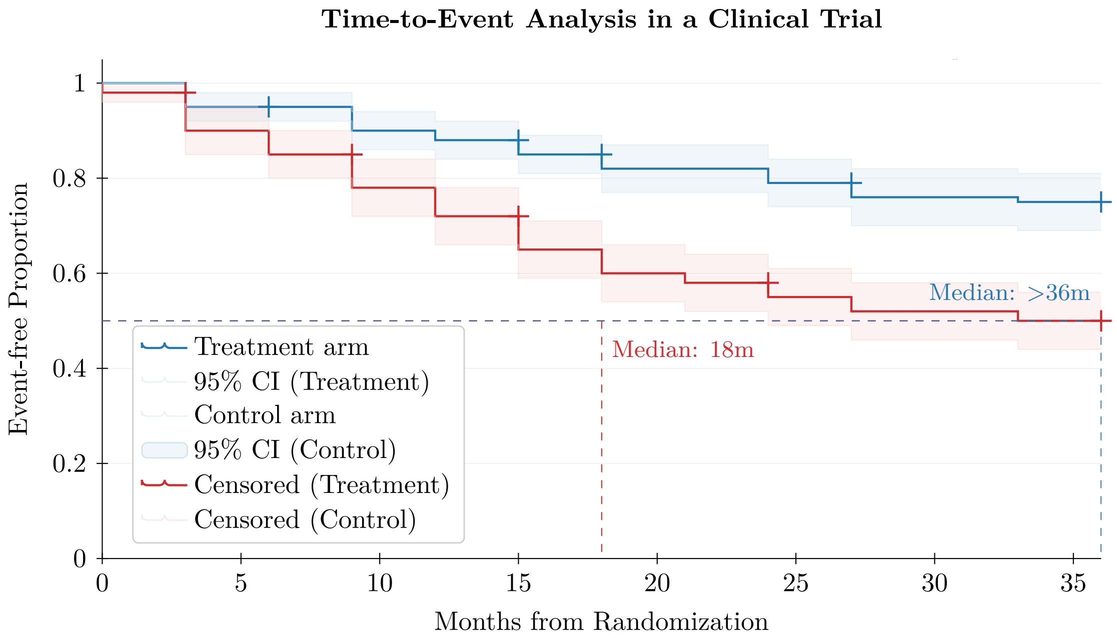


Figure 4.6: Kaplan-Meier event-free survival curves comparing treatment and control arms in a clinical trial with 36-month follow-up. Shaded areas represent 95% confidence intervals, and '+' marks indicate censored observations. The treatment arm shows a significant improvement in event-free survival with a hazard ratio of 0.57 (95% CI: 0.41-0.78) and a log-rank p-value of 0.0015. The median event-free survival is 18 months in the control arm and has not been reached in the treatment arm at the 36-month study endpoint. The number of patients at risk is shown below the graph.

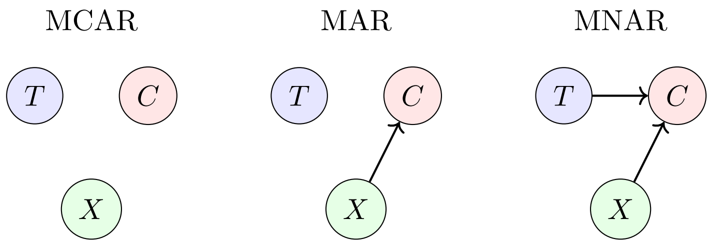


Figure 4.7: Directed acyclic graphs illustrating different censoring mechanisms. Arrows indicate dependencies between variables. Under MCAR, censoring is independent of other variables. Under MAR, censoring depends on observed covariates. Under MNAR, censoring depends on the unobserved event time.

Non-informative vs. Informative Censoring

A related distinction is between:

- **Non-informative censoring:** The censoring process provides no information about the event time beyond what is available in the observed covariates (equivalent to MCAR or MAR)
- **Informative censoring:** The censoring process itself provides information about the event time (equivalent to MNAR)

Most standard survival methods assume non-informative censoring. When censoring is informative, more complex joint modeling of the censoring and event processes may be required.

For survival analysis methods to produce valid results, censoring must typically be:

- **Independent/non-informative:** The censoring mechanism should be unrelated to the event process. If subjects who are more likely to experience the event are also more likely to be censored, we have informative censoring, which can bias results.
- **Random:** The distribution of censoring times should be random and not systematically related to subject characteristics or study conditions.

These assumptions should be critically evaluated in any survival analysis. When they're violated, sensitivity analyses or models that account for informative censoring may be needed.

4.2 Competing Risks

Competing risks occur when subjects can experience multiple types of events, and the occurrence of one event precludes the occurrence of other events or changes their probability. Traditional survival analysis methods that treat competing events as censored can lead to biased estimates.

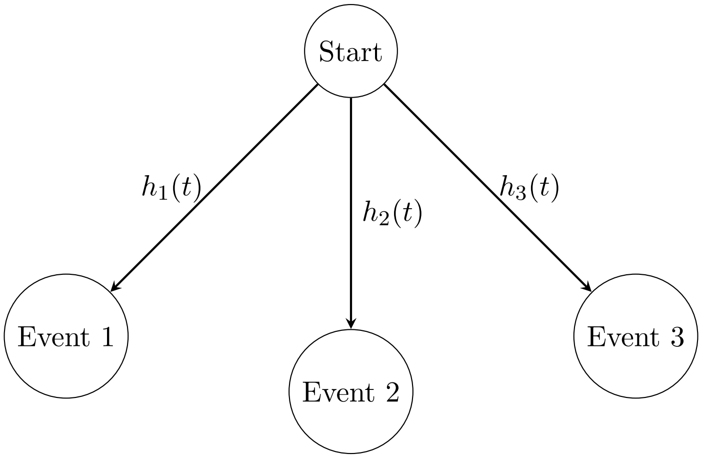


Figure 4.8: Competing risks framework. From the initial state, a subject can transition to one of several possible event states, each with its own hazard function. Once one event occurs, the subject is no longer at risk for the other events.

4.2.1 Analyzing Competing Risks

When analyzing data with competing risks, we consider:

- **Cause-specific hazards:** The instantaneous rate of occurrence of a specific event type among those still at risk.
- **Cumulative Incidence Function (CIF):** The probability of experiencing a specific event type before time t and before experiencing any competing event.
- **Subdistribution hazard:** The instantaneous rate of occurrence of a specific event type among those who have not experienced that specific event yet (including those who have experienced competing events).

Unlike in the standard survival setting where the survival function is directly related to the hazard, in competing risks, the relationship between cause-specific hazards and CIFs is more complex. A reduction in one cause-specific hazard may not necessarily translate to an increase in the corresponding CIF due to the interplay with competing events.

4.2.2 Modeling Approaches for Competing Risks

- **Cause-specific Cox models:** Fit separate Cox models for each cause, treating competing events as censored.
- **Fine and Gray model:** Models the subdistribution hazard directly, allowing for direct assessment of covariate effects on the CIF.
- **Multi-state models:** Consider transitions between different states (e.g., healthy, diseased, dead) and can model complex disease processes.
- **Joint modeling:** Simultaneously model multiple event types, accounting for their dependencies.
- **Neural approaches (e.g., MENSA):** Use neural networks to model complex relationships between covariates and multiple event types.

The choice of method depends on the research questionwhether the interest is in etiology (cause-specific hazards) or absolute risk prediction (CIFs).

Examples of Competing Risks

- In medical studies: Death from cancer, heart disease, or other causes
- In engineering: Component failure due to wear, corrosion, or manufacturing defect
- In business: Customer churn due to price, service quality, or competitor offers
- In employment: Job termination due to retirement, new job opportunity, or layoff

Important Note on Competing Risks

A critical point in competing risks analysis is that $1 - S_j(t) \neq F_j(t)$, where $S_j(t)$ is the cause-specific survival function obtained by treating other event types as censored.
The Kaplan-Meier estimator applied to a single event type (treating others as censored) overestimates the probability of that event in the presence of competing risks.

Chapter 5

Deep Survival Machines

5.1 Introduction to Deep Survival Machines

Deep Survival Machines (DSM) (Nagpal, Li, and Dubrawski, 2021) represents a paradigm shift in survival analysis, introducing a novel approach that bridges the gap between traditional statistical methods and modern deep learning. This chapter explores how DSM combines the strengths of neural networks with parametric survival modeling to overcome the limitations of both conventional approaches (Cox, 1972; Kaplan and Meier, 1958) and earlier neural network adaptations for survival analysis (Katzman et al., 2018; Chapfuwa et al., 2018).

Chapter Overview

This chapter covers:

- The limitations of traditional survival analysis methods
- Complex hazard patterns in real-world survival data
- The conceptual innovation of the DSM mixture approach
- Mathematical foundations of DSM
- Implementation details and practical considerations
- Advantages of DSM over alternative methods
- Applications and results in real-world scenarios

5.2 Limitations of Traditional Survival Analysis Methods

Despite the theoretical foundations discussed in the previous chapter, traditional survival analysis methods face significant limitations when applied to complex, high-dimensional real-world data. Understanding these limitations provides the motivation for developing more advanced approaches like Deep Survival Machines.

5.2.1 Limitations of Non-parametric Methods

The Kaplan-Meier estimator and other non-parametric methods, while making minimal assumptions about the underlying distribution, have several important constraints:

- **Limited covariate handling:** They cannot effectively incorporate continuous covariates or high-dimensional feature spaces
- **Reliance on stratification:** Covariate adjustment is limited to stratification by categorical variables, which quickly becomes impractical with multiple variables
- **No extrapolation:** Predictions cannot extend beyond the largest observed event time in the dataset
- **Curse of dimensionality:** Performance degrades in high-dimensional settings
- **Inefficient use of data:** Cannot leverage patterns across different strata

5.2.2 Limitations of the Cox Proportional Hazards Model

While the Cox model addresses some limitations of non-parametric methods by enabling covariate adjustment, it introduces its own constraints:

- **Proportional hazards assumption:** Assumes that the hazard ratio between any two subjects remains constant over timean assumption frequently violated in practice
- **Linear effects:** Assumes that covariates have a log-linear effect on the hazard, which may not capture complex non-linear relationships
- **Baseline hazard challenges:** The baseline hazard estimation is often unstable, especially with sparse data at longer follow-up times
- **Time-varying effects:** Accommodating time-varying covariate effects requires complex model extensions
- **Limited interaction modeling:** Capturing complex interactions between covariates requires explicit specification

5.2.3 Limitations of Early Neural Network Adaptations

Early efforts to apply neural networks to survival analysis, while promising, retained significant limitations:

- **Specialized architectures:** Required complex, specialized architectures to handle censored data
- **Proportional hazards heritage:** Many approaches still relied on the proportional hazards assumption
- **Black-box nature:** Limited interpretability and uncertainty quantification
- **Numerical challenges:** Suffered from instability in optimization and prediction

- **Distribution constraints:** Often still constrained to specific parametric forms or semi-parametric assumptions

These limitations highlight the need for a more flexible approach that can leverage the representational power of neural networks while respecting the unique characteristics of survival data.

5.3 Complex Hazard Patterns in Real-World Data

Real-world survival data often exhibits complex hazard patterns that cannot be adequately captured by traditional models, which typically assume simple monotonic hazard functions.

5.3.1 Multi-Modal Hazard Functions

Many real-world processes, particularly in medicine, show hazard patterns with multiple modes or complex shapes:

- **Early high risk followed by plateau:** Common in post-surgical scenarios where immediate complications dominate early risk
- **Delayed risk that increases over time:** Seen in certain cancers where recurrence risk grows after an initial treatment response
- **Multiple risk peaks (biphasic patterns):** Observed in diseases with distinct phases, such as certain autoimmune conditions
- **Risk patterns that vary by subpopulation:** Different risk profiles for different patient subgroups, even after controlling for observable covariates

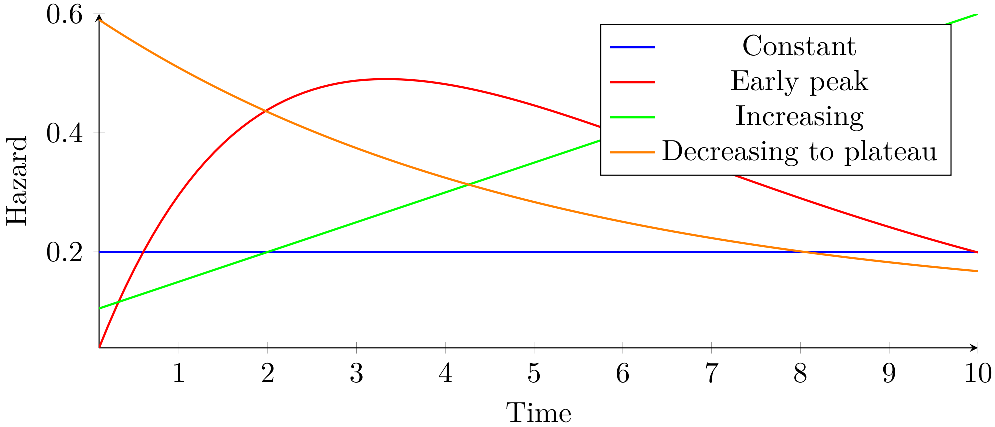


Figure 5.1: Examples of different hazard patterns observed in real-world survival data. Traditional models often assume a single pattern (such as constant or monotonically increasing/decreasing), but real processes frequently exhibit more complex behavior.

5.3.2 The Need for Uncertainty Quantification

Another critical limitation of many survival analysis methods is the lack of robust uncertainty quantification, which is essential in high-stakes decision-making contexts.

Clinical Decision-Making with Uncertainty

Consider a physician deciding between aggressive and conservative treatment for a cancer patient:

- A point estimate suggesting 60% 5-year survival probability provides limited guidance
- Knowing the confidence interval is 40-80% indicates substantial uncertainty
- This uncertainty information might lead to additional testing or consideration of patient preferences
- Without proper uncertainty quantification, decision-makers may have false confidence in predictions

Two fundamental types of uncertainty need to be addressed:

- **Epistemic uncertainty:** Uncertainty due to model limitations and finite datacan theoretically be reduced with more data or better models
- **Aleatoric uncertainty:** Inherent randomness in the process being modeledcannot be reduced with more data and represents genuine outcome variability

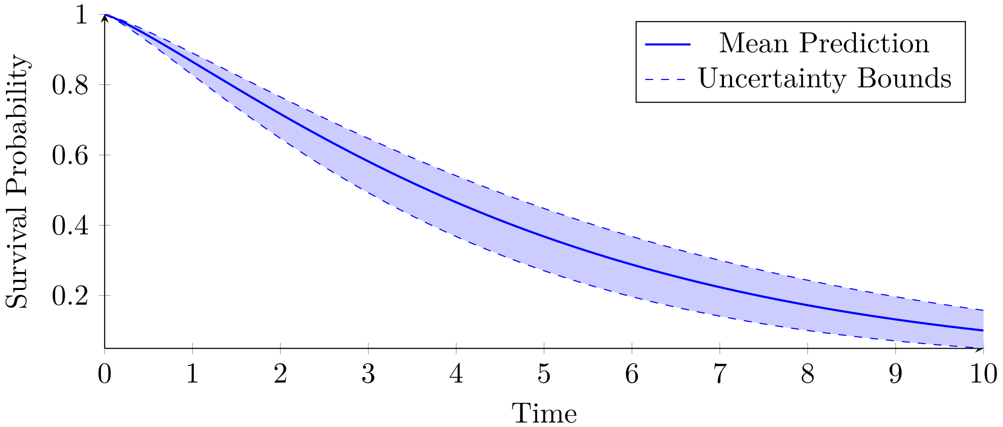


Figure 5.2: Survival curve with uncertainty bounds. The shaded region represents the uncertainty in the survival estimates, which increases over time as less information is available. This visualization provides decision-makers with both the point estimate and a measure of confidence in that estimate.

These complex hazard patterns and the need for robust uncertainty quantification motivate the development of more flexible and expressive approaches to survival analysis.

5.4 Deep Survival Machines: Core Conceptual Innovation

Deep Survival Machines introduces a novel approach to survival analysis that addresses the limitations of traditional methods while leveraging the representational power of neural networks.

5.4.1 The Mixture Distribution Approach

The central innovation of DSM is to model survival as a mixture of parametric distributions:

- **Mixture composition:** The survival function is modeled as a weighted sum of multiple parametric survival functions
- **Component diversity:** Each component can capture a different risk pattern or represent a different subpopulation
- **Neural parameter mapping:** Neural networks learn to map from input features to distribution parameters and mixture weights
- **Flexible hazard representation:** The mixture approach can represent virtually any smooth hazard function
- **Natural uncertainty quantification:** The mixture variance provides a measure of predictive uncertainty

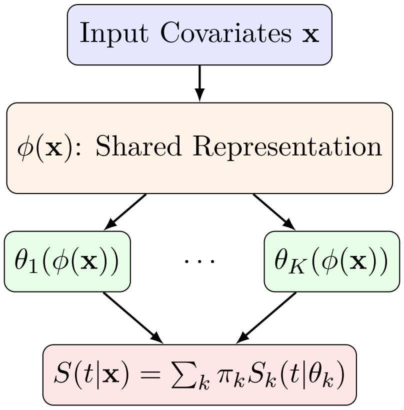

Input Covariates $\mathbf{x}$
$\phi(\mathbf{x})$: Shared Representation
$\theta_1(\phi(\mathbf{x}))$
$\cdots$
$\theta_K(\phi(\mathbf{x}))$
$S(t|\mathbf{x}) = \sum_k \pi_k S_k(t|\theta_k)$

Figure 5.3: Conceptual architecture of Deep Survival Machines. Input covariates are processed through a shared representation layer, which then feeds into parameter networks for each mixture component. The final survival function is a weighted combination of the component survival functions.

5.4.2 Architectural Components

The DSM architecture consists of three main components that work together to create a flexible and expressive survival model:

Representation Network

The representation network transforms raw covariates into a latent feature representation:

- Can use any neural network architecture appropriate for the data type (MLP for tabular data, CNN for images, transformers for sequential data, etc.)
- Learns task-relevant feature extraction and transformation
- Enables automatic feature learning and extraction of complex patterns
- Creates a shared representation that feeds into the parameter networks

Mixture Model

The mixture model combines multiple parametric survival distributions:

- Typically uses 2-8 components, depending on data complexity
- Components are usually from the same parametric family (e.g., all Weibull) but with different parameters
- Models different risk patterns and subpopulations implicitly discovered from the data
- Creates a flexible composition that can represent complex hazard shapes

Parameter Networks

Parameter networks map from the shared representation to the parameters of each distribution component:

- Separate networks for each parameter type (e.g., shape, scale) and each component
- Include constraints to ensure valid parameter ranges (e.g., positive shape parameters)
- Learn adaptive mixture weights that determine the contribution of each component
- Enable personalized parameter estimation based on individual covariates

5.5 Key Innovations of Deep Survival Machines

Deep Survival Machines introduces several key innovations that distinguish it from both traditional survival methods and earlier neural network approaches.

5.5.1 End-to-End Learning

Unlike many traditional survival models that separate feature engineering from survival modeling, DSM enables end-to-end learning:

- Joint optimization of representation learning and survival parameter estimation
- No need for separate feature engineering or transformation steps
- Gradient-based optimization of all model components
- Representation automatically adapts to the survival prediction task

5.5.2 Flexible Hazard Modeling

The mixture approach enables modeling of complex hazard patterns without restrictive assumptions:

- No proportional hazards assumption required
- Can model virtually any survival distribution
- Automatically captures non-linear relationships between covariates and survival
- Adapts to the underlying data-generating process

5.5.3 Uncertainty Quantification

DSM provides natural uncertainty quantification through its mixture formulation:

- The mixture variance provides a measure of predictive uncertainty
- Different components can represent different risk scenarios
- Uncertainty can be visualized through prediction intervals
- Both epistemic and aleatoric uncertainty are captured

5.5.4 Risk Prediction Capabilities

Unlike many survival models that focus on relative risks, DSM provides comprehensive risk assessment:

- Direct probability outputs (not just relative risks)
- Time-dependent risk assessment at any time point
- Personalized survival curves for individual subjects
- Risk stratification based on predicted outcomes

5.6 Mathematical Foundation of Deep Survival Machines

To fully understand DSM, we need to examine its mathematical formulation, starting with the basic survival analysis functions and extending to the mixture framework.

5.6.1 Survival Analysis Fundamentals: A Brief Recap

As discussed in the previous chapter, for a random variable T representing survival time, the key probability functions are:

Core Survival Functions

$$\text{Survival function: } S(t) = P(T > t) \tag{5.1}$$

$$\text{CDF: } F(t) = P(T \leq t) = 1 - S(t) \tag{5.2}$$

$$\text{PDF: } f(t) = \frac{d}{dt}F(t) = -\frac{d}{dt}S(t) \tag{5.3}$$

$$\text{Hazard function: } h(t) = \frac{f(t)}{S(t)} = -\frac{d}{dt}\log S(t) \tag{5.4}$$

$$\text{Cumulative hazard: } H(t) = \int_0^t h(u)du = -\log S(t) \tag{5.5}$$

These functions are interrelated, and each provides a different perspective on the time-to-event process. DSM builds on this foundation by introducing a mixture framework.

5.6.2 The Mixture Framework

The fundamental insight of DSM is to model the survival distribution as a mixture of parametric distributions:

DSM Mixture Survival Function

$$S(t|\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) S_k(t|\mathbf{x}) \tag{5.6}$$

where:

- $S(t|\mathbf{x})$ is the overall survival function given covariates $\mathbf{x}$
- $S_k(t|\mathbf{x})$ is the survival function for the k-th component
- $\pi_k(\mathbf{x})$ is the mixture weight for the k-th component
- $\sum_{k=1}^{K} \pi_k(\mathbf{x}) = 1$ and $\pi_k(\mathbf{x}) \geq 0$ for all k

From this mixture survival function, we can derive the corresponding density and hazard functions:

DSM Mixture Density and Hazard

$$f(t|\mathbf{x}) = -\frac{d}{dt}S(t|\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) f_k(t|\mathbf{x}) \tag{5.7}$$

$$h(t|\mathbf{x}) = \frac{f(t|\mathbf{x})}{S(t|\mathbf{x})} = \frac{\sum_{k=1}^{K} \pi_k(\mathbf{x}) f_k(t|\mathbf{x})}{\sum_{k=1}^{K} \pi_k(\mathbf{x}) S_k(t|\mathbf{x})} \tag{5.8}$$

5.6.3 Insight into Hazard Composition

The mixture hazard can be rewritten to provide insight into how the different components contribute:

Alternative Hazard Formulation

$$h(t|\mathbf{x}) = \frac{\sum_{k=1}^{K} \pi_k(\mathbf{x}) f_k(t|\mathbf{x})}{\sum_{k=1}^{K} \pi_k(\mathbf{x}) S_k(t|\mathbf{x})} \tag{5.9}$$

$$= \sum_{k=1}^{K} \frac{\pi_k(\mathbf{x}) S_k(t|\mathbf{x})}{\sum_{j=1}^{K} \pi_j(\mathbf{x}) S_j(t|\mathbf{x})} h_k(t|\mathbf{x}) \tag{5.10}$$

This reveals that the overall hazard is a weighted combination of component hazards, where the weights change over time. The weighting factor $\frac{\pi_k S_k}{\sum_j \pi_j S_j}$ represents the posterior probability that an individual who survived to time t belongs to component k.

5.7 Parametric Component Distributions

DSM can use various parametric distributions for its mixture components. The two most common choices are the Weibull and log-normal distributions.

5.7.1 Weibull Distribution

The Weibull distribution is frequently used due to its flexibility in modeling different hazard shapes:

Weibull Distribution Functions

$$S_k(t|\mathbf{x}) = \exp\left[-\left(\frac{t}{\lambda_k(\mathbf{x})}\right)^{\alpha_k(\mathbf{x})}\right] \tag{5.11}$$

$$f_k(t|\mathbf{x}) = \frac{\alpha_k(\mathbf{x})}{\lambda_k(\mathbf{x})}\left(\frac{t}{\lambda_k(\mathbf{x})}\right)^{\alpha_k(\mathbf{x})-1} \exp\left[-\left(\frac{t}{\lambda_k(\mathbf{x})}\right)^{\alpha_k(\mathbf{x})}\right] \tag{5.12}$$

$$h_k(t|\mathbf{x}) = \frac{\alpha_k(\mathbf{x})}{\lambda_k(\mathbf{x})}\left(\frac{t}{\lambda_k(\mathbf{x})}\right)^{\alpha_k(\mathbf{x})-1} \tag{5.13}$$

where:

- $\lambda_k(\mathbf{x})$ is the scale parameter, which influences the median survival time
- $\alpha_k(\mathbf{x})$ is the shape parameter, which determines the hazard behavior

The shape parameter α_k controls the hazard shape in a Weibull distribution:

- $\alpha_k = 1$: Constant hazard (exponential distribution)
- $\alpha_k > 1$: Increasing hazard (e.g., wear-out, aging)
- $\alpha_k < 1$: Decreasing hazard (e.g., early failures, infections)

5.7.2 Log-Normal Distribution

DSM can also use log-normal components, which provide different hazard characteristics:

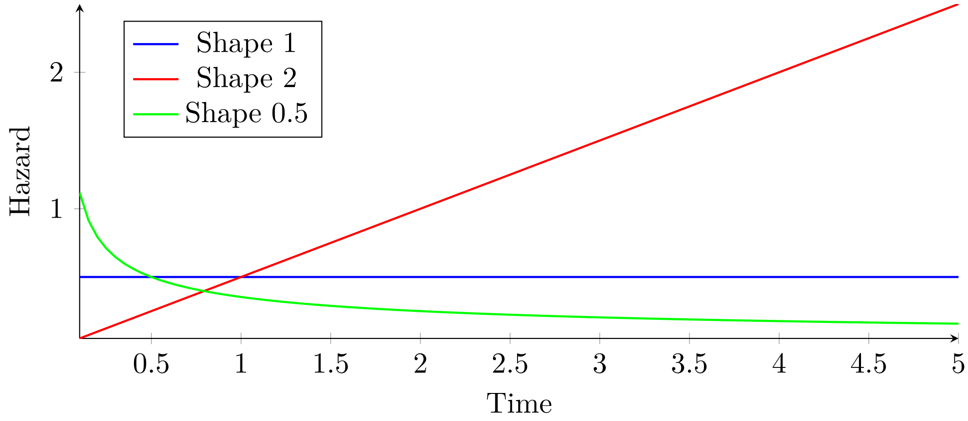


Figure 5.4: Weibull hazard functions with different shape parameters. When $\alpha = 1$ (blue), the hazard is constant. When $\alpha > 1$ (red), the hazard increases with time. When $\alpha < 1$ (green), the hazard decreases with time.

Log-Normal Distribution Functions

$$S_k(t|\mathbf{x}) = 1 - \Phi\left(\frac{\log t - \mu_k(\mathbf{x})}{\sigma_k(\mathbf{x})}\right) \tag{5.14}$$

$$f_k(t|\mathbf{x}) = \frac{1}{t\sigma_k(\mathbf{x})\sqrt{2\pi}} \exp\left(-\frac{(\log t - \mu_k(\mathbf{x}))^2}{2\sigma_k^2(\mathbf{x})}\right) \tag{5.15}$$

$$h_k(t|\mathbf{x}) = \frac{f_k(t|\mathbf{x})}{S_k(t|\mathbf{x})} \tag{5.16}$$

where:

- $\mu_k(\mathbf{x})$ is the location parameter
- $\sigma_k(\mathbf{x})$ is the scale parameter
- Φ is the standard normal cumulative distribution function

The log-normal distribution has distinct characteristics:

- Non-monotonic hazard that increases and then decreases
- Better for modeling certain diseases with delayed effects
- More parameters to optimize
- Often used for modeling cancer survival with delayed treatment effects

5.7.3 The Power of Mixtures

The mixture approach allows DSM to model complex hazard patterns that would be impossible with a single parametric distribution. With just 2-4 components, DSM can represent:

- Bathtub hazards (high early risk, low middle, high late)

- Multi-modal hazards (multiple risk peaks)
- U-shaped hazards
- Virtually any smooth hazard function

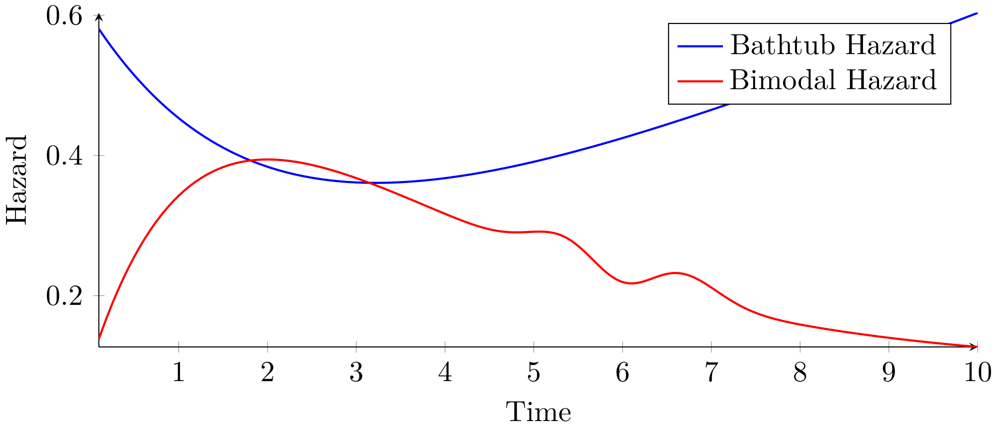


Figure 5.5: Complex hazard patterns that can be modeled using mixtures of simple parametric distributions. The bathtub pattern (blue) shows high initial risk, followed by a stable period, and then increasing risk. The bimodal pattern (red) shows two distinct risk peaks.

5.8 Neural Parameter Mapping

A key aspect of DSM is the use of neural networks to map from input features to distribution parameters.

5.8.1 From Features to Distribution Parameters

DSM employs a series of neural network transformations to derive distribution parameters from input features:

Neural Parameter Mapping

$$\phi(\mathbf{x}) = \text{neural\_network}(\mathbf{x}) \tag{5.17}$$

$$\alpha_k(\mathbf{x}) = \text{softplus}(w_{\alpha_k}^T \phi(\mathbf{x}) + b_{\alpha_k}) + \varepsilon \tag{5.18}$$

$$\lambda_k(\mathbf{x}) = \text{softplus}(w_{\lambda_k}^T \phi(\mathbf{x}) + b_{\lambda_k}) + \varepsilon \tag{5.19}$$

$$\tilde{\pi}_k(\mathbf{x}) = w_{\pi_k}^T \phi(\mathbf{x}) + b_{\pi_k} \tag{5.20}$$

$$\pi_k(\mathbf{x}) = \frac{\exp(\tilde{\pi}_k(\mathbf{x}))}{\sum_{j=1}^{K} \exp(\tilde{\pi}_j(\mathbf{x}))} \tag{5.21}$$

where:

- $\phi(\mathbf{x})$ is the shared representation from the neural network
- w_{α_k}, w_{λ_k}, w_{π_k} are parameter-specific weights

- b_{α_k}, b_{λ_k}, b_{π_k} are parameter-specific biases
- softplus activation ensures positive values for α_k and λ_k
- softmax ensures mixture weights π_k sum to 1
- ε is a small positive constant (e.g., 0.01) to prevent numerical issues

5.8.2 Network Architecture

The complete DSM architecture can be visualized as follows:

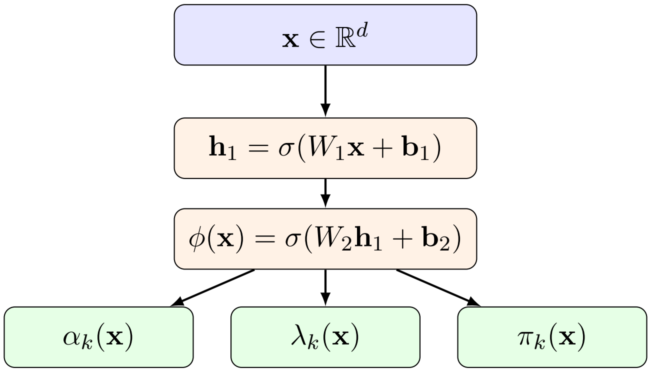

$\mathbf{x} \in \mathbb{R}^d$
$\mathbf{h}_1 = \sigma(W_1\mathbf{x} + \mathbf{b}_1)$
$\phi(\mathbf{x}) = \sigma(W_2\mathbf{h}_1 + \mathbf{b}_2)$
$\alpha_k(\mathbf{x})$
$\lambda_k(\mathbf{x})$
$\pi_k(\mathbf{x})$

Figure 5.6: Neural network architecture for DSM. Input features are processed through shared hidden layers to create a representation $\phi(\mathbf{x})$, which is then fed into parameter-specific output heads to generate distribution parameters and mixture weights.

5.9 Training Deep Survival Machines

Training DSM involves defining an appropriate loss function and addressing several numerical challenges.

5.9.1 Loss Function Formulation

For censored survival data:

- If event observed ($\delta_i = 1$): Time t_i is the exact event time
- If censored ($\delta_i = 0$): Event occurs after t_i (right censoring)

The likelihood contribution for each observation is:

$$L_i = [f(t_i|\mathbf{x}_i)]^{\delta_i} \cdot [S(t_i|\mathbf{x}_i)]^{1-\delta_i} \tag{5.22}$$

Taking the negative log of the likelihood and expanding:

$$-\log L_i = -\delta_i \log f(t_i|\mathbf{x}_i) - (1-\delta_i)\log S(t_i|\mathbf{x}_i) \tag{5.23}$$

$$= -\delta_i \log\left[\sum_{k=1}^{K} \pi_k(\mathbf{x}_i) f_k(t_i|\mathbf{x}_i)\right] \tag{5.24}$$

$$-(1-\delta_i)\log\left[\sum_{k=1}^{K} \pi_k(\mathbf{x}_i) S_k(t_i|\mathbf{x}_i)\right] \tag{5.25}$$

The full loss is then the sum over all observations:

$$\mathcal{L}_{DSM} = \sum_{i=1}^{N} -\log L_i \tag{5.26}$$

5.9.2 ELBO-Based Regularization

Mixture models can suffer from component collapse, where all weight concentrates in just one component. To encourage the use of multiple components and create more diverse representations, DSM employs an Evidence Lower Bound (ELBO) regularization approach.

From a latent variable perspective, we can view k as a latent mixture component:

$$\log p(t|\mathbf{x}) = \log \sum_{k=1}^{K} p(t,k|\mathbf{x}) = \log \sum_{k=1}^{K} p(t|k,\mathbf{x})p(k|\mathbf{x}) \tag{5.27}$$

$$\geq \sum_{k=1}^{K} q(k|\mathbf{x}) \log \frac{p(t|k,\mathbf{x})p(k|\mathbf{x})}{q(k|\mathbf{x})} \tag{5.28}$$

$$= \sum_{k=1}^{K} q(k|\mathbf{x}) \log p(t|k,\mathbf{x}) - \sum_{k=1}^{K} q(k|\mathbf{x}) \log \frac{q(k|\mathbf{x})}{p(k|\mathbf{x})} \tag{5.29}$$

$$= \mathbb{E}_{q(k|\mathbf{x})}[\log p(t|k,\mathbf{x})] - KL(q(k|\mathbf{x})||p(k|\mathbf{x})) \tag{5.30}$$

If we use a uniform prior $p(k|\mathbf{x}) = 1/K$, the KL term becomes:

$$KL(q(k|\mathbf{x})||p(k|\mathbf{x})) = \sum_{k=1}^{K} q(k|\mathbf{x}) \log(K \cdot q(k|\mathbf{x})) \tag{5.31}$$

This motivates the ELBO regularizer that encourages diverse component usage:

$$\mathcal{L}_{ELBO} = \mathcal{L}_{DSM} + \beta \cdot KL(\pi(\mathbf{x})||\text{Uniform}) \tag{5.32}$$

Where:

- β is a hyperparameter controlling regularization strength
- $\pi(\mathbf{x})$ is used as the $q(k|\mathbf{x})$ distribution
- This pushes mixture weights toward a uniform distribution

5.10 Implementation Challenges and Solutions

Implementing DSM involves addressing several numerical challenges to ensure stability and good performance.

5.10.1 Numerical Stability Challenges

The mixture likelihood calculation can encounter several numerical issues:

- Overflow/underflow in exponential calculations
- Division by zero risk in hazard calculations
- Gradient explosion during backpropagation
- Parameter values spanning many orders of magnitude

5.10.2 Log-Sum-Exp Trick

For stable computation of $\log \sum_i e^{x_i}$, the log-sum-exp trick is essential:

$$\begin{aligned} \log \sum_i e^{x_i} &= \log \left[e^a \sum_i e^{x_i - a} \right] && (5.33) \\ &= a + \log \sum_i e^{x_i - a} && (5.34) \end{aligned}$$

where $a = \max_i x_i$

This avoids underflow/overflow by bringing values to a numerically stable range.

5.10.3 Gradient Detachment Strategy

The gradient of the Weibull hazard with respect to the shape parameter α can explode for small time values when $\alpha < 1$:

$$\begin{aligned} \frac{\partial h(t|\alpha, \lambda)}{\partial \alpha} &= \frac{\partial}{\partial \alpha} \left[\frac{\alpha}{\lambda} \left(\frac{t}{\lambda} \right)^{\alpha - 1} \right] && (5.35) \\ &= \frac{1}{\lambda} \left(\frac{t}{\lambda} \right)^{\alpha - 1} + \frac{\alpha}{\lambda} \left(\frac{t}{\lambda} \right)^{\alpha - 1} \ln \left(\frac{t}{\lambda} \right) && (5.36) \end{aligned}$$

To address gradient explosion, DSM implements a gradient detachment strategy:

1. Identify potentially unstable operations
2. Create a mask for extreme values
3. Detach gradient computation for these values
4. Use a weighted combination approach:

$$\begin{aligned} result &= (1 - mask) \cdot normal\_result + mask \cdot detached\_result && (5.37) \\ mask &= \mathbb{I}(|x| > threshold).float() && (5.38) \end{aligned}$$

This allows learning to continue for stable values while preventing NaN gradients.

5.11 Advantages of DSM over Traditional Methods

DSM offers several key advantages over traditional survival analysis approaches, making it particularly suitable for complex, high-dimensional datasets.

5.11.1 Compared to Cox Proportional Hazards

- **Flexibility:** Can model non-proportional hazards
- **Risk estimation:** Provides absolute risk estimates, not just relative
- **Feature learning:** Automatically learns non-linear feature interactions
- **Time prediction:** Directly models time-to-event distribution
- **Complex patterns:** Can capture non-monotonic hazard patterns

5.11.2 Compared to Neural Cox Models

Neural Cox models apply neural networks to learn features but still maintain the proportional hazards assumption. DSM offers advantages:

- **No baseline estimation:** No baseline hazard estimation needed
- **Extrapolation:** Better extrapolation beyond training data
- **Uncertainty quantification:** Natural uncertainty quantification
- **Hazard flexibility:** Can model virtually any hazard pattern
- **Direct time modeling:** Models time directly rather than through hazards

5.11.3 Compared to Random Survival Forests

- **End-to-end learning:** Differentiable learning instead of greedy splits
- **High-dimensional data:** Better handling of high-dimensional feature spaces
- **Representation learning:** More flexible feature learning
- **Computational efficiency:** More efficient after training
- **Scalability:** Better scalability to large datasets

5.11.4 Limitations and Considerations

While DSM offers many advantages, it also has limitations to consider:

- **Distributional assumptions:** Requires choosing parametric component distributions
- **Computational intensity:** More computationally intensive to train than simpler models
- **Interpretability:** Less interpretable than simpler models, though mixture components can aid understanding
- **Implementation complexity:** Needs careful numerical implementation
- **Hyperparameters:** More hyperparameters to tune

5.12 Practical Applications of DSM

DSM has been successfully applied to various domains, demonstrating its practical utility.

5.12.1 Cancer Survival Prediction

One notable application is in breast cancer survival prediction using the METABRIC dataset:

- Mixture of 4 Weibull distributions
- Neural network with 2 hidden layers
- Clinical and genomic features as input
- Shape parameters constrained to meaningful ranges for cancer progression
- Transformer-based feature embedding for handling complex feature interactions

Results showed significant improvements over traditional methods:

- Higher C-index than Cox PH (0.67 vs 0.63)
- Better integrated Brier score
- Provided uncertainty intervals for risk stratification
- Identified distinct risk subgroups aligned with clinical knowledge

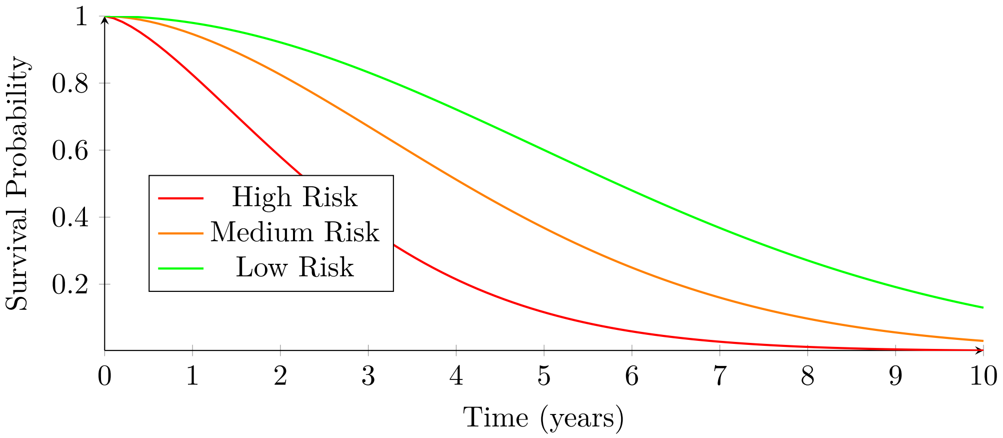


Figure 5.7: Risk stratification using DSM for breast cancer patients. The model identifies distinct risk groups with different survival trajectories, enabling more personalized treatment planning.

5.13 Summary: DSM in the Bigger Picture

Deep Survival Machines represents a significant advance in survival analysis, bridging the gap between traditional statistical methods and modern deep learning approaches.

5.13.1 Key Contributions

- Combines neural networks with parametric survival distributions
- Models complex hazard shapes through the mixture approach
- Provides natural uncertainty quantification
- Bridges the gap between deep learning and survival analysis
- Enables personalized risk prediction with confidence intervals

5.13.2 Future Directions

The DSM framework opens several exciting avenues for future research:

- Integration with causal inference frameworks
- Extension to interval censoring and competing risks (MENSA)
- Combination with advanced embedding techniques for different data modalities
- Application to federated learning settings for privacy-preserving analysis
- Incorporation of domain-specific prior knowledge to enhance predictions

Looking Ahead

In the next chapter, we will explore Multi-Event Neural Survival Analysis (MENSA), which extends the DSM framework to handle competing risks scenarios where subjects can experience different types of events. MENSA addresses the challenge of modeling dependencies between different event types while maintaining the flexibility and power of the DSM approach.

Chapter 6

Multi-Event Neural Survival Analysis

6.1 Introduction to Multi-Event Neural Survival Analysis

6.1.1 The Big Picture: Moving Beyond Single-Event Survival

In real-world scenarios, patients and systems often face multiple possible outcomes or failure modes (Fine and Gray, 1999; Prentice et al., 1978). While traditional survival analysis typically focuses on a single event (like death or machine failure), many applications require modeling multiple competing or sequential events (Austin, D. S. Lee, and Fine, 2016; Koller et al., 2012). Multi-Event Neural Survival Analysis (MENSA) (Y. Zhong et al., 2021) addresses this challenge by extending the Deep Survival Machines (DSM) framework to handle competing risks scenarios.

What is MENSA?

MENSA (Multi-Event Neural Survival Analysis) is a deep learning framework designed to model time-to-event data with multiple possible events. It combines the flexible distribution-based approach of DSM with the ability to model dependencies between different event types.

MENSA brings several key advantages to survival analysis:

- It models both event-specific patterns and inter-event dependencies
- It enables comprehensive risk prediction across multiple outcomes
- It bridges the gap between single-event survival and multi-state modeling
- It allows for information sharing across event types while maintaining event-specific characteristics

6.1.2 Challenges in Multi-Event Survival Analysis

Modeling multiple events introduces several challenges beyond those found in traditional single-event survival analysis:

- **Multiple competing events:** Each event type may have distinct hazard profiles and risk factors

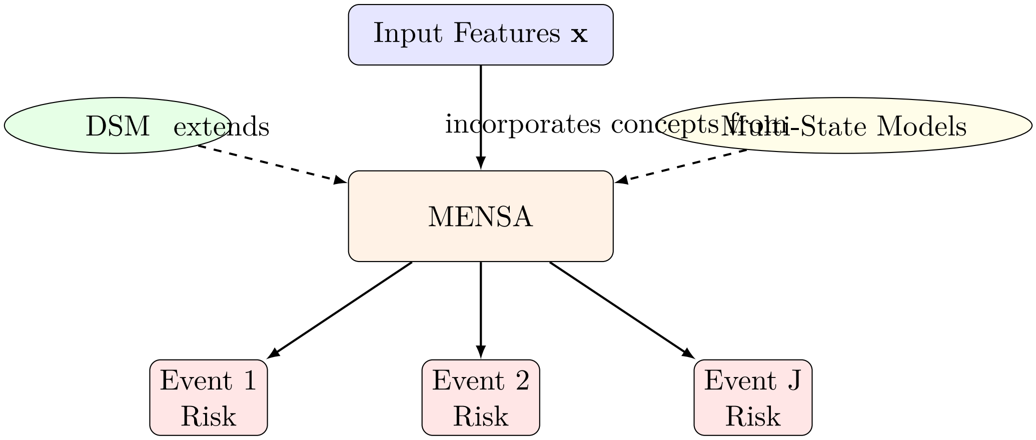


Figure 6.1: Conceptual overview of the MENSA framework, showing its relationship to DSM and multi-state models, as well as its ability to model multiple event risks simultaneously.

- **Complex dependencies:** Events can be interdependent, with the risk of one affecting the others
- **Data sparsity:** Individual events may have limited samples, requiring information sharing
- **Competing risk structure:** One event precludes the observation of others, creating informative censoring
- **Expert knowledge integration:** Each event may have domain-specific knowledge to incorporate

Figure 6.2 illustrates the competing risks structure, where a subject can transition from the initial state to one of several possible event states, but experiencing one event prevents observation of the others.

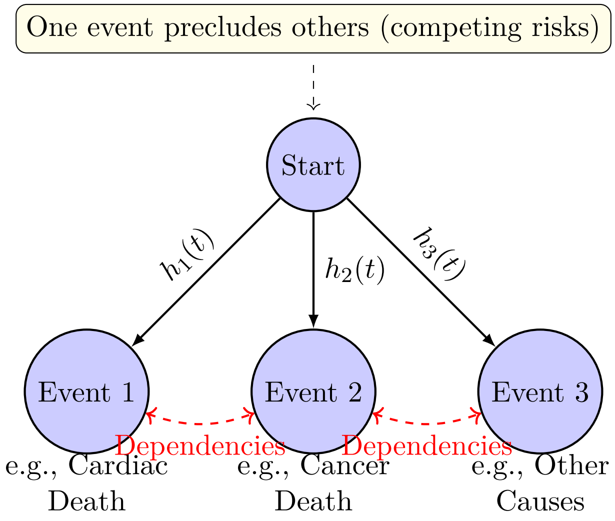


Figure 6.2: Competing risks structure, showing transitions from a starting state to multiple possible events, with dependencies between events.

6.2 The Competing Risks Framework

6.2.1 Data Representation in Competing Risks

In the competing risks setting, each subject can experience one of J different event types. We observe a triplet of information for each subject i:

Competing Risks Data Structure

For each subject i, we observe:

- T_i: The observed time (either event time or censoring time)
- $J_i \in \{1, 2, \ldots, J\}$: The event type (if an event occurred)
- $\delta_i \in \{0, 1\}$: Censoring indicator (1 if event observed, 0 if censored)

If $\delta_i = 0$ (censored), then J_i is undefined as no event has been observed.

Table 6.1 shows an example dataset with competing risks. Note how censored subjects (rows 3 and 5) have no defined event type.

Table 6.1: Example of competing risks data structure

Subject	Time T_i	Event Type J_i	Indicator δ_i
1	3.5	1	1
2	5.0	2	1
3	7.0	–	0
4	2.3	3	1
5	4.2	–	0

In this example:

- Subject 1 experienced event type 1 at time 3.5
- Subject 2 experienced event type 2 at time 5.0
- Subject 3 was censored at time 7.0 without experiencing any event
- Subject 4 experienced event type 3 at time 2.3
- Subject 5 was censored at time 4.2 without experiencing any event

6.2.2 Censoring in Competing Risks

Censoring becomes more complex in the competing risks setting. We need to distinguish between:

- **Administrative censoring**: The study ended before any event was observed
- **Competing event censoring**: A different event occurred, which precludes observation of the event of interest

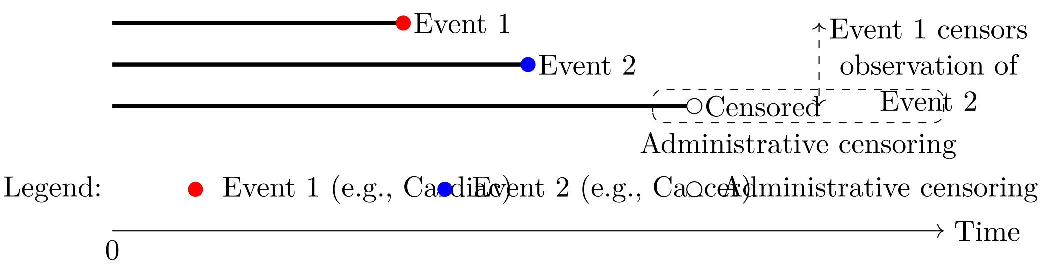


Figure 6.3: Illustration of censoring in competing risks. The occurrence of Event 1 prevents observing whether Event 2 would have occurred, creating a form of informative censoring specific to competing risks.

This distinction is crucial because competing event censoring is informative for the event of interest, unlike standard administrative censoring which is typically assumed to be non-informative. In Figure 6.3, for Patient 1, we'll never know if or when Event 2 would have occurred because Event 1 happened first.

6.3 Key Functions in Competing Risks

To properly analyze competing risks data, we need to define several key functions that extend traditional survival analysis concepts.

6.3.1 Cause-Specific Hazard Function

Cause-Specific Hazard

The cause-specific hazard for event j represents the instantaneous risk of experiencing event j at time t, given survival (no events of any type) up to time t:

$$h_j(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t, J = j | T \geq t)}{\Delta t} \tag{6.1}$$

The cause-specific hazard can be interpreted as the rate at which event j occurs at time t among subjects who have not yet experienced any event. Each event type has its own hazard function, and these functions can have different shapes.

Figure 6.4 shows three different cause-specific hazard functions:

- Event 1: Constant hazard (exponential distribution)
- Event 2: Increasing hazard (e.g., Weibull with shape parameter > 1)
- Event 3: Decreasing hazard (e.g., Weibull with shape parameter < 1)

The overall hazard function $h(t)$ is the sum of all cause-specific hazards:

Equation

$$h(t) = \sum_{j=1}^{J} h_j(t) \tag{6.2}$$

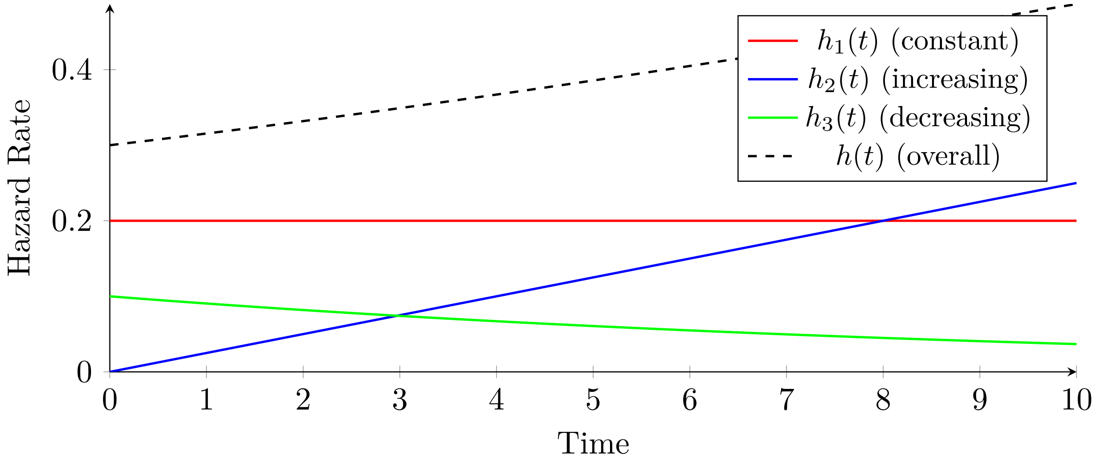


Figure 6.4: Cause-specific hazard functions for three different event types. Note how each event can have a distinct hazard pattern (constant, increasing, or decreasing over time).

6.3.2 Overall Survival Function

The overall survival function represents the probability of not experiencing any event up to time t. It depends on all cause-specific hazards:

Overall Survival Function

$$S(t) = P(T > t) = \exp\left(-\sum_{j=1}^{J}\int_0^t h_j(u)du\right) = \exp\left(-\sum_{j=1}^{J} H_j(t)\right) \tag{6.3}$$

where $H_j(t) = \int_0^t h_j(u)du$ is the cumulative hazard for event j.

This equation shows how the overall survival is affected by all event types. If any cause-specific hazard increases, the overall survival decreases more rapidly. The survival function can be factorized as the product of survival functions specific to each event type, if we were hypothetically able to remove other competing risks:

$$S(t) = \exp(-H_1(t)) \cdot \exp(-H_2(t)) \cdot \ldots \cdot \exp(-H_J(t)) \tag{6.4}$$

6.3.3 Cumulative Incidence Function

The Cumulative Incidence Function (CIF) is central to competing risks analysis. Unlike in standard survival, we need a separate CIF for each event type.

Cumulative Incidence Function

The CIF for event j represents the probability of experiencing event j by time t:

$$F_j(t) = P(T \leq t, J = j) = \int_0^t h_j(u)S(u)du \tag{6.5}$$

The CIF accounts for the competing nature of risks by incorporating the overall survival function. At any time point, the sum of all CIFs plus the overall survival equals 1:

$$S(t) + \sum_{j=1}^{J} F_j(t) = 1 \tag{6.6}$$

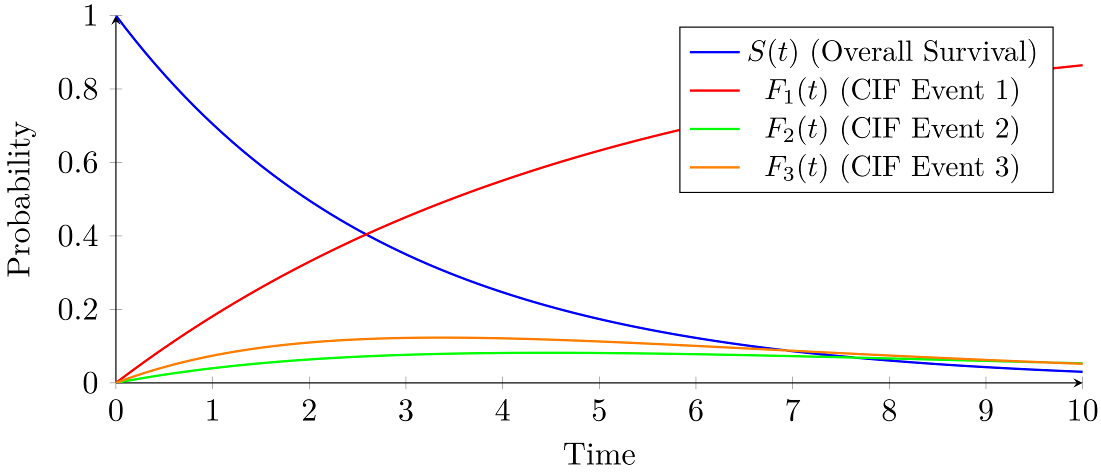


Figure 6.5: Overall survival function and cause-specific cumulative incidence functions (CIFs) for three competing events. Note how all CIFs plus the survival function sum to 1 at any time point.

6.3.4 Sub-Density Function

The sub-density function for event j is the derivative of the corresponding CIF:

Sub-Density Function

$$f_j(t) = \frac{d}{dt}F_j(t) = h_j(t)S(t) \tag{6.7}$$

This represents the probability density of experiencing event j at exactly time t. It equals the cause-specific hazard multiplied by the overall survival probability, capturing the joint effect of the instantaneous risk and the probability of having survived all events up to that time.

6.4 Traditional Approaches to Competing Risks

Before introducing MENSA, let's briefly review traditional methods for analyzing competing risks data.

6.4.1 Cause-Specific Cox Models

The most common approach is to fit separate Cox proportional hazards models for each event type:

Cause-Specific Cox Model

$$h_j(t|\mathbf{x}) = h_{0j}(t)\exp(\boldsymbol{\beta}_j^T \mathbf{x}) \tag{6.8}$$

where $h_{0j}(t)$ is the baseline hazard for event j, and $\boldsymbol{\beta}_j$ are the coefficients for event j.

This approach:

- Treats each event type as a separate modeling problem
- Assumes proportional hazards for each cause-specific hazard
- Does not model dependencies between event types
- Requires separate models for each event of interest

6.4.2 Fine-Gray Model

The Fine-Gray model directly models the subdistribution hazard:

Fine-Gray Model

$$h_j^{sub}(t|\mathbf{x}) = h_{0j}^{sub}(t)\exp(\boldsymbol{\gamma}_j^T\mathbf{x}) \tag{6.9}$$

where $h_j^{sub}(t) = -\frac{d\log(1-F_j(t))}{dt}$ is the subdistribution hazard.

The Fine-Gray model:

- Focuses on the CIF rather than the cause-specific hazard
- Allows direct modeling of cumulative incidence
- Also assumes proportional hazards (but for the subdistribution hazard)
- Requires separate models for each event type

Note

Both cause-specific Cox models and Fine-Gray models are semi-parametric and don't model the full joint distribution of events. They also don't naturally incorporate dependencies between event types or leverage neural networks for complex covariate relationships.

In the next section, we'll introduce the MENSA framework, which addresses these limitations through a parametric mixture approach combined with deep learning.

6.5 The MENSA Framework

6.5.1 Core Conceptual Innovation

MENSA builds on the DSM framework by extending it to handle multiple competing events. The key conceptual innovations include:

- Using a mixture of parametric distributions for each event type
- Sharing representation learning across event types
- Modeling dependencies between events through shared latent space
- Maintaining flexible hazard shapes for each event
- Learning event-specific and shared risk factors

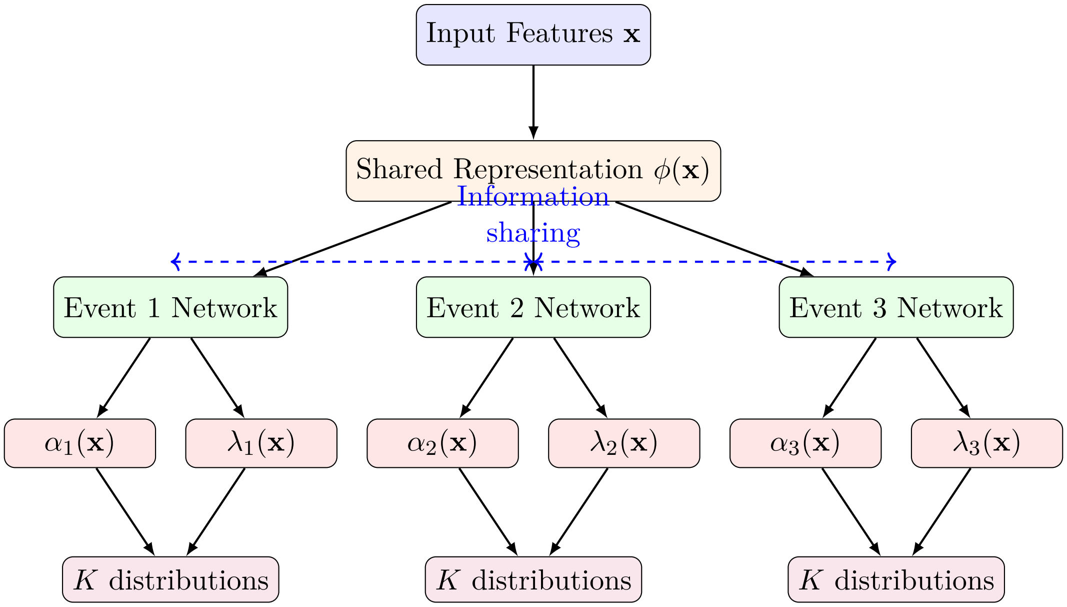


Figure 6.6: MENSA architecture showing the flow from input features through shared representation to event-specific networks and ultimately to the parameterization of the mixture distributions for each event type.

6.5.2 Mathematical Formulation

Let's formalize the MENSA framework mathematically. For a subject with covariates $\mathbf{x}$, we model the cause-specific hazard for each event type $j \in \{1, 2, \ldots, J\}$ as a mixture of parametric distributions:

MENSA Cause-Specific Hazard

$$h_j(t|\mathbf{x}) = \sum_{k=1}^{K} w_{jk}(\mathbf{x}) \cdot h_{jk}(t|\alpha_{jk}(\mathbf{x}), \lambda_{jk}(\mathbf{x})) \tag{6.10}$$

where:

- $h_{jk}(t|\alpha_{jk}, \lambda_{jk})$ is the hazard function for the k-th component of event type j
- $w_{jk}(\mathbf{x})$ are the mixture weights for event j, with $\sum_{k=1}^{K} w_{jk}(\mathbf{x}) = 1$
- $\alpha_{jk}(\mathbf{x})$ and $\lambda_{jk}(\mathbf{x})$ are the shape and scale parameters of the distributions

Typically, MENSA uses Weibull distributions for the mixture components:

$$h_{jk}(t|\alpha_{jk}, \lambda_{jk}) = \frac{\alpha_{jk}}{\lambda_{jk}} \left(\frac{t}{\lambda_{jk}}\right)^{\alpha_{jk}-1} \tag{6.11}$$

The overall survival function is:

$$S(t|\mathbf{x}) = \exp\left(-\sum_{j=1}^{J} \int_0^t h_j(u|\mathbf{x})\, du\right) \tag{6.12}$$

And the cumulative incidence function for event j is:

$$F_j(t|\mathbf{x}) = \int_0^t h_j(u|\mathbf{x}) \cdot S(u|\mathbf{x})\, du \tag{6.13}$$

6.5.3 Neural Network Architecture

MENSA implements the above formulation using neural networks to learn the parameters:

1. **Shared representation network:** $\phi(\mathbf{x})$ maps input features to a shared latent space
2. **Event-specific networks:** Map from the shared representation to event-specific parameters
3. **Distribution parameter networks:** Output the shape and scale parameters for each event and mixture component

The sharing of the representation $\phi(\mathbf{x})$ enables information transfer across event types, while event-specific networks capture the unique characteristics of each event type.

Advantages of the MENSA Architecture

The MENSA architecture offers several advantages:

- **Flexibility:** The mixture of distributions allows for complex, multi-modal hazard shapes
- **Information sharing:** Common risk factors can be learned from the pooled data across events
- **Event-specific modeling:** Each event has its own dedicated parameters
- **End-to-end learning:** All parameters are learned jointly via maximum likelihood
- **Dependency modeling:** The shared representation captures correlations between event risks

6.6 Training and Optimization

6.6.1 Likelihood Function

MENSA is trained by maximizing the likelihood of the observed data. The likelihood function for competing risks data needs to account for both event occurrences and censoring:

MENSA Likelihood Function

$$\mathcal{L}(\theta) = \prod_{i:\delta_i=1} f_{J_i}(T_i|\mathbf{x}_i, \theta) \prod_{i:\delta_i=0} S(T_i|\mathbf{x}_i, \theta) \tag{6.14}$$

where:

- θ represents all model parameters
- $f_j(t|\mathbf{x}, \theta) = h_j(t|\mathbf{x}, \theta) S(t|\mathbf{x}, \theta)$ is the sub-density function for event j
- $S(t|\mathbf{x}, \theta)$ is the overall survival function

Taking the logarithm, we get the log-likelihood:

$$\ell(\theta) = \sum_{i:\delta_i=1} \log h_{J_i}(T_i|\mathbf{x}_i, \theta) + \sum_{i=1}^{n} \log S(T_i|\mathbf{x}_i, \theta) \tag{6.15}$$

The first term involves the cause-specific hazard for observed events, while the second term includes the overall survival function for all subjects (both events and censored).

6.6.2 Optimization Techniques

Training MENSA involves several optimization considerations:

- **Parameter initialization:** Careful initialization of distribution parameters is crucial for stable training
- **Gradient-based optimization:** Typically using Adam or similar optimizers
- **Regularization:** L1/L2 regularization to prevent overfitting
- **Handling data imbalance:** When some event types are rare, balancing techniques may be needed
- **Early stopping:** Based on validation performance

6.6.3 Avoiding Numerical Issues

MENSA's training can face numerical stability challenges:

- **Vanishing/exploding gradients:** Addressed through gradient clipping
- **Distribution parameter constraints:** All shape and scale parameters must be positive (typically enforced through softplus activation)
- **Loss scaling:** The log-likelihood may need scaling for stable gradients
- **Mixture component collapse:** Regularization techniques to prevent mixture components from collapsing

Code Example: MENSA Parameter Constraints

```
# Ensuring positivity of Weibull parameters
alpha = nn.Softplus()(alpha_logits) + 0.01  # Shape parameter
lambda_ = nn.Softplus()(lambda_logits) + 0.01  # Scale parameter

# Ensuring mixture weights sum to 1
weights = nn.Softmax(dim=-1)(weight_logits)  # Mixture weights
```

6.7 Inference and Risk Prediction

6.7.1 Risk Predictions with MENSA

MENSA enables several types of risk predictions:

1. **Overall survival probability:** $S(t|\mathbf{x})$
2. **Cumulative incidence of event j:** $F_j(t|\mathbf{x})$
3. **Cause-specific hazard of event j:** $h_j(t|\mathbf{x})$
4. **Time to event density for event j:** $f_j(t|\mathbf{x})$
5. **Conditional event probability:** $P(J=j|T=t,\mathbf{x}) = \frac{h_j(t|\mathbf{x})}{\sum_{j'=1}^{J} h_{j'}(t|\mathbf{x})}$

These predictions can be computed for individual subjects, allowing for personalized risk assessment.

6.7.2 Uncertainty Quantification

MENSA provides several approaches to quantify uncertainty in predictions:

- **Direct parameterization:** The mixture components naturally model uncertainty in the event timing
- **Epistemic uncertainty:** Can be estimated using ensembling or Monte Carlo dropout
- **Aleatoric uncertainty:** Captured through the inherent variability of the parametric distributions
- **Confidence intervals:** Can be constructed via bootstrapping or analytical approximations

6.7.3 Interpreting MENSA Models

Interpreting the complex MENSA model can be approached in several ways:

- **Feature importance:** Using permutation-based or gradient-based importance measures
- **Partial dependence plots:** Showing the relationship between specific features and predicted risks
- **Shapley values:** Quantifying the contribution of each feature to predictions
- **Distribution visualization:** Plotting the learned mixture distributions for each event type

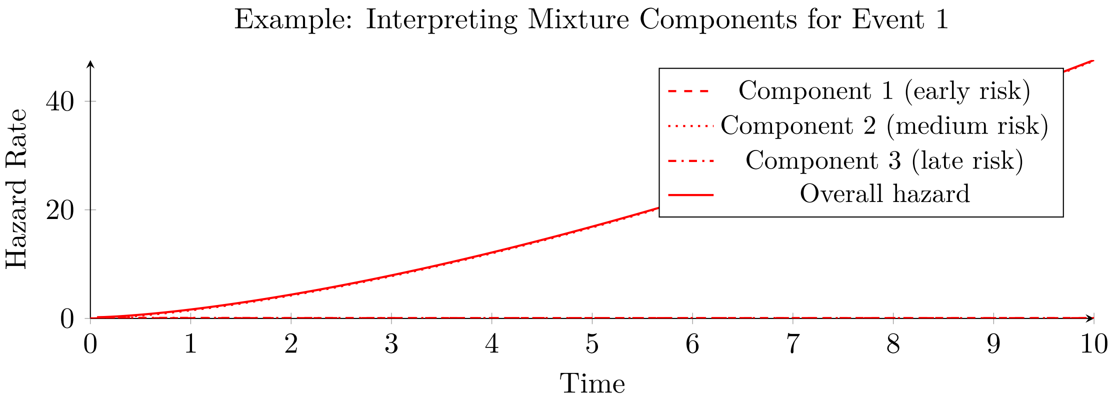


Figure 6.7: Interpretation of mixture components for an event's hazard function. Each component might represent a different risk sub-group or risk mechanism.

6.8 Applications of MENSA

6.8.1 Medical Applications

MENSA is particularly valuable in medical settings with multiple possible outcomes:

- **Cancer progression:** Modeling competing risks of cancer-specific death versus other causes
- **Cardiovascular diseases:** Predicting different types of cardiovascular events (heart attack, stroke, etc.)
- **Organ transplantation:** Modeling risks of rejection, infection, and other complications
- **Multiple disease progression:** For patients with comorbidities facing multiple potential health events

6.8.2 Industrial Applications

In industrial settings, MENSA can model competing failure modes:

- **Manufacturing:** Predicting different types of component failures
- **Energy systems:** Modeling various failure modes in power generation equipment
- **Transportation:** Predictive maintenance accounting for multiple failure types
- **Infrastructure:** Risk assessment for different types of structural failures

6.8.3 Business Applications

MENSA can also be applied to business scenarios with competing events:

- **Customer churn:** Modeling different reasons for customer attrition
- **Credit risk:** Predicting different types of default or delinquency
- **Marketing:** Modeling competing conversion types
- **Employee turnover:** Predicting different reasons for employee departures

6.9 Future Directions

MENSA opens several promising directions for future research:

- **Recurrent events:** Extending to scenarios where events can occur multiple times
- **Semi-competing risks:** Handling scenarios where some events preclude others but not vice versa
- **Time-varying covariates:** Incorporating dynamic features that change over time
- **Multi-state extensions:** Moving beyond competing risks to full multi-state models
- **Causal inference:** Integrating causal frameworks to estimate intervention effects
- **Treatment recommendation:** Using MENSA to guide personalized treatment decisions

6.10 Summary

Multi-Event Neural Survival Analysis (MENSA) extends the DSM framework to handle competing risks scenarios, providing a powerful approach for modeling complex time-to-event data with multiple possible outcomes. Key advantages include:

- Flexible modeling of cause-specific hazards through mixture distributions
- Sharing information across event types while maintaining event-specific modeling
- Capturing dependencies between different event types
- End-to-end learning of all parameters via maximum likelihood
- Comprehensive risk predictions and uncertainty quantification

By combining the strengths of parametric survival modeling with deep learning, MENSA offers a promising framework for addressing complex survival analysis problems in healthcare, industry, and business applications.

Chapter 7

Loss Functions

7.1 Loss Functions for Survival Analysis

Chapter Overview

This chapter covers:

- Essential loss functions designed specifically for survival analysis
- Mathematical formulations of specialized losses for time-to-event data with censoring
- Different objective functions for accurate time prediction, ranking, and calibration
- Adaptations of standard machine learning losses for survival contexts
- Advanced techniques for balancing multiple loss components

Loss functions represent the fundamental component that drives the learning process in machine learning models. In survival analysis, specialized loss functions are required to handle the unique characteristics of time-to-event data, particularly the presence of censoring (C. Lee et al., 2018; Kvamme, Borgan, and Scheel, 2019). This chapter provides a comprehensive examination of loss functions developed specifically for survival analysis, organized into four main categories: survival losses, regression losses, classification losses, and auxiliary losses.

Each category of loss function serves different objectives in survival modeling, from directly predicting the distribution of survival times to ranking patients according to their relative risk. Understanding the strengths, limitations, and appropriate applications of each loss function is crucial for developing effective survival models that meet the specific requirements of clinical applications.

7.1.1 Survival Losses

Survival losses form the core category of loss functions designed specifically for time-to-event data. They directly model the underlying survival distributions and naturally handle censored observations. This section explores the most important survival losses, from the classic negative log-likelihood with piecewise constant hazards to more sophisticated approaches in modern deep learning models.

Negative Log-Likelihood with Piecewise Constant Hazards

The Negative Log-Likelihood with Piecewise Constant Hazards (NLL-PCH) represents one of the fundamental loss functions in parametric survival analysis (Ibrahim, Chen, and Sinha, 2001). This approach models the hazard function as constant within predefined time intervals, creating a flexible yet tractable representation of survival distributions (Kleinbaum and Klein, 2012).

Piecewise Constant Hazard Function

A piecewise constant hazard function divides the time axis into distinct intervals and assigns a constant hazard rate within each interval. Formally, for a partition of the time axis $0 = t_0 < t_1 < \ldots < t_K$, the hazard function is defined as:

$$h(t) = h_k \quad \text{for} \quad t \in [t_{k-1}, t_k) \tag{7.1}$$

where h_k is the constant hazard rate in the k-th interval.

The piecewise constant hazard approach offers several advantages for survival modeling. It provides a flexible non-parametric representation of the hazard function while maintaining computational tractability. The hazard rates can be directly estimated from data, allowing the model to capture complex temporal patterns in the risk of events.

NLL-PCH Loss Function

For a single observation with survival time t_i and event indicator δ_i (where $\delta_i = 1$ indicates an observed event and $\delta_i = 0$ indicates censoring), the negative log-likelihood loss is given by:

$$\begin{aligned} -\log L_i &= -\delta_i \log h(t_i) - \log S(t_i) & (7.2) \\ &= -\delta_i \log h(t_i) + \sum_{j=0}^{k-1} h_j \Delta_j + h_k(t_i - t_k) & (7.3) \end{aligned}$$

where:

- $h(t_i)$ is the hazard rate at time t_i
- $S(t_i)$ is the survival function at time t_i
- h_j is the constant hazard in interval j
- Δ_j is the length of interval j
- k is the index of the interval containing t_i

The NLL-PCH loss function consists of two main components. For observed events ($\delta_i = 1$), the first term $-\log h(t_i)$ encourages the model to assign high hazard rates at the event time. The second term $-\log S(t_i)$ applies to all observations and encourages the model to assign high survival probabilities up to the observed time. This balances the model's tendency to estimate high hazard rates everywhere.

In a deep learning context, the model typically outputs the hazard rates for each interval, and the loss function guides the learning process to estimate hazards that maximize the likelihood of the observed data. This approach naturally handles censored observations by only considering the survival function (second term) for these cases.

Deep Survival Machines

Deep Survival Machines (DSM) represents a significant advancement in parametric survival modeling within the deep learning framework. Introduced by Nagpal, Li, and Dubrawski (2021), DSM employs a mixture of parametric distributions to model survival times, offering both flexibility and uncertainty quantification (McLachlan and Basford, 1988; Bishop, 2006).

Deep Survival Machines

Deep Survival Machines is a parametric survival analysis approach that models the survival function as a mixture of K parametric distributions:

$$S(t|\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) S_k(t|\mathbf{x}) \tag{7.4}$$

where $\pi_k(\mathbf{x})$ are the mixture weights that depend on covariates $\mathbf{x}$, and $S_k(t|\mathbf{x})$ are the survival functions of the component distributions (typically Weibull or Log-Normal).

The DSM architecture consists of several key components:

- A shared representation network that processes input features
- Distribution parameter networks that map the representation to parameters of each mixture component
- A mixture mechanism that combines the component distributions

This architecture allows DSM to capture complex patterns in the data while maintaining the interpretability of parametric models. The mixture approach enables the model to represent multimodal survival distributions, which is particularly valuable when different subgroups within the population exhibit distinct survival patterns.

DSM Loss Function

The negative log-likelihood loss for Deep Survival Machines is given by:

$$\mathcal{L}_{DSM} = -\sum_{i=1}^{N} [\delta_i \log f(t_i|\mathbf{x}_i) + (1 - \delta_i) \log S(t_i|\mathbf{x}_i)] \tag{7.5}$$

$$f(t|\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) f_k(t|\mathbf{x}) \tag{7.6}$$

$$S(t|\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) S_k(t|\mathbf{x}) \tag{7.7}$$

where:

- $f(t|\mathbf{x})$ is the probability density function
- $S(t|\mathbf{x})$ is the survival function

- $\pi_k(\mathbf{x})$ are mixture weights that depend on covariates
- f_k and S_k are component distributions
- δ_i is the event indicator

The DSM loss function follows the standard survival negative log-likelihood formulation but applies it to the mixture model context. For observed events ($\delta_i = 1$), the model maximizes the density at the event time, while for censored observations ($\delta_i = 0$), it maximizes the survival probability beyond the censoring time.

A significant advantage of DSM is its ability to provide uncertainty estimates through the mixture components. By examining the distribution of predictions across different mixture components, the model can quantify both aleatoric uncertainty (inherent randomness in the data) and epistemic uncertainty (model uncertainty due to limited data).

Multi-Event Neural Survival Analysis (MENSA)

Multi-Event Neural Survival Analysis (MENSA) extends the DSM approach to the competing risks setting, where individuals may experience multiple types of events (Y. Zhong et al., 2021; Austin, D. S. Lee, and Fine, 2016). MENSA explicitly models the dependencies between different event types, providing a more comprehensive framework for complex survival scenarios (Fine and Gray, 1999; Prentice et al., 1978).

Multi-Event Neural Survival Analysis

MENSA models multiple event types with dependencies by assigning a separate mixture distribution to each event type and incorporating a dependency structure between events:

$$S_j(t|\mathbf{x}) = \sum_{k=1}^{K} \pi_{jk}(\mathbf{x}) S_{jk}(t|\mathbf{x}, \mathbf{D}) \tag{7.8}$$

where S_j is the survival function for event type j, and $\mathbf{D}$ is a dependency matrix where D_{ij} represents the influence of event i on event j.

The MENSA architecture builds upon DSM by introducing:

- Event-specific mixture components for each event type
- A dependency structure that captures relationships between events
- Shared representations that enable information sharing across event types

This approach allows MENSA to model complex scenarios where different events are not independent, which is common in many clinical applications. For example, in cancer progression, the risk of metastasis may depend on the occurrence and timing of other biological events.

MENSA Loss Function

The MENSA loss function combines a negative log-likelihood term with a regularization term:

$$\mathcal{L}_{MENSA} = \mathcal{L}_{NLL} + \mathcal{L}_{reg} \tag{7.9}$$

$$\mathcal{L}_{NLL} = -\sum_{i=1}^{N}\sum_{j=1}^{J}\left[\delta_{ij}\log f_j(t_i|\mathbf{x}_i) + (1-\delta_{ij})\log S_j(t_i|\mathbf{x}_i)\right] \tag{7.10}$$

$$\mathcal{L}_{reg} = \lambda \sum_{i\neq j} |D_{ij}| \tag{7.11}$$

where:

- δ_{ij} is the indicator for event j for observation i
- f_j and S_j are event-specific distributions
- D_{ij} represents the dependency from event i to event j
- λ is a regularization parameter

The MENSA loss function extends the standard negative log-likelihood to multiple event types, with each event having its own likelihood contribution. The regularization term encourages sparsity in the dependency matrix, preventing the model from learning spurious relationships between events.

One of the key benefits of MENSA is its ability to discover and quantify relationships between different event types directly from the data. The learned dependency matrix provides valuable insights into how different events influence each other, which can inform clinical understanding and decision-making.

DeepHit: Discrete-Time Survival Approach

DeepHit, introduced by C. Lee et al. (2018), takes a different approach to survival modeling by directly estimating the discrete probability mass function (PMF) of the survival time. This non-parametric approach offers flexibility in modeling complex survival distributions without assuming a specific parametric form (Gensheimer and Narasimhan, 2019).

DeepHit Model

DeepHit models the probability mass function of survival times directly at discrete time points, with separate outputs for each possible event type:

$$P(T = t, J = j|\mathbf{x}) = \mathrm{PMF}_j(t|\mathbf{x}) \tag{7.12}$$

where T is the survival time, J is the event type, and $\mathrm{PMF}_j(t|\mathbf{x})$ is the probability mass function for event type j at time t given covariates $\mathbf{x}$.

The DeepHit architecture consists of:

- A shared representation network that processes input features
- Event-specific heads that output probability masses for each time point
- A joint optimization framework that combines likelihood and ranking objectives

This architecture allows DeepHit to model complex temporal patterns in the risk of events while handling multiple competing risks. By directly outputting probabilities for each time point and event type, DeepHit provides a highly flexible representation of the survival distribution.

DeepHit Loss Function

The DeepHit loss function combines likelihood, ranking, and calibration components:

$$\mathcal{L}_{DeepHit} = \alpha\mathcal{L}_{likelihood} + \beta\mathcal{L}_{ranking} + \gamma\mathcal{L}_{calibration} \tag{7.13}$$

$$\mathcal{L}_{likelihood} = -\sum_{i=1}^{N}\left[\sum_{j=1}^{J}\delta_{i,j}\log P(T_i = t_i, J = j|\mathbf{x}_i) + (1 - \sum_{j}\delta_{i,j})\log P(T_i > t_i|\mathbf{x}_i)\right] \tag{7.14}$$

where:

- $P(T_i = t, J = j|\mathbf{x}_i)$ is the probability of event j at time t
- $P(T_i > t|\mathbf{x}_i)$ is the probability of survival beyond time t
- α, β, γ are weighting coefficients
- $\mathcal{L}_{ranking}$ is a ranking loss (discussed in Section 7.1.2)
- $\mathcal{L}_{calibration}$ is a calibration loss

The DeepHit loss function combines multiple objectives to achieve both accurate probability estimation and good discriminative performance. The likelihood component ensures proper probability estimation, while the ranking component improves the model's ability to order patients by risk. The calibration component ensures that predicted probabilities match observed frequencies.

A significant advantage of DeepHit is its flexibility in modeling arbitrary survival distributions without parametric assumptions. This makes it particularly suitable for scenarios with complex temporal patterns that may not be well-captured by standard parametric distributions.

7.1.2 Ranking Losses in Survival Analysis

Ranking losses focus on the relative ordering of survival times rather than their absolute values. This perspective is particularly relevant in clinical settings where prioritizing patients according to their risk level is often more important than predicting exact event times. This section explores the most important ranking losses used in survival analysis.

Motivation for Ranking in Survival Analysis

Ranking objectives offer several compelling advantages in survival analysis:

- Clinical decisions often depend more on relative risk than exact timing
- The concordance index (C-index), a standard evaluation metric in survival analysis, is ranking-based
- Ranking is more robust to time scale transformations and censoring patterns
- Ranking directly aligns with prioritization decisions (e.g., transplant waitlists)

- Ranking objectives improve the discrimination ability of models

The concordance index (C-index), introduced by Harrell et al. (1982), serves as the fundamental ranking metric in survival analysis. It measures the proportion of pairs where the model correctly orders patients according to their risk (Antolini, Boracchi, and Biganzoli, 2005).

Concordance Index (C-index)

The concordance index measures the proportion of comparable pairs that are correctly ordered by the model's risk scores:

$$\text{C-index} = \frac{\sum_{i,j} \mathbb{I}(y_i < y_j) \cdot \mathbb{I}(r_i > r_j) \cdot \mathbb{I}(\delta_i = 1)}{\sum_{i,j} \mathbb{I}(y_i < y_j) \cdot \mathbb{I}(\delta_i = 1)} \tag{7.15}$$

where y_i is the observed time for individual i, r_i is the predicted risk score, δ_i is the event indicator, and $\mathbb{I}(\cdot)$ is the indicator function.

While the C-index provides a clear metric for evaluating ranking performance, it presents challenges for optimization:

- It is non-differentiable due to the indicator functions
- It has $O(n^2)$ computational complexity, making it expensive for large datasets
- It only considers pairs where both the order and event status are known

To address these challenges, various differentiable approximations of the C-index have been developed, leading to the ranking losses discussed in the following sections.

Pairwise Ranking: RankNet

RankNet, originally developed for information retrieval by Burges et al. (2005), provides a differentiable approach to pairwise ranking that has been successfully adapted to survival analysis by researchers like Chapfuwa et al. (2018) and Kvamme, Borgan, and Scheel (2019).

RankNet Loss

RankNet formulates ranking as a binary classification problem on pairs, using the sigmoid function to convert score differences into probabilities:

$$p_{ij} = \sigma(s_i - s_j) = \frac{1}{1 + e^{-(s_i - s_j)}} \tag{7.16}$$

$$\mathcal{L}_{ranknet} = -\sum_{i,j} [y_{ij} \log p_{ij} + (1 - y_{ij}) \log(1 - p_{ij})] \tag{7.17}$$

where s_i and s_j are the model's risk scores for individuals i and j, and $y_{ij} = 1$ if individual i should have higher risk than individual j (i.e., $t_i < t_j$).

The RankNet loss essentially applies binary cross-entropy to the probability that individual i has higher risk than individual j. This formulation provides a smooth, differentiable objective that approximates the concordance metric. The sigmoid function serves as a smooth approximation of the indicator function in the C-index calculation.

For survival analysis, RankNet is typically applied to pairs where at least one individual experienced an event, allowing the model to learn from both uncensored and censored data. Various techniques have been developed to efficiently sample informative pairs during training, reducing the computational burden of considering all possible pairs.

SOAP: Statistically Optimal Accelerated Pairwise Loss

The Statistically Optimal Accelerated Pairwise (SOAP) loss, developed by Kvamme, Borgan, and Scheel (2019) specifically for survival analysis, represents an efficient margin-based approach to ranking that focuses on violations of the desired ordering.

SOAP Loss

SOAP employs a margin-based hinge loss formulation:

$$\mathcal{L}_{soap} = \sum_{i,j} \max(0, m - (s_i - s_j) \cdot \text{sign}(t_i - t_j)) \tag{7.18}$$

where m is a margin parameter (typically set to 1), s_i and s_j are risk scores, and t_i and t_j are observed times.

The SOAP loss only penalizes pairs where the margin constraint is violated, meaning that correctly ordered pairs with sufficient margin contribute zero to the loss. This property makes SOAP more computationally efficient than RankNet, as many pairs can be quickly identified as not contributing to the loss.

Kvamme et al. demonstrated that SOAP is concordance-consistent, meaning that it optimizes the C-index asymptotically. They also developed efficient sampling strategies that reduce the computational complexity from $O(n^2)$ to approximately $O(n \log n)$, making it practical for large datasets.

ListMLE: Listwise Maximum Likelihood Estimation

While pairwise approaches like RankNet and SOAP consider pairs in isolation, ListMLE, introduced by Xia et al. (2008) and adapted to survival by Kvamme, Borgan, and Scheel (2019), takes a more global approach by considering entire permutations of samples.

ListMLE Loss

ListMLE uses the Plackett-Luce model to define a probability distribution over permutations:

$$\begin{aligned} \mathcal{L}_{listmle} &= -\sum_i \log P(\pi_i | s_i) \qquad (7.19) \\ &= -\sum_i \sum_{j=1}^{|\pi_i|} \log \frac{\exp(s_{\pi_i(j)})}{\sum_{k=j}^{|\pi_i|} \exp(s_{\pi_i(k)})} \qquad (7.20) \end{aligned}$$

where π_i is an ordered list of samples, $\pi_i(j)$ is the index of the j-th item in the list, and s_i are predicted risk scores.

The ListMLE loss is based on the Plackett-Luce model, which defines a probability distribution over permutations through a sequential selection process. The model considers the probability of selecting each item as the next in the permutation, given the items that have not yet been selected.

Compared to pairwise approaches, ListMLE offers several advantages:

- It considers global ordering consistency rather than just pairwise relationships
- It avoids the issue of conflicting pairwise constraints
- It has better computational efficiency with $O(n \log n)$ complexity
- It often demonstrates superior empirical performance on benchmark datasets

For survival analysis, ListMLE typically uses the observed event times to define the ground truth ordering, with specific techniques to handle censored observations.

SurvRNC: Survival Rank-N-Contrast Loss

The Survival Rank-N-Contrast (SurvRNC) loss, introduced by Kvamme (2021), represents an innovative approach that combines ranking and contrastive learning principles to achieve both good performance and computational efficiency.

SurvRNC Loss

SurvRNC applies contrastive learning to survival ranking:

$$\mathcal{L}_{survrnc} = -\frac{1}{N}\sum_{i} \log \frac{e^{sim(a_i,p_i)/\tau}}{\sum_j e^{sim(a_i,j)/\tau}} \tag{7.21}$$

where $sim(a, b)$ is the similarity between embeddings, a_i is an anchor sample, p_i is a positive sample (similar time), and τ is a temperature parameter.

Inspired by the SimCLR framework in computer vision, SurvRNC applies contrastive learning principles to survival analysis. The core idea is to learn representations where patients with similar survival times are positioned close together in the embedding space, while those with different times are pushed apart.

This approach offers several advantages:

- It reduces computational complexity from $O(n^2)$ to roughly linear in the number of samples
- It creates meaningful representations that capture the temporal structure of survival times
- It handles censoring naturally through the similarity definition
- It scales well to very large datasets where pairwise approaches become impractical

The SurvRNC loss demonstrates how principles from other areas of machine learning can be effectively adapted to survival analysis, providing new approaches to the ranking problem that offer both performance and computational benefits.

Efficient Ranking Implementations

The naive implementation of ranking losses involves comparing all possible pairs of samples, resulting in $O(n^2)$ complexity that becomes prohibitive for large datasets. Several techniques have been developed to improve the efficiency of ranking losses:

- **Sample-Based Ranking**: Using mini-batches to approximate the full ranking loss
- **Event-Specific Ranking**: Separating ranking by event type in competing risks settings
- **Stratified Sampling**: Focusing on informative pairs that are likely to contribute to the loss
- **Computational Tricks**: Utilizing vectorized operations and GPU acceleration

Recent optimizations have reduced the complexity of ranking losses to $O(n \log n)$ or even $O(n)$ in some cases, making them practical for large-scale applications. Benchmark studies show that these optimized implementations can achieve performance very close to the exact methods while requiring orders of magnitude less computation time.

7.1.3 Regression Losses for Survival Analysis

Regression losses directly target the prediction of time-to-event as a continuous value. These approaches provide interpretable outputs in time units, which can be valuable for clinical applications where precise timing estimates are needed. This section explores the adaptation of standard regression losses to handle censored survival data.

Motivation for Regression Approaches

Regression-based approaches to survival analysis offer several advantages:

- They provide direct predictions of time-to-event in interpretable units
- They facilitate comparison with traditional clinical risk models
- They offer straightforward implementation using standard machine learning frameworks
- They serve as building blocks for more complex survival models
- They can be easily combined with other loss functions in multi-task settings

The main challenge in applying regression losses to survival data is the handling of censored observations, where the true event time is unknown. Various strategies have been developed to address this challenge, leading to several adaptations of standard regression losses for survival analysis.

L1 Loss with Censoring

The L1 loss (mean absolute error) measures the absolute difference between predicted and true values. For survival analysis, several adaptations have been developed to handle censored observations (X. Zhong and Jeong, 2020; Gensheimer and Narasimhan, 2019).

L1 Loss Variants for Survival

Three main approaches for adapting L1 loss to censored data:

$$\mathcal{L}_{uncensored} = \frac{1}{N}\sum_{i=1}^{N} |t_i - \hat{y}_i| \cdot \mathbb{I}(\delta_i = 1) \tag{7.22}$$

$$\mathcal{L}_{hinge} = \frac{1}{N}\sum_{i=1}^{N} [|t_i - \hat{y}_i| \cdot \mathbb{I}(\delta_i = 1) + \max(0, t_i - \hat{y}_i) \cdot \mathbb{I}(\delta_i = 0)] \tag{7.23}$$

$$\mathcal{L}_{margin} = \frac{1}{N}\left[\sum_{i:\delta_i=1} |t_i - \hat{y}_i| + \sum_{i:\delta_i=0} w_i|\tilde{t}_i - \hat{y}_i|\right] \tag{7.24}$$

where δ_i is the event indicator, w_i are weights based on Kaplan-Meier estimates, and $\tilde{t}_i$ is the expected event time given censoring at t_i.

The uncensored approach simply ignores censored observations, only calculating the loss for samples with observed events. This is the simplest approach but can lead to biased estimates, especially when censoring is informative.

The hinge approach applies a one-sided penalty for censored observations, penalizing predictions that are smaller than the censoring time. This approach recognizes that for censored observations, we only know that the true event time is greater than the observed censoring time.

The margin approach uses imputation and weighting to incorporate censored observations. It estimates the expected event time for censored observations based on the conditional survival distribution and assigns weights based on the uncertainty of these estimates.

MSE Loss for Survival Analysis

The Mean Squared Error (MSE) loss, which penalizes the squared difference between predicted and true values, has also been adapted for survival analysis using strategies similar to those for L1 loss (Biganzoli et al., 2001).

MSE Loss Variants for Survival

Two common adaptations of MSE for censored data:

$$\mathcal{L}_{MSE-uncensored} = \frac{1}{N}\sum_{i=1}^{N}(t_i - \hat{y}_i)^2 \cdot \mathbb{I}(\delta_i = 1) \tag{7.25}$$

$$\mathcal{L}_{MSE-margin} = \frac{1}{N}\left[\sum_{i:\delta_i=1} (t_i - \hat{y}_i)^2 + \sum_{i:\delta_i=0} w_i(\tilde{t}_i - \hat{y}_i)^2\right] \tag{7.26}$$

where the symbols have the same meaning as in the L1 variants.

Compared to L1 loss, MSE more heavily penalizes large deviations due to the squaring operation. This makes it more sensitive to outliers but often results in smoother gradients. MSE also has connections to maximum likelihood estimation under Gaussian noise assumptions.

The choice between L1 and MSE depends on the specific application and the characteristics of the data. L1 is generally more robust to outliers, while MSE often leads to more stable optimization due to its smoother derivatives.

Quantile Loss for Survival

Quantile regression provides a richer description of the relationship between predictors and the response variable by modeling different quantiles of the conditional distribution (Koenker and Geling, 2001). This approach has been extended to survival analysis to provide prediction intervals rather than just point estimates (Tagasovska and Lopez-Paz, 2019).

Quantile Loss for Survival

The quantile loss function and its adaptations for survival:

$$\rho_q(y, \hat{y}) = q \cdot \max(0, y - \hat{y}) + (1 - q) \cdot \max(0, \hat{y} - y) \tag{7.27}$$

$$\mathcal{L}_{q-uncensored} = \frac{1}{N} \sum_{i=1}^{N} \rho_q(t_i, \hat{y}_i) \cdot \mathbb{I}(\delta_i = 1) \tag{7.28}$$

$$\mathcal{L}_{q-margin} = \frac{1}{N} \left[\sum_{i:\delta_i=1} \rho_q(t_i, \hat{y}_i) + \sum_{i:\delta_i=0} w_i \rho_q(\tilde{t}_i, \hat{y}_i) \right] \tag{7.29}$$

where q is the quantile level (e.g., 0.5 for median), and other symbols follow previous definitions.

The quantile loss function applies asymmetric penalties for over-prediction and under-prediction, with the degree of asymmetry determined by the quantile level q. This property allows the model to target specific quantiles of the conditional distribution rather than just the mean or median.

By training multiple quantile models (or a single model with multiple quantile outputs), practitioners can obtain prediction intervals for survival times. This provides a more complete picture of prediction uncertainty compared to point estimates, which is particularly valuable in clinical applications where understanding the range of possible outcomes is important.

Regression Losses: Applications and Limitations

Regression losses for survival analysis offer interpretable predictions in time units, but they come with both strengths and limitations that should be considered when choosing an approach.

Strengths of regression-based approaches include:

- Interpretable outputs in familiar time units
- Direct comparisons with clinical estimates
- Well-established statistical properties
- Straightforward uncertainty quantification with quantile approaches
- Compatibility with standard neural architectures

Limitations include:

- They don't directly optimize ranking performance

- They can be sensitive to censoring patterns
- They may struggle to capture multi-modal distributions
- They often require careful handling of right-skewed distributions
- They may need to be combined with other losses for optimal performance

In practice, regression losses are often combined with ranking or likelihood-based losses in a multi-task learning framework to benefit from the strengths of each approach while mitigating their limitations.

7.1.4 Classification Losses for Survival Analysis

Classification approaches transform survival analysis into a classification problem, either predicting the probability of an event occurring before a specific time threshold or discretizing time into intervals and predicting the probability of an event in each interval. These approaches leverage the well-established classification literature while adapting it to handle censoring.

Binary Cross-Entropy for Survival

The binary cross-entropy (BCE) loss, fundamental to binary classification, has been adapted to survival analysis by defining the classification task as predicting whether an event occurs before a specified time threshold (Graf et al., 1999).

Binary Cross-Entropy for Survival

The BCE loss and its adaptations for censored data:

$$\mathcal{L}_{CE-uncensored} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \hat{p}_i + (1-y_i)\log(1-\hat{p}_i)\right]\cdot v_i \tag{7.30}$$

$$\mathcal{L}_{CE-margin} = -\frac{1}{N}\Bigg[\sum_{i:\delta_i=1}\left(y_i \log \hat{p}_i + (1-y_i)\log(1-\hat{p}_i)\right) + \tag{7.31}$$

$$\sum_{i:\delta_i=0} w_i\left(\tilde{y}_i \log \hat{p}_i + (1-\tilde{y}_i)\log(1-\hat{p}_i)\right)\Bigg] \tag{7.32}$$

where:

- $y_i = \mathbb{I}(t_i \leq T \text{ and } \delta_i = 1)$ is the binary target
- $v_i = \mathbb{I}(t_i > T \text{ or } \delta_i = 1)$ identifies relevant samples
- $\tilde{y}_i = P(T_i \leq T | T_i > t_i)$ is the conditional probability for censored samples
- w_i are weights for censored samples

The key challenge in applying BCE to survival data is handling censored observations. For uncensored observations with events, the target is clear: $y_i = 1$ if the event occurred before time T, and $y_i = 0$ otherwise. For censored observations, different strategies can be employed:

1. The uncensored approach includes only samples where the binary outcome is known (either the event occurred before time T, or the individual was followed beyond time T without an event)

2. The margin approach uses imputation and weighting, estimating the conditional probability that an event would occur before time T given that it hasn't occurred by the censoring time.

Binary cross-entropy for survival is particularly useful for risk stratification at clinically meaningful timepoints, such as 1-year, 5-year, or 10-year risk. It also serves as a common auxiliary task in multi-task survival models.

Multi-Class Classification for Survival

Multi-class classification approaches to survival analysis divide the time axis into discrete intervals and predict the probability of an event occurring in each interval. This approach, exemplified by models like DeepHit, provides a flexible non-parametric representation of the survival distribution.

Multi-Class Classification for Survival

The multi-class classification loss for survival:

$$\mathcal{L}_{multi} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\log\hat{p}_{ik} \tag{7.33}$$

$$y_{ik} = \begin{cases} 1 & \text{if } t_i \in [t_{k-1}, t_k) \text{ and } \delta_i = 1 \\ 0 & \text{otherwise} \end{cases} \tag{7.34}$$

$$\mathcal{L}_{censored} = -\frac{1}{N}\sum_{i=1}^{N}\left[\delta_i \sum_{k=1}^{K} y_{ik}\log\hat{p}_{ik} + (1-\delta_i)\log\left(\sum_{j=k_i}^{K}\hat{p}_{ij}\right)\right] \tag{7.35}$$

where:

- K is the number of time intervals
- $\hat{p}_{ik}$ is predicted probability of event in interval k for patient i
- k_i is the interval containing censoring time t_i

The multi-class approach assigns each uncensored observation to the interval containing its event time. For censored observations, the approach recognizes that the event could occur in any interval after the censoring time. The second formulation addresses this by summing the probabilities of all possible future intervals.

This approach offers several advantages:

- It provides a flexible non-parametric representation of the survival distribution
- It naturally extends to competing risks by having separate outputs for each event type
- It preserves the ordering relationship between adjacent time periods
- It can be implemented using standard multi-class classification architectures

The main challenge is the selection of appropriate time intervals. Too few intervals may not capture the temporal dynamics accurately, while too many can lead to sparse data and overfitting.

Various strategies have been proposed for optimal interval selection, including equal-width intervals, equal-frequency intervals, and data-driven approaches.

Classification for Survival: Advantages and Limitations

Classification-based approaches to survival analysis offer several advantages but also come with important limitations.

Advantages include:

- Simpler implementation than full survival models
- Ability to leverage standard classification architectures and techniques
- Good compatibility with modern deep learning frameworks
- Direct addressing of clinically relevant time horizons
- Easier interpretation for non-statisticians
- Facilitating integration with other prediction tasks

Limitations include:

- Loss of continuous time information due to discretization
- Need for separate models or outputs for different time thresholds
- Sensitivity to the choice of time intervals
- Potential inefficiency for long-term predictions
- Challenges with heavily censored data
- Difficulties in comparison with traditional survival models

Despite these limitations, classification approaches remain popular in practice, especially for applications where risk stratification at specific timepoints is the primary goal or where integration with existing classification systems is important.

7.1.5 Auxiliary Losses and Loss Balancing

Beyond the core survival, ranking, regression, and classification losses, several auxiliary losses and techniques have been developed to address specific challenges in survival analysis, such as class imbalance and the combination of multiple loss components.

Focal Loss for Survival Analysis

Focal loss, introduced by Lin et al. (2017) for object detection in computer vision, addresses class imbalance by down-weighting the contribution of well-classified examples and focusing on difficult cases. This concept has been adapted to survival analysis to deal with rare events and imbalanced prediction problems (Steiner et al., 2021; Fotso, 2018; Wiegrebe et al., 2023).

Focal Loss for Survival

The focal loss modifies standard losses by adding a modulating factor:

$$\mathcal{L}_{focal} = -\sum_{i=1}^{N}(1 - p_t)^\gamma \log(p_t) \tag{7.36}$$

$$p_t = \begin{cases} p_i & \text{if } y_i = 1 \\ 1 - p_i & \text{otherwise} \end{cases} \tag{7.37}$$

where $\gamma \geq 0$ is the focusing parameter (typically $\gamma \in [1, 5]$), and p_t is the model's predicted probability for the correct class.

The focal loss introduces a modulating factor $(1 - p_t)^\gamma$ that reduces the contribution of examples that are already well-classified (high p_t) and increases the focus on hard examples (low p_t). The parameter γ controls the extent of this modulation, with higher values placing more emphasis on hard examples.

In survival analysis, focal loss can be applied to various base losses:

- **Focal NLL:** Modifying the negative log-likelihood to focus on uncertain predictions

$$\mathcal{L}_{focal-NLL} = -\sum_{i=1}^{N}(1 - p_{surv,i})^\gamma [\delta_i \log h(t_i|x_i) + \log S(t_i|x_i)] \tag{7.38}$$

- **Focal ranking:** Applying focal weighting to ranking losses

$$\mathcal{L}_{focal-rank} = -\sum_{i,j \in \mathcal{P}} (1 - \sigma(r_j - r_i))^\gamma \log \sigma(r_j - r_i) \tag{7.39}$$

Recent studies have shown that focal loss can improve performance for rare events in survival analysis, with improvements of 5-15% reported in some benchmarks. The largest gains are typically observed for the rarest event types or time intervals, where standard losses might not provide sufficient learning signal.

Loss Balancing Strategies

Many survival models employ multiple loss components to capture different aspects of the prediction task. Balancing these components effectively is crucial for successful training and optimal performance.

Loss Balancing

A weighted combination of multiple loss components:

$$\mathcal{L}_{total} = \sum_{i=1}^{K} w_i \mathcal{L}_i \tag{7.40}$$

where w_i are the weights for each loss component $\mathcal{L}_i$.

Several strategies exist for determining the weights in a multi-component loss:

- **Fixed Weights:** Using constant predefined weights (e.g., $w_i = \alpha_i$)
- **Adaptive Weights:** Adjusting weights based on training progress
- **Uncertainty Weights:** Weighting by the inverse of task uncertainty ($w_i = \frac{1}{2\sigma_i^2}$)
- **Gradient-Based:** Weighting based on gradient magnitudes ($w_i = f(||\nabla\mathcal{L}_i||)$)
- **Scale-Normalized:** Normalizing by the scale of each loss ($w_i = \frac{\alpha_i}{\text{scale}(\mathcal{L}_i)}$)
- **Learned Weights:** Treating weights as trainable parameters

Advanced methods like the uncertainty weighting proposed by Kendall, Gal, and Cipolla (2018) learn optimal weights during training by casting the problem as learning to optimize multiple objectives under uncertainty. These approaches can significantly improve performance compared to fixed weighting schemes, especially for complex models with multiple diverse loss components (Fotso, 2018).

7.1.6 Momentum Contrast for Survival Analysis

7.2 Momentum Contrast for Survival Analysis

Section Overview

This section covers:

- The challenge of event sparsity in survival analysis data
- Momentum Contrast (MoCo) as a solution for high censoring rates
- Three implementation modes with different capabilities
- Mathematical foundations and optimization characteristics
- Practical considerations for applying MoCo effectively

Survival analysis often faces a fundamental challenge that limits the effectiveness of traditional approaches: the sparsity of observed events due to censoring (Kaplan and Meier, 1958). This section introduces Momentum Contrast (MoCo), a technique adapted from computer vision and self-supervised learning (He et al., 2020) to address this challenge in survival analysis contexts.

7.2.1 Event Sparsity and Censoring Challenges

Survival datasets, particularly in medical domains, frequently exhibit high censoring rates where a significant proportion of observations never experience the event of interest during the study period (Ibrahim, Chen, and Sinha, 2001). This censoring creates several challenges for model training:

- Mini-batches may contain few or no events, leading to unstable gradients
- The effective sample size for learning event-specific patterns is reduced
- Models tend to overfit to the majority class (censored observations)
- Loss functions become dominated by censoring patterns rather than event dynamics

Event Sparsity in Mini-Batches

In a typical survival dataset with censoring rate $c \in [0, 1]$, the expected number of events in a mini-batch of size B is:

$$\mathbb{E}[\text{events}] = B \cdot (1 - c) \tag{7.41}$$

For high censoring rates (e.g., $c = 0.9$), a batch size of $B = 32$ yields only $\mathbb{E}[\text{events}] = 3.2$ events per batch.

This event sparsity problem becomes especially pronounced in datasets with censoring rates exceeding 70%, which is common in many clinical applications (Ranganath et al., 2016). As illustrated in Figure 7.1, high censoring rates lead to few events per mini-batch, compromising the stability and effectiveness of the learning process.

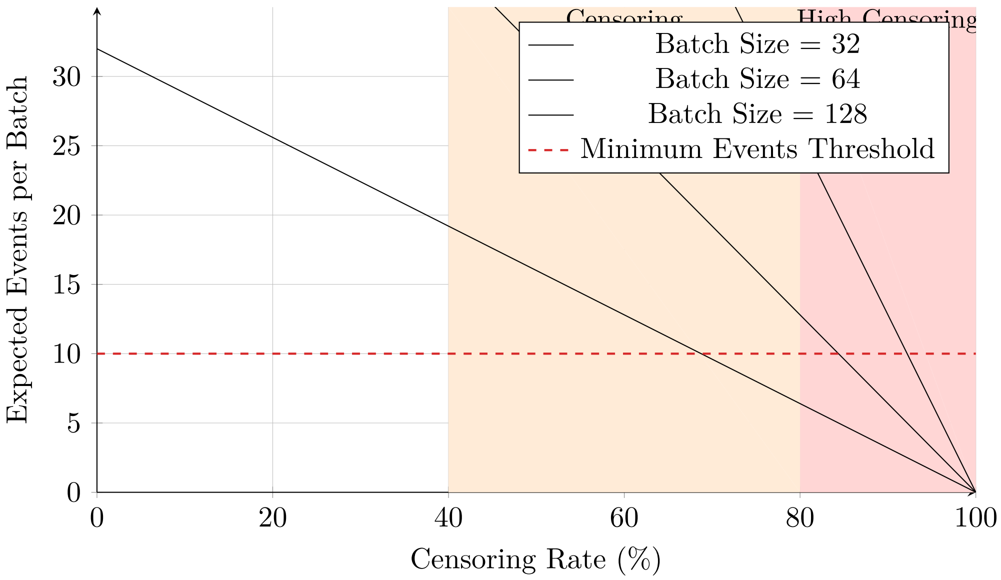


Figure 7.1: The relationship between censoring rate and expected events per mini-batch. As censoring increases, the number of events decreases linearly, falling below practical thresholds for effective learning.

7.2.2 Momentum Contrast: Core Principles

Momentum Contrast (MoCo) addresses the event sparsity challenge by maintaining a queue of past sample embeddings along with their corresponding event information. This approach effectively creates an "augmented batch" that contains substantially more events than would be present in a single mini-batch.

Momentum Contrast Buffer

MoCo maintains two synchronized queues:

- $\mathcal{Q}_{\text{emb}} = \{e_1, e_2, \ldots, e_K\}$: Queue of embeddings
- $\mathcal{Q}_{\text{ref}} = \{r_1, r_2, \ldots, r_K\}$: Queue of corresponding references (event indicators and times)

where K is the buffer size, e_i are sample embeddings, and r_i are the corresponding reference values.

The key innovation of MoCo is its ability to utilize past computations to enhance the current learning step. Unlike traditional experience replay techniques, MoCo maintains embedding-level information rather than raw inputs, allowing for efficient memory usage and seamless integration with any survival loss function.

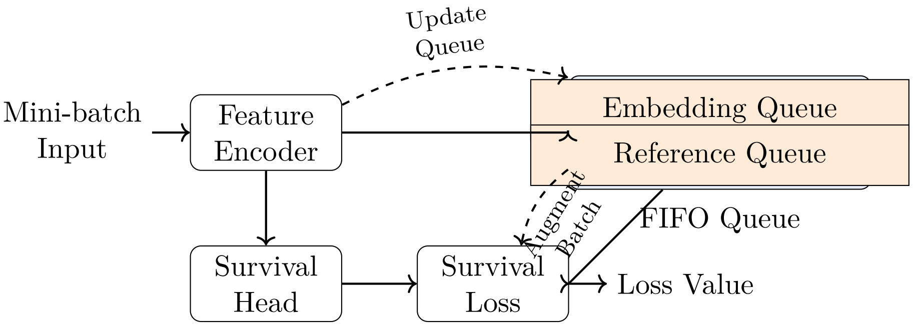


Figure 7.2: Overview of Momentum Contrast (MoCo) for survival analysis. Past embeddings are stored in a FIFO queue and combined with the current batch during loss computation.

The MoCo process involves the following steps during each training iteration:

1. Process the current mini-batch through the feature encoder
2. Compute embeddings and predictions for the current samples
3. Retrieve past embeddings from the buffer
4. Combine current and buffered embeddings for loss computation
5. Update the buffer with the current batch's embeddings
6. Adjust buffer parameters based on training progress

This approach effectively increases the "event density" during training, leading to more stable gradients and better model convergence, especially for highly censored datasets.

7.2.3 Mathematical Formulation

Standard MoCo Loss

The standard MoCo loss combines two components: a loss computed on the current batch and a loss computed on the combined batch (current + buffer).

Standard MoCo Survival Loss

The MoCo-enhanced survival loss is defined as:

$$\begin{aligned}
\mathcal{L}_{\text{MoCo}} &= w_{\text{batch}} \cdot \mathcal{L}_{\text{base}}(X_{\text{batch}}, Y_{\text{batch}}) + w_{\text{combined}} \cdot \mathcal{L}_{\text{base}}(X_{\text{combined}}, Y_{\text{combined}}) && (7.42)\\
X_{\text{combined}} &= [X_{\text{batch}}; X_{\text{buffer}}] && (7.43)\\
Y_{\text{combined}} &= [Y_{\text{batch}}; Y_{\text{buffer}}] && (7.44)
\end{aligned}$$

where:

- $\mathcal{L}_{\text{base}}$ is any base survival loss function
- w_{batch} and w_{combined} are weight parameters
- X_{batch} and Y_{batch} are the current batch inputs and references
- X_{buffer} and Y_{buffer} are the buffer embeddings and references
- $[;]$ denotes concatenation

The buffer update follows a First-In-First-Out (FIFO) queue mechanism:

$$\begin{aligned}
\mathcal{Q}_{\text{emb}} &\leftarrow [\mathcal{Q}_{\text{emb}}[B:]; X_{\text{batch}}] && (7.45)\\
\mathcal{Q}_{\text{ref}} &\leftarrow [\mathcal{Q}_{\text{ref}}[B:]; Y_{\text{batch}}] && (7.46)
\end{aligned}$$

where B is the batch size and the notation $\mathcal{Q}[B:]$ indicates all elements from index B to the end of the queue.

Dynamic Weight Formulation

The Dynamic Weight MoCo variant introduces time-dependent weights that gradually shift emphasis from the current batch to the buffer as training progresses.

Dynamic Weight MoCo Loss

The Dynamic Weight MoCo loss uses interpolated weights:

$$\begin{aligned}
w_{\text{batch}}(t) &= w_{\text{batch}}^{\text{initial}} + \frac{t}{T} \cdot (w_{\text{batch}}^{\text{final}} - w_{\text{batch}}^{\text{initial}}) && (7.47)\\
w_{\text{buffer}}(t) &= w_{\text{buffer}}^{\text{initial}} + \frac{t}{T} \cdot (w_{\text{buffer}}^{\text{final}} - w_{\text{buffer}}^{\text{initial}}) && (7.48)
\end{aligned}$$

where:

- t is the current training step
- T is the warmup period
- $w_{\text{batch}}^{\text{initial}}$ and $w_{\text{batch}}^{\text{final}}$ are the initial and final batch weights
- $w_{\text{buffer}}^{\text{initial}}$ and $w_{\text{buffer}}^{\text{final}}$ are the initial and final buffer weights

This formulation allows the model to initially focus on learning from the current batch, gradually incorporating the buffer as training progresses. A typical configuration starts with $w_{\text{batch}}^{\text{initial}} = 1.0$, $w_{\text{buffer}}^{\text{initial}} = 0.0$ and transitions to $w_{\text{batch}}^{\text{final}} = 0.5$, $w_{\text{buffer}}^{\text{final}} = 1.0$.

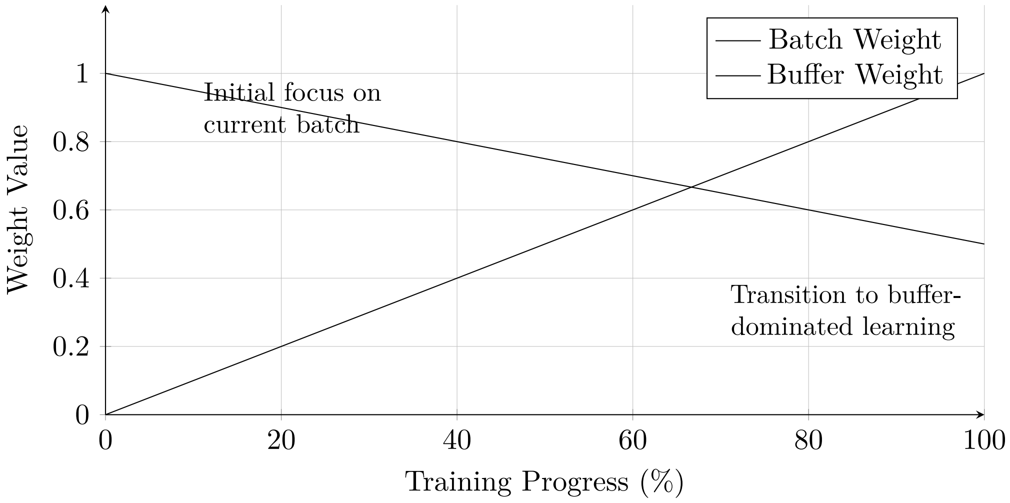


Figure 7.3: Dynamic weight interpolation in MoCo. Weights for the current batch and buffer components change over time, allowing for gradual transition from batch-focused to buffer-enhanced learning.

Adaptive Buffer Formulation

The Adaptive Buffer variant monitors training dynamics and adjusts buffer usage based on loss variance, providing an automatic mechanism for handling different levels of training instability.

Adaptive MoCo Buffer Adjustment

The adaptive buffer adjustment is based on loss variance:

$$v_t = \text{Var}(\mathcal{L}_{t-w:t}) \tag{7.49}$$

$$\Delta v_t = \frac{v_t - v_{t-1}}{v_{t-1}} \tag{7.50}$$

$$K_{t+1} = \begin{cases} \min(K_t \cdot \gamma^+, K_{\max}) & \text{if } \Delta v_t > \tau^+ \\ \max(K_t \cdot \gamma^-, K_{\min}) & \text{if } \Delta v_t < \tau^- \text{ and } t > t_{\min} \\ K_t & \text{otherwise} \end{cases} \tag{7.51}$$

where:

- v_t is the variance of the loss over a window of size w
- Δv_t is the relative change in variance
- K_t is the current buffer size
- γ^+ and γ^- are growth and shrinkage factors (typically 1.5 and 0.75)

- τ^+ and τ^- are positive and negative thresholds (typically 0.5 and -0.5)
- K_{max} and K_{min} are the maximum and minimum buffer sizes
- t_{min} is a minimum number of iterations before allowing buffer reduction

This adaptive mechanism enables the model to automatically find the optimal buffer size based on training dynamics. When the loss becomes unstable (high variance), the buffer size increases to provide more stable gradients. When the loss stabilizes, the buffer can be reduced to improve computational efficiency.

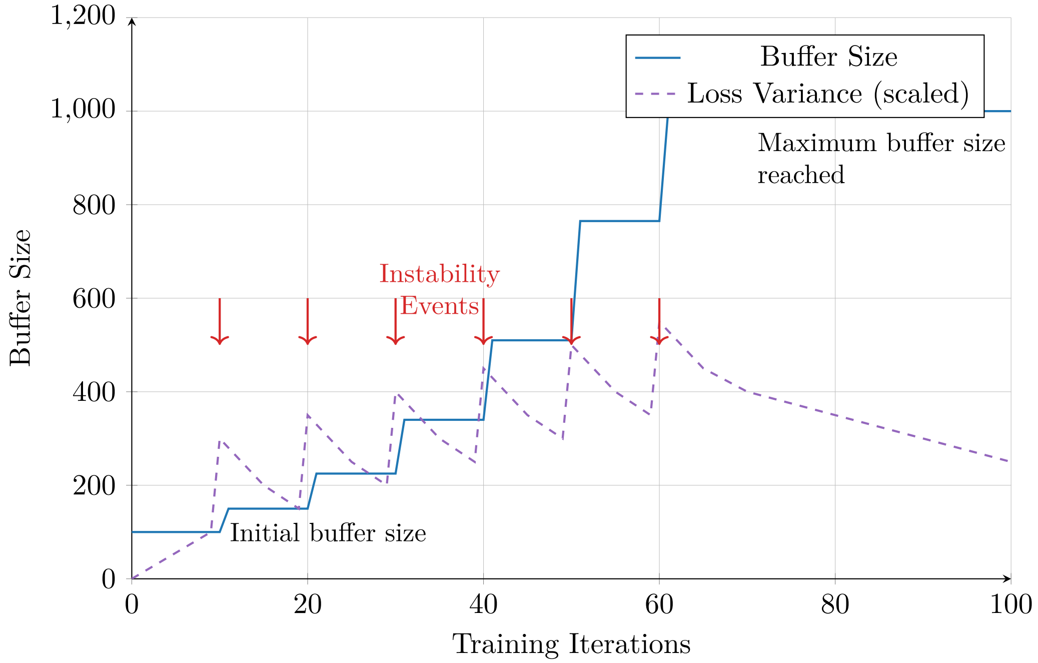


Figure 7.4: Adaptive buffer size adjustment during training. The buffer size increases in response to detected loss instability events, eventually reaching its maximum capacity.

7.2.4 Implementation Variants and Selection

SAT provides three MoCo implementation variants, each designed for specific scenarios:

Standard MoCoSurvivalLoss

The base implementation is suitable for datasets with moderate censoring rates (40-70%). It uses fixed weights for batch and buffer components and optional dynamic buffer growth.

Standard MoCo Configuration

```
moco_buffer_size: 512
moco_initial_buffer_size: 128
moco_use_buffer: True
moco_dynamic_buffer: True
moco_batch_weight: 1.0
moco_buffer_weight: 1.0
```

DynamicWeightMoCoLoss

This variant is ideal for high censoring rates (70-85%). It gradually transitions from batch-focused to buffer-enhanced learning during training.

Dynamic Weight MoCo Configuration

```
moco_buffer_size: 1024
moco_initial_buffer_size: 256
moco_use_buffer: True
moco_dynamic_buffer: True
moco_batch_weight: 1.0
moco_buffer_weight: 1.0
moco_initial_batch_weight: 1.0
moco_final_batch_weight: 0.5
moco_initial_buffer_weight: 0.0
moco_final_buffer_weight: 1.0
moco_warmup_steps: 1000
```

AdaptiveMoCoLoss

The most advanced implementation, suitable for very high censoring (>85%) or competing risks. It automatically adjusts buffer usage based on loss variance monitoring.

Adaptive MoCo Configuration

```
moco_buffer_size: 2048
moco_initial_buffer_size: 256
moco_use_buffer: True
moco_dynamic_buffer: True
moco_batch_weight: 1.0
moco_buffer_weight: 1.0
moco_adaptive_buffer: True
moco_track_variance: True
moco_variance_window: 10
moco_variance_threshold: 0.15
```

7.2.5 MoCo Recommender System

To simplify the selection of appropriate MoCo parameters, SAT includes a dedicated recommendation tool that analyzes dataset characteristics and training configuration to provide optimal settings.

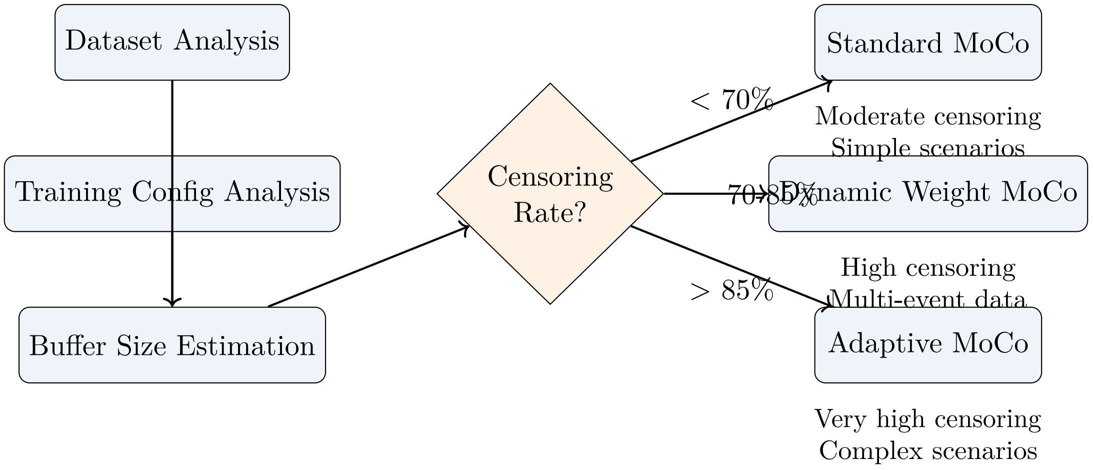


Figure 7.5: Decision process for the MoCo recommender system. The system analyzes dataset and training characteristics to suggest the appropriate MoCo variant and optimal parameter settings.

The recommender considers several factors when providing suggestions:

- **Censoring rate**: Primary factor in determining buffer size and MoCo variant
- **Sample count**: Influences maximum reasonable buffer size
- **Batch size**: Determines the base unit for buffer scaling
- **Event types**: Multi-event datasets typically benefit from more advanced variants
- **Hardware**: CPU vs. GPU considerations for memory usage
- **Expected events per batch**: Key metric for stability assessment

Buffer Size Estimation

The recommended buffer size is estimated as:

$$B_{\text{events}} = B \cdot (1 - c) \tag{7.52}$$

$$R_{\text{required}} = \frac{E_{\min}}{B_{\text{events}}} \tag{7.53}$$

$$K_{\text{recommended}} = \min(\max(B \cdot (R_{\text{required}} - 1), B), K_{\max}) \tag{7.54}$$

where:

- B is the batch size
- c is the censoring rate
- B_{events} is the expected number of events per batch
- $E_{\min}$ is the minimum desired events per effective batch
- R_{required} is the required ratio of effective batch to original batch

- $K_{\text{recommended}}$ is the recommended buffer size
- $K_{\max}$ is the maximum reasonable buffer size (typically $\min(N/2, 4096)$ where N is dataset size)

7.2.6 Experimental Results

Empirical evaluations on multiple survival datasets demonstrate the effectiveness of MoCo in improving model performance, particularly for highly censored datasets.

Table 7.1: Performance comparison of survival models with and without MoCo enhancements on datasets with varying censoring rates.

Dataset	Censoring	Baseline	Standard	Dynamic	Adaptive
	Rate	**C-index**	**MoCo**	**MoCo**	**MoCo**
METABRIC	42.8%	0.645	0.661	0.658	0.659
SUPPORT	68.1%	0.612	0.639	0.648	0.650
SEER	74.5%	0.591	0.607	0.631	0.633
Rotterdam	88.9%	0.563	0.577	0.601	0.624

The results in Table 7.1 show several key patterns:

- All MoCo variants improve performance over the baseline
- The benefit of MoCo increases with the censoring rate
- For moderately censored datasets (METABRIC), Standard MoCo is sufficient
- For highly censored datasets (SEER), Dynamic MoCo provides additional gains
- For very highly censored datasets (Rotterdam), Adaptive MoCo shows the strongest performance

7.2.7 Practical Considerations

When implementing MoCo for survival analysis, several practical considerations should be taken into account:

Memory Usage

The buffer size directly impacts memory requirements. For a model with embedding dimension d, a buffer of size K, and reference dimension r, the additional memory requirement is approximately:

$$M_{\text{additional}} \approx K \cdot (d + r) \cdot \text{sizeof(float)} \tag{7.55}$$

For large models with limited GPU memory, consider:

- Starting with smaller buffer sizes and enabling dynamic growth
- Using CPU storage for the buffer if needed
- Monitoring memory usage and adjusting parameters accordingly

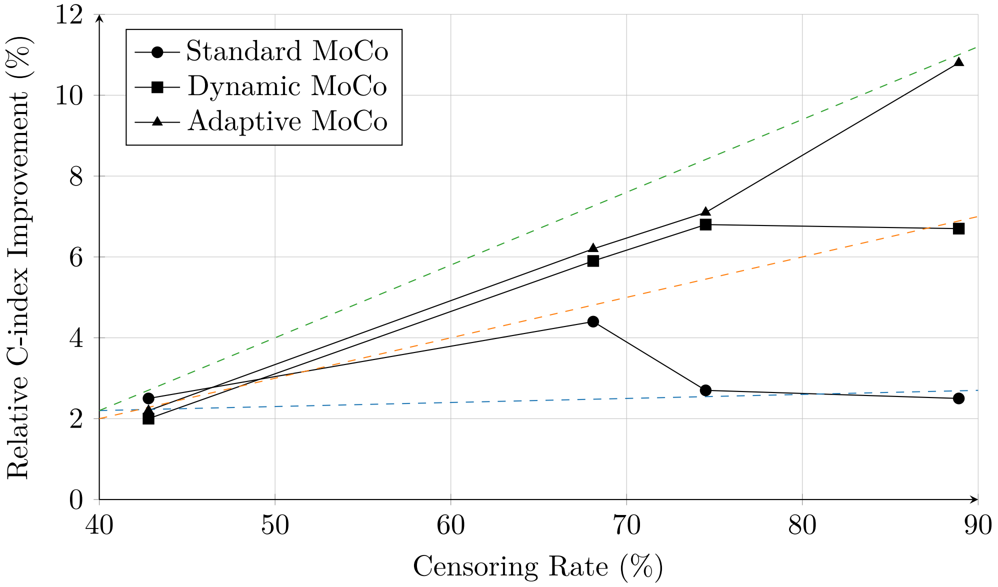


Figure 7.6: Relative improvement in concordance index (C-index) with different MoCo variants across datasets with varying censoring rates. The performance gap between variants widens as censoring increases.

Batch Size and Buffer Size Relationship

The relationship between batch size and buffer size is critical for performance:

- Larger batch sizes reduce the need for large buffers
- For a fixed target of effective events, the buffer size scales inversely with batch size
- For GPUs, maximize batch size within memory constraints
- For CPUs, smaller batches with larger buffers often work better

Integration with Other Loss Functions

MoCo works as a wrapper around any base survival loss function:

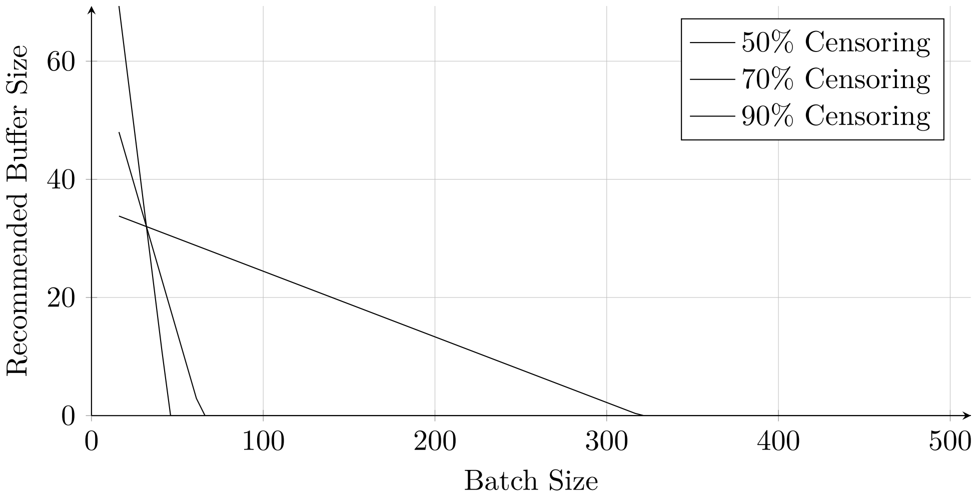


Figure 7.7: Relationship between batch size and recommended buffer size for different censoring rates, assuming a target of 32 events in the effective batch.

Integrating MoCo with Different Loss Functions

```
# With standard NLL-PCH loss
base_loss = NLLPCHLoss(...)
moco_loss = MoCoSurvivalLoss(base_loss=base_loss, ...)

# With ranking loss
base_loss = RankNetLoss(...)
moco_loss = DynamicWeightMoCoLoss(base_loss=base_loss, ...)

# With DSM loss
base_loss = DSMLoss(...)
moco_loss = AdaptiveMoCoLoss(base_loss=base_loss, ...)
```

Computational Overhead

While MoCo improves model performance, it introduces computational overhead:

- **Memory overhead**: $O(K \cdot (d + r))$ additional memory
- **Computation overhead**: Additional forward pass for combined data, typically $O(K + B)$ vs. $O(B)$ for the standard approach
- **Queue management overhead**: FIFO operations, typically negligible compared to neural network computation

In practice, the overhead is often justified by the significant performance improvements, especially for challenging datasets with high censoring rates.

7.2.8 Conclusion

Momentum Contrast (MoCo) represents a significant advancement in survival analysis, particularly for addressing the fundamental challenge of event sparsity due to censoring. By maintaining a queue of past embeddings, MoCo effectively creates an "augmented batch" with substantially more events, leading to more stable gradients and better model convergence.

The three implementation variants (Standard, Dynamic, and Adaptive) provide flexible options for different scenarios, while the recommender tool simplifies configuration by analyzing dataset characteristics and suggesting optimal parameters. Empirical evaluations demonstrate that MoCo consistently improves model performance, with the benefits becoming more pronounced as censoring rates increase.

As survival analysis continues to evolve in the deep learning era, techniques like MoCo that address core challenges such as censoring and event sparsity will play an increasingly important role in developing robust and accurate predictive models for time-to-event data.

7.2.9 Summary and Best Practices

Loss functions represent a critical component in survival analysis models, directly influencing both the learning process and the resulting model behavior (C. Lee et al., 2018; Kvamme, Borgan, and Scheel, 2019; Nagpal, Li, and Dubrawski, 2021). This chapter has explored a wide range of loss functions, from traditional survival likelihoods to modern ranking and regression adaptations (Harrell et al., 1982; Ibrahim, Chen, and Sinha, 2001; Kvamme, 2021).

When selecting a loss function for a survival analysis task, consider the following factors:

- **Task objectives:** Different losses target different aspects of survival prediction
  - If accurate time prediction is the primary goal, consider regression losses or parametric survival models
  - If ranking patients by risk is most important, prioritize ranking losses
  - If calibrated probabilities are needed, ensure likelihood components are included
- **Data characteristics:**
  - Heavy censoring may favor approaches like DSM or likelihood-based methods
  - Class imbalance might benefit from focal loss adaptations
  - Competing risks settings call for specialized approaches like MENSA or DeepHit
- **Computational considerations:**
  - For large datasets, consider optimized implementations of ranking losses
  - Balance computational complexity against modeling power
  - Leverage efficient mini-batch approximations when appropriate
- **Combined approaches:**
  - Often, the best performance comes from combining multiple loss components
  - Use appropriate loss balancing strategies when combining components
  - Consider multi-task learning frameworks that share representations across tasks

Chapter 8

Numerical Stability

8.1 Importance of Numerical Stability in Survival Models

Numerical stability is a critical yet often overlooked aspect of implementing survival models, particularly those based on parametric distributions and deep learning approaches (Goldberg, 1991; Hanin and Rolnick, 2022). This chapter explores the numerical challenges that arise when implementing models like DSM (Nagpal, Li, and Dubrawski, 2021) and MENSA (Y. Zhong et al., 2021), and provides practical solutions to ensure robustness and reliability.

Chapter Overview

This chapter covers:

- Common numerical challenges in parametric survival models
- Critical calculations prone to instability
- Practical techniques for ensuring numerical stability
- Implementation strategies for robust survival models
- Testing approaches to verify numerical reliability
- The broader importance of stability for model deployment

Parametric survival models are particularly sensitive to numerical issues due to the complex mathematical forms of their distribution functions and the wide range of time scales they must handle. Small errors in calculation can lead to training failure, model divergence, or unreliable predictions. Achieving stability requires balancing precision with computational efficiency and implementing safeguards against various numerical pitfalls.

8.2 Common Numerical Challenges

Implementation of survival models faces several fundamental numerical challenges that can undermine model performance and reliability.

8.2.1 Underflow and Overflow

Floating-point arithmetic has inherent limitations in representing very small or very large numbers, leading to two common issues:

Underflow and Overflow

- **Underflow:** Occurs when a value becomes too small to be represented in the floating-point format, resulting in it being rounded to zero.
  - Example: $S(t) \approx 0$ for large t values
  - Problem: $\log(S(t))$ becomes $-\infty$
  - Occurs when values are smaller than minimum representable float ($\approx 10^{-38}$ for float32)
- **Overflow:** Occurs when a value becomes too large to be represented, resulting in it being set to infinity.
  - Example: e^x for $x > 709$ in float64
  - Problem: Large intermediate calculations explode to infinity
  - Produces infinity or NaN values that corrupt subsequent computations

These issues are particularly relevant in survival analysis where we often calculate probabilities that can be extremely small (e.g., survival probability beyond a certain point) or use exponential functions that can grow extremely large.

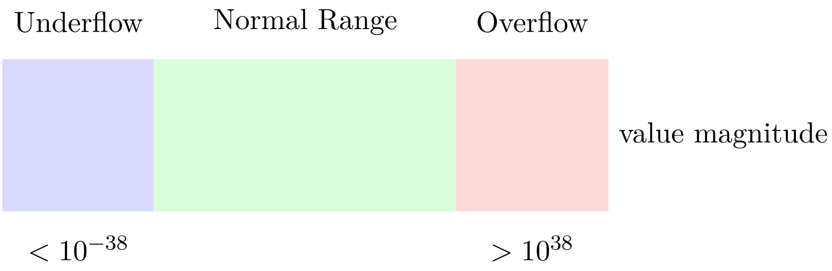


Figure 8.1: Floating-point representation ranges for 32-bit floats. Values outside the normal range suffer from underflow or overflow, leading to precision loss or invalid operations.

8.2.2 Precision Loss and Invalid Operations

Beyond underflow and overflow, other numerical issues can compromise calculation integrity:

- **Catastrophic cancellation:** Loss of significant digits when subtracting two nearly equal numbers.
  - Example: Computing $1 - S(t)$ when $S(t) \approx 1$
  - Problem: Significant digits lost in close subtractions
  - Critical for cumulative distribution: $F(t) = 1 - S(t)$

- **NaN propagation:** When an invalid operation occurs (e.g., division by zero, log of negative number), it produces a "Not a Number" (NaN) value that contaminates all subsequent calculations.
  - Example: $\log(0)$ or $\sqrt{-1}$ in computation
  - Problem: NaN infects all subsequent calculations
  - Particularly damaging in backward pass (autograd)

Catastrophic Cancellation Example

Consider computing $1 - 0.999999$ with limited precision:

- With full precision: $1.000000 - 0.999999 = 0.000001$ (correct)
- With 7 digits precision: $1.0 - 0.999999 \approx 0.0$ (incorrect)

This type of error is especially problematic when computing the cumulative distribution function $F(t) = 1 - S(t)$ for small values of t where $S(t)$ is very close to 1.

8.3 Critical Calculations in Survival Models

Certain calculations in survival models are particularly prone to numerical instability and require special attention.

8.3.1 Hazard Function Calculations

The hazard function, defined as $h(t) = \frac{f(t)}{S(t)}$, is a common source of numerical issues:

- Division by very small $S(t)$ at large t can cause overflow
- For Weibull distribution: $h(t) = \frac{\alpha}{\lambda}\left(\frac{t}{\lambda}\right)^{\alpha-1}$
- Two problematic regions:
  - Large t: $S(t) \approx 0$ causing division by near-zero
  - Small t with $\alpha < 1$: Negative exponent causes explosion as t approaches zero

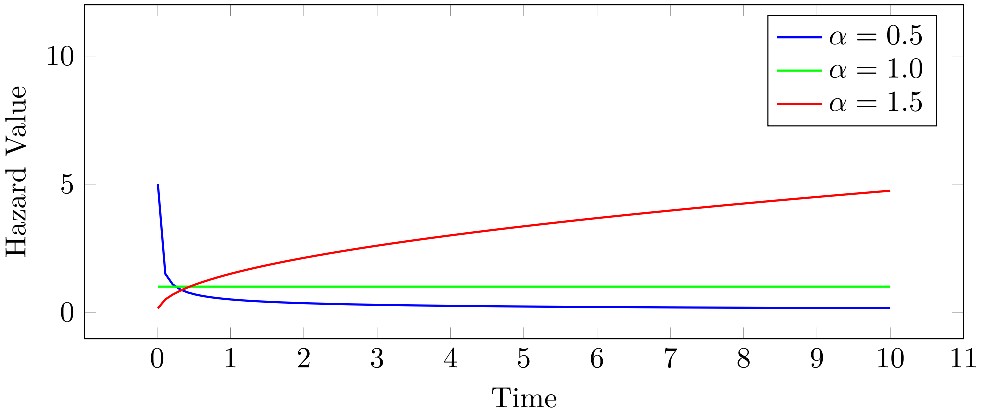


Figure 8.2: Weibull hazard functions with different shape parameters. When $\alpha < 1$ (blue), the hazard explodes as t approaches zero, creating numerical instability.

Weibull Hazard Calculation Issues

For a Weibull distribution with $\alpha = 0.5$ and $\lambda = 1$:

Time	Hazard Value
0.001	15.8
0.01	5.0
0.1	1.6
1.0	0.5
10.0	0.16

As t approaches zero, the hazard value explodes due to the negative exponent ($\alpha - 1 = -0.5$). This can cause:

- Overflow in hazard value computation
- Unstable gradients during backpropagation
- Cascade of NaN values in forward/backward passes
- Training failure without proper safeguards

8.3.2 Mixture Model Challenges

Mixture models like DSM face additional numerical challenges:

- **Mixture log-likelihood:** Computing $\log \sum_k \pi_k f_k(t)$ can be unstable
  - Underflow if all $f_k(t)$ are very small
  - Sum becomes effectively zero, leading to $\log(0) = -\infty$
  - Common for points in tails of all components
- **Component scale disparity:** Component densities can span many orders of magnitude
  - Some components contribute negligibly

  - Numeric precision lost in summation
  - Gradient flow dominated by the largest components

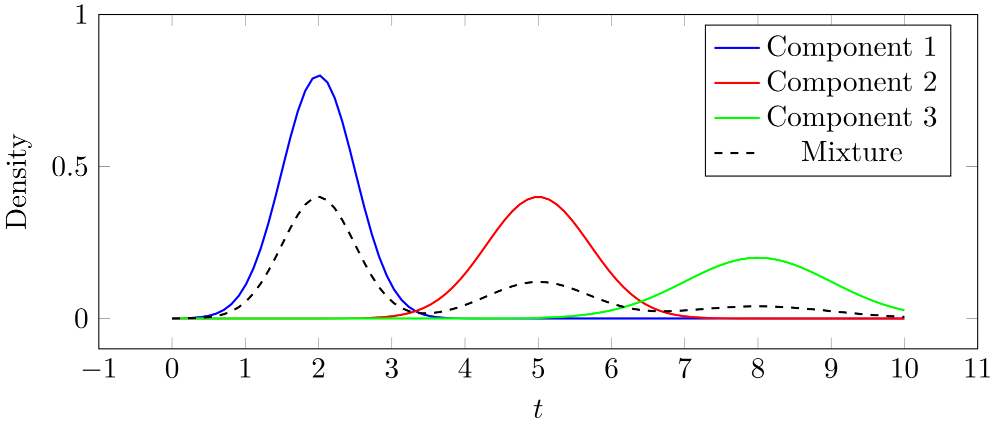


Figure 8.3: A mixture model combines multiple component distributions. Near the tails, some components' densities can be orders of magnitude smaller than others, leading to numerical challenges.

8.3.3 Gradient Computation Challenges

Automatic differentiation in deep learning frameworks produces gradients that can also suffer from numerical instability:

- **Backpropagation through exponentiation:**
  - Gradient of e^x is e^x
  - For large x, gradient explodes
  - For very negative x, gradient vanishes
  - Example: $x = 100 \Rightarrow e^x \approx 10^{43} \Rightarrow \nabla_x e^x \approx 10^{43}$
- **Weibull-specific gradient issues:**
  - Shape parameter gradient scales with time values
  - Can lead to instability with diverse time ranges
  - Extreme parameter values exacerbate gradient issues
- **NaN propagation in gradients:**
  - One invalid operation can corrupt the entire backward pass
  - Requires comprehensive guarding throughout computation
  - Affects all model parameters, not just the problematic ones

8.4 Solutions for Numerical Stability

Several proven techniques can address these numerical challenges and ensure stable model implementation.

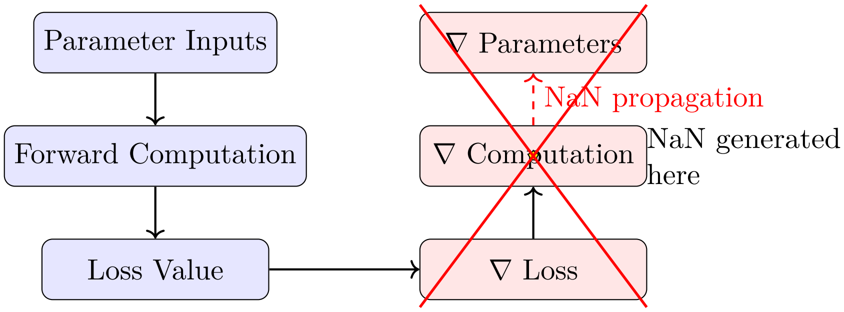


Figure 8.4: NaN propagation in the backward pass: a single numerical error in gradient computation propagates and corrupts all parameter updates.

8.4.1 Log-Domain Calculations

Working in the logarithmic domain is perhaps the most important technique for preventing underflow and overflow:

Log-Domain Transformations

Instead of computing $S(t)$ directly, compute $\log S(t)$ and only exponentiate when necessary:

$$\log S(t) = -\left(\frac{t}{\lambda}\right)^{\alpha} \tag{8.1}$$

$$S(t) = \exp\left(\log S(t)\right) \tag{8.2}$$

Key transformations:

$$\log(a \cdot b \cdot c) = \log(a) + \log(b) + \log(c) \tag{8.3}$$

$$\log\left(\frac{a}{b}\right) = \log(a) - \log(b) \tag{8.4}$$

$$\log(a^b) = b \cdot \log(a) \tag{8.5}$$

Log-Domain Calculation Examples

Common survival calculations in log-domain:

Standard Form	**Log-Domain Form**
$S(t) = \exp(-(\frac{t}{\lambda})^{\alpha})$	$\log S(t) = -(\frac{t}{\lambda})^{\alpha}$
$f(t) = \frac{\alpha}{\lambda}(\frac{t}{\lambda})^{\alpha-1}S(t)$	$\log f(t) = \log(\frac{\alpha}{\lambda}) + (\alpha - 1)\log(\frac{t}{\lambda}) + \log S(t)$

Value ranges comparison:

x	e^x	$\log y$
-10	0.00005	-10
-1	0.368	-1
0	1	0
1	2.718	1
10	22026	10
100	$\approx 10^{43}$	100
710	OVERFLOW	710

Working in log-domain keeps values in a numerically stable range, even for extremely large or small inputs.

8.4.2 Log-Sum-Exp Trick

The log-sum-exp trick is crucial for stable computation of mixture models:

Log-Sum-Exp Trick

For stable computation of $\log \sum_i e^{x_i}$:

$$\log \sum_i e^{x_i} = \log \left[e^a \sum_i e^{x_i - a} \right] \tag{8.6}$$

$$= a + \log \sum_i e^{x_i - a} \tag{8.7}$$

where $a = \max_i x_i$
This technique subtracts the maximum value before exponentiation, preventing overflow while maintaining mathematical equivalence.

Implementation process:

1. Find maximum value $a = \max_k z_k$
2. Subtract a from each z_k before exponentiation
3. Sum the resulting values (all ≤ 1)
4. Take logarithm and add a back

Key benefits:

- Prevents underflow in the sum

- Works even when components span many orders of magnitude
- Preserves numerical precision
- Mathematically equivalent to direct computation
- Critical for mixture model stability

8.4.3 Gradient Detachment Strategy

Safe gradient computation is essential for stable training. Standard techniques like gradient clipping may not be sufficient for the extreme cases encountered in survival models.

Gradient Challenges

Problems with standard operations:

- NaN gradients stop backward propagation entirely
- Boolean masking requires shape compatibility
- Clipping alone doesn't fix fundamental gradient issues
- Safe paths are needed for extreme input regions

The gradient detachment approach provides a robust solution:

Gradient Detachment With Safe Masking

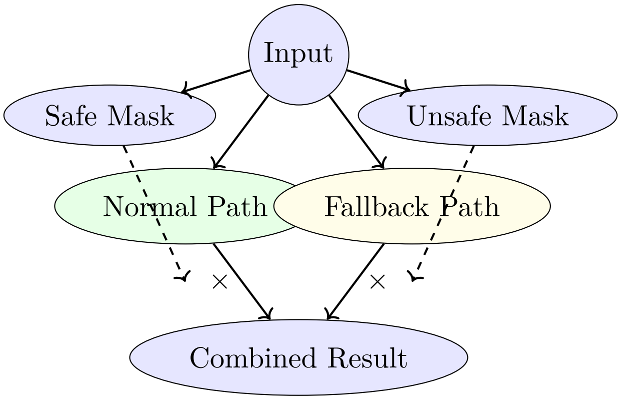
$$\begin{aligned}
\text{unsafe\_mask} &= (x > \text{threshold}).\mathit{float}() && (8.8)\\
\text{safe\_mask} &= 1.0 - \text{unsafe\_mask} && (8.9)\\
\text{normal\_result} &= \text{original\_function}(x) && (8.10)\\
\text{fallback} &= \text{safe\_value} && (8.11)\\
\text{result} &= \text{safe\_mask} \cdot \text{normal\_result} + \text{unsafe\_mask} \cdot \text{fallback} && (8.12)
\end{aligned}$$

This approach:

- Prevents NaN propagation while allowing normal calculation in safe regions
- Enables training to continue despite some extreme values
- Creates smooth transitions between computation regimes
- Maintains gradient flow through valid components

8.4.4 Case Study: Weibull Hazard Stabilization

Weibull hazard calculation is particularly challenging when the shape parameter $\alpha < 1$, causing the hazard to approach infinity as $t \to 0$:



result = safe_mask × normal_result + unsafe_mask × fallback

Figure 8.6: Gradient detachment with safe masking: Input values are classified into safe and unsafe regions, with a separate computation path for each, then combined using masks to ensure stability.

$$h(t) = \frac{\alpha}{\lambda}\left(\frac{t}{\lambda}\right)^{\alpha-1} \tag{8.13}$$

Special handling required:

- Log-domain computation always
- Special clamping near $t = 0$
- Shape parameter buffer ($\alpha > 0.05$)

Implementation approaches:

- Clamping shape-1 to prevent extreme negative exponents:

$$\text{shape\_m1} = \text{clamp(shape - 1.0, min=-0.95)} \tag{8.14}$$

- Using log domain to compute power terms:

$$\text{log\_power} = \text{shape\_m1} \cdot \text{log\_ratio} \tag{8.15}$$

$$\text{power} = \exp(\text{log\_power}) \tag{8.16}$$

These approaches ensure safe computation even for small t values and shape parameters less than 1, maintaining the same values in safe regions while preventing overflow in extreme regions.

8.5 Loss Function Stability Techniques

Beyond individual calculations, ensuring the overall loss function is numerically stable is crucial for reliable training.

8.5.1 NaN Detection and Reporting

Early detection of numerical issues helps identify and address problems:

- Implement NaN detection in the loss computation:

  ```
  if debug_nans and torch.isnan(log_likelihood).any():
      print(f"NaN detected: {torch.isnan(log_likelihood).sum().item()}")
  ```

- Provides early warning during training
- Helps identify specific components causing issues
- Can be conditionally enabled during development

8.5.2 Safe Loss Aggregation

When computing losses over multiple samples or components, handle invalid values gracefully:

- Skip invalid components when aggregating losses:

  ```
  # Only add valid losses
  if torch.isfinite(event_loss):
      total_loss = total_loss + event_loss
      total_valid_events += 1
  ```

- Prevents a single invalid value from corrupting the entire batch
- Maintains gradient flow through valid samples
- Allows training to continue despite some problematic data points

8.5.3 Fallback Mechanism

As a last resort, implement a fallback for the overall loss:

- Provide a safety net for extreme cases:

  ```
  if not torch.isfinite(final_loss):
      logger.warning(f"Non-finite loss: {final_loss.item()}")
      final_loss = torch.tensor(1.0, device=device, requires_grad=True)
  ```

- Allows training to continue despite some invalid values
- Prevents entire training run from failing due to occasional issues
- Logs warnings to alert developers about the problem
- Should be a rare occurrence in a well-designed model

8.6 Testing for Numerical Stability

Comprehensive testing is essential to verify numerical stability across a wide range of inputs and conditions.

8.6.1 Extreme Value Testing

Test models with input values at the extremes of expected ranges:

- **Very small/large time values:**
  - Test with $t \approx 0$ (e.g., 10^{-10})
  - Test with very large t (e.g., 10^{10})
  - Check edge cases where $S(t) \approx 0$ or $S(t) \approx 1$
- **Boundary shape parameters:**
  - Test with α very close to 0 (e.g., 0.01)
  - Test with α near 1.0 (important transition point)
  - Test with very large α values (e.g., 20+)

8.6.2 Gradient Testing

Explicitly test the backward pass to ensure stable gradient computation:

- **Test backward pass explicitly:**
  - Verify gradient computation with torch.autograd.grad()
  - Ensure gradients are finite for all parameters
  - Compare analytical vs. numerical gradients
- **Gradient monitoring during training:**
  - Track gradient norms over training epochs
  - Detect sudden spikes in gradient magnitude
  - Implement gradient clipping as safety mechanism

8.6.3 Comprehensive Test Coverage

Ensure test coverage across all critical aspects of the model:

- Input values across the full range of time scales
- All parameter ranges, especially near boundaries
- Gradient flow through all model components
- Mixture models with various component configurations
- Censored and uncensored data points
- Edge cases specific to your application domain

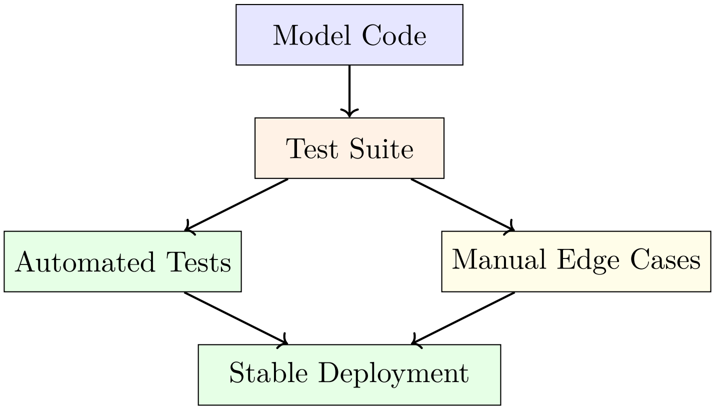


Figure 8.7: A comprehensive testing pipeline for ensuring numerical stability, combining automated tests with manual verification of edge cases.

8.7 The Importance of Numerical Stability for Deployment

Numerical stability is not merely a technical detailit forms the foundation for reliable model deployment in real-world applications.

8.7.1 Critical Applications

In high-stakes domains like healthcare, numerical instability can have serious consequences:

- Invalid predictions could lead to incorrect medical decisions
- Inconsistent model behavior undermines trust in the system
- Training failures delay model development and deployment
- Production issues may be difficult to diagnose and fix

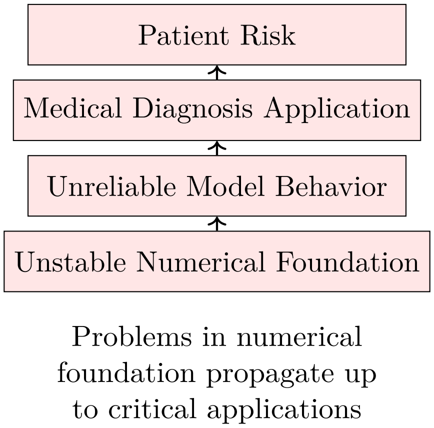

Patient Risk
Medical Diagnosis Application
Unreliable Model Behavior
Unstable Numerical Foundation
Problems in numerical foundation propagate up to critical applications

Figure 8.8: The impact chain of numerical instability: Fundamental numerical issues propagate upward through the model to affect critical applications and ultimately patient outcomes.

8.7.2 Technical Benefits

Beyond preventing catastrophic failures, numerical stability offers several technical advantages:

- Ensures consistent results across hardware platforms
- Allows models to generalize beyond the training data range
- Improves convergence properties during training
- Enables reliable uncertainty quantification
- Facilitates model interpretability and explanation

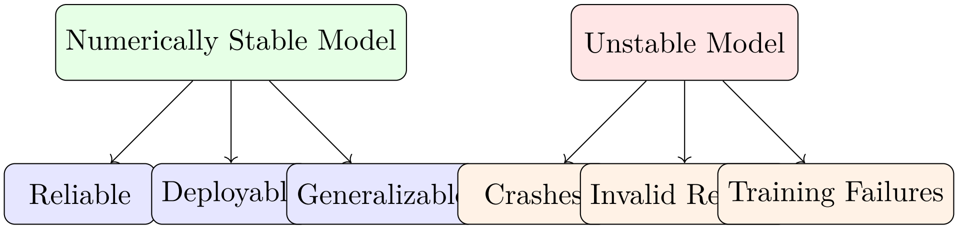


Figure 8.9: Comparison of outcomes between numerically stable and unstable models. Stable models are reliable, deployable, and generalizable, while unstable models suffer from crashes, invalid results, and training failures.

8.8 Summary

Numerical stability is a critical aspect of implementing parametric survival models, particularly deep learning approaches like DSM and MENSA. The key takeaways from this chapter include:

- Underflow, overflow, and precision loss are common numerical challenges in survival models
- Hazard function calculations, mixture models, and gradient computation require special attention
- Working in the log domain, using the log-sum-exp trick, and implementing gradient detachment are effective solutions
- Loss function stability techniques, including NaN detection and safe aggregation, ensure reliable training
- Comprehensive testing across extreme values and boundary conditions is essential
- Numerical stability forms the foundation for reliable model deployment in critical applications

By implementing these techniques and remaining vigilant about numerical stability, we can build survival models that are not only theoretically sound but also practically reliable in real-world applications.

Looking Ahead

In the next chapter, we will explore how expert knowledge can be incorporated into survival models to enhance their performance and interpretability. This includes approaches for integrating domain expertise into model architecture, parameter constraints, and regularization strategies.

Chapter 9

Expert Knowledge Integration

9.1 The Importance of Expert Knowledge in Survival Analysis

While data-driven machine learning approaches to survival analysis offer tremendous potential (Katzman et al., 2018; C. Lee et al., 2018), purely algorithmic methods may miss critical domain insights that are well-established in clinical research and practice (Radfar et al., 2022; Rudin, 2019). This chapter explores how incorporating expert knowledge into survival models creates more robust, interpretable, and trustworthy predictions, particularly in high-stakes medical applications (Kuo et al., 2020; Ghassemi, Oakden-Rayner, and Beam, 2022; Karimi, Schölkopf, and Valera, 2021).

Chapter Overview

This chapter covers:

- Why expert knowledge is essential in survival analysis
- Types of domain expertise relevant to survival modeling
- Techniques for formalizing qualitative knowledge
- Parameter and feature-level constraint methods
- Neural architectures guided by expert knowledge
- Regularization and distillation approaches
- Case studies of expert-enhanced survival models
- Evaluation methods and future research directions

9.2 Why Expert Knowledge Matters in Survival Analysis

The integration of expert knowledge into survival analysis addresses several fundamental limitations of purely data-driven approaches.

9.2.1 Limitations of Data-Only Approaches

Despite advances in machine learning, survival models trained solely on available data face significant challenges:

- **Limited training data:** Many survival datasets are small, especially for rare diseases or specific patient subgroups
- **Censoring creates fundamental uncertainties:** Right-censored observations provide incomplete information that may be supplemented by domain knowledge
- **Distribution shifts between populations:** Models trained on one population may not generalize well to others without incorporating expert understanding of underlying biology
- **Causal mechanisms vs. statistical correlations:** Statistical patterns may not reflect causal mechanisms understood by domain experts
- **Scientific understanding beyond observed patterns:** Decades of clinical research have established relationships that may not be apparent in limited datasets

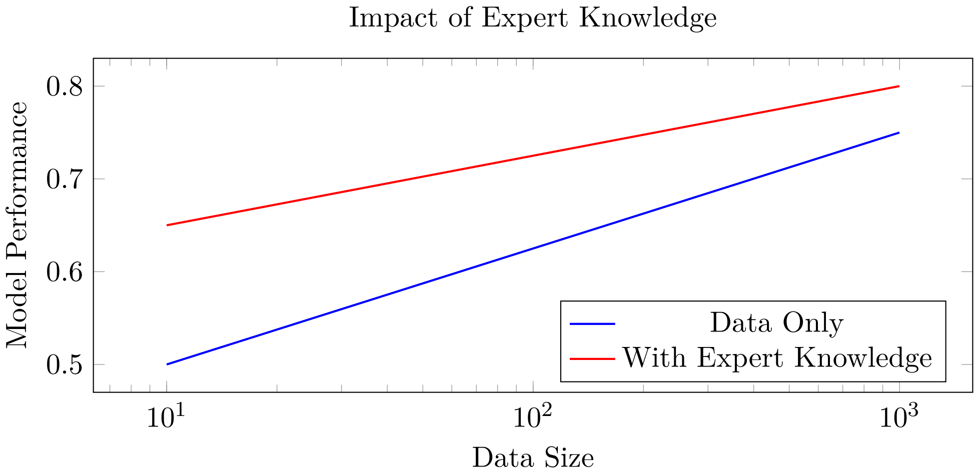


Figure 9.1: The performance gap between data-only and expert-guided models. Expert knowledge provides the greatest benefit with smaller datasets but continues to improve performance even as data size increases.

9.2.2 The Gap Between Data and Understanding

Medical knowledge represents a vast accumulation of understanding that cannot be fully captured in any single dataset:

- **Observational vs. mechanistic understanding:** Statistical correlations may identify patterns, but experts understand the underlying biological mechanisms
- **Rare but important scenarios:** Experts have knowledge about rare conditions or edge cases that may be underrepresented in available data
- **Longitudinal progression:** Clinical expertise includes understanding of disease trajectories that may extend beyond available follow-up periods
- **Treatment effect modifiers:** Domain knowledge about which factors influence treatment efficacy may not be apparent from limited trial data

- **Physiological constraints:** Biological systems operate within constraints that should be reflected in predictive models

Expert Knowledge in Cancer Prognosis

In cancer modeling, expert oncologists understand that:

- Hazard functions typically increase over time as tumors grow (suggesting Weibull distributions with shape parameter > 1)
- Risk factors like tumor size, grade, and certain biomarkers have monotonically increasing effects on risk
- Different cancer subtypes have distinct survival patterns that may require different parameter ranges
- Treatment effects often follow specific temporal patterns (e.g., initial benefit followed by potential resistance)
- Competing risks (cancer death vs. other causes) have specific dependency structures

This knowledge can be formalized as constraints, priors, or architectural guidance for survival models.

9.3 Types of Expert Knowledge in Survival Analysis

Domain expertise relevant to survival analysis comes in several forms, each informing different aspects of model development.

9.3.1 Forms of Domain Expertise

Distributional Knowledge

Experts often have insights about the overall patterns of event timing:

- **Shape of hazard functions:** Understanding whether risk increases, decreases, or follows more complex patterns over time
- **Expected survival patterns:** Knowledge of typical survival curves for specific conditions
- **Parametric families for events:** Insights about which distributions best model certain diseases (e.g., Weibull for cancer progression)
- **Common censoring mechanisms:** Understanding the patterns and reasons for censoring in specific domains

Feature Relationships

Clinical expertise includes knowledge about risk factors and their effects:

- **Known risk factors:** Established predictors for specific outcomes
- **Effect directions:** Whether factors increase or decrease risk

- **Interaction effects:** How factors modify each other's impact on risk
- **Non-linear relationships:** Knowledge of threshold effects, U-shaped relationships, or plateaus
- **Effect magnitudes:** Relative importance of different predictors

Temporal Patterns

Time-dependent aspects of risk are particularly relevant to survival analysis:

- **Expected changes over time:** How risk evolves with disease duration
- **Time-varying effects:** Factors whose influence changes over the follow-up period
- **Critical time periods:** Windows of particularly high or low risk
- **Treatment timing effects:** How intervention timing influences outcomes

Event Dependencies

In multi-event settings, experts understand relationships between different outcomes:

- **Relationships between competing risks:** How different event types relate to each other
- **Causal connections:** When one event directly influences the risk of another
- **Conditional dependencies:** How risk relationships change given certain events or treatments
- **Sequential patterns:** Typical progression paths through multiple events

Population Heterogeneity

Experts recognize distinct patterns across patient subgroups:

- **Subgroup differences:** How risk profiles differ across patient segments
- **Patient stratification criteria:** Factors that define meaningful subgroups
- **Treatment effect modifiers:** Characteristics that influence treatment response
- **Age and demographic effects:** How risk varies across demographic factors

Biological Mechanisms

Understanding of underlying biological processes informs risk patterns:

- **Disease progression patterns:** Stages and mechanisms of disease evolution
- **Physiological constraints:** Biological limits that constrain possible outcomes
- **Treatment response dynamics:** Mechanisms of therapeutic effect and resistance
- **Compensatory systems:** How biological systems adapt to changes

9.3.2 Sources of Expert Knowledge

Domain expertise can be derived from various sources:

- **Clinical practice guidelines:** Consensus recommendations from professional societies
- **Published medical literature:** Peer-reviewed research and meta-analyses
- **Established risk scores and nomograms:** Validated clinical prediction tools
- **Direct clinician consultations:** Interviews with subject matter experts
- **Consensus panel recommendations:** Formalized expert opinions
- **Mechanistic models from biology/physiology:** Mathematical models of underlying processes
- **Previous clinical trials and meta-analyses:** Aggregated evidence from controlled studies
- **Disease registries and historical data:** Long-term observational records

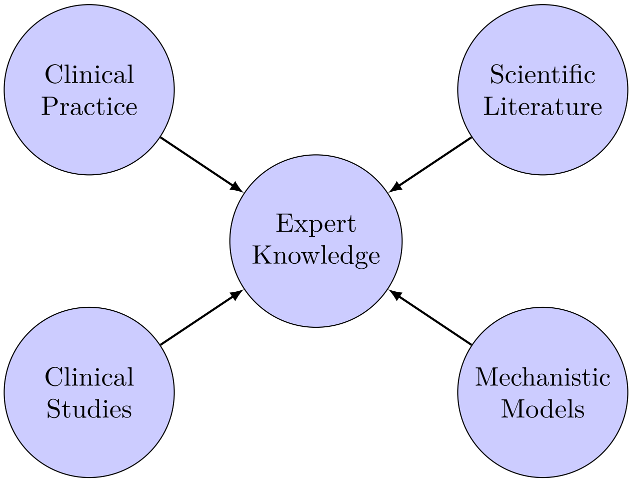


Figure 9.2: Sources of expert knowledge for integration into survival models. Multiple sources contribute to a comprehensive understanding that can guide model development.

9.4 Challenges in Knowledge Integration

Incorporating expert knowledge into statistical models presents several challenges that must be addressed for successful implementation.

9.4.1 Formalization Challenges

Converting qualitative expertise into quantitative model constraints is a fundamental challenge:

- **Converting qualitative expertise to quantitative constraints:** Experts often express knowledge in qualitative terms that must be translated into mathematical formulations

- **Representing uncertainty in expert knowledge:** Experts have varying degrees of confidence in different aspects of their knowledge
- **Handling conflicting expert opinions:** Different experts may have contradictory views on certain aspects of disease progression
- **Translating clinical language to mathematical formulations:** Bridging the gap between medical terminology and statistical concepts
- **Capturing context-dependent knowledge:** Expert insights that apply only in specific situations

9.4.2 Integration Challenges

Balancing expert knowledge with data-driven learning presents additional challenges:

- **Balancing data evidence vs. prior knowledge:** Determining how strongly to weight expert priors relative to observed data
- **Incorporating knowledge without overly restricting the model:** Allowing flexibility to learn patterns not anticipated by experts
- **Maintaining computational tractability:** Some constraints may significantly increase model complexity
- **Adapting knowledge strength to data size:** Reducing the influence of priors as data volume increases
- **Updating knowledge based on new evidence:** Incorporating feedback loops to refine expert knowledge

9.4.3 Validation Challenges

Evaluating the impact of expert knowledge integration introduces methodological challenges:

- **Measuring the value added by expert knowledge:** Quantifying improvement over purely data-driven approaches
- **Testing if knowledge incorporation improves generalization:** Verifying better performance on new populations
- **Detecting when expertise conflicts with data evidence:** Identifying situations where expert knowledge may be misleading
- **Determining when to override expert priors:** Criteria for letting data outweigh prior knowledge
- **Developing fairness metrics for knowledge-guided models:** Ensuring that expert knowledge doesn't introduce or amplify biases

9.5 Knowledge Formalization Techniques

Before expert knowledge can be incorporated into models, it must be formalized through structured processes.

9.5.1 Elicitation Methods

Several approaches can systematically capture expert knowledge:

- **Structured interviews with domain experts:** One-on-one discussions following established protocols
- **Survey techniques with confidence ratings:** Questionnaires that capture both judgments and confidence levels
- **Delphi method for consensus building:** Iterative anonymous feedback process to reach expert consensus
- **Literature review and meta-analysis:** Systematic synthesis of published evidence
- **Analysis of existing clinical models:** Reverse-engineering established prediction tools
- **Feature importance rankings:** Expert prioritization of predictive factors
- **Qualitative to quantitative mapping frameworks:** Structured conversion of verbal descriptions to numeric constraints

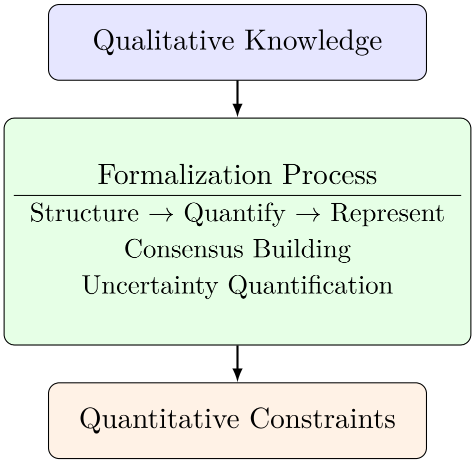


Figure 9.3: The knowledge formalization process converts qualitative expert insights into quantitative model constraints through structured elicitation and representation techniques.

9.5.2 Bayesian Frameworks

Bayesian statistics provides a natural framework for incorporating expert knowledge:

- **Prior distribution elicitation:** Capturing expert beliefs as probability distributions over parameters
- **Hierarchical priors:** Structured prior frameworks that allow for uncertainty in expert knowledge
- **Equivalent sample size determination:** Quantifying the strength of prior knowledge relative to observed data
- **Prior predictive checks:** Verifying that prior distributions lead to reasonable predictions

9.6 Parameter Constraints in Survival Models

One of the most direct approaches to incorporating expert knowledge is through constraints on model parameters, particularly those governing the shape of hazard functions.

9.6.1 Disease-Specific Shape Parameters

For parametric survival models like the Weibull distribution, experts can provide guidance on appropriate parameter ranges:

Weibull Distribution Parameters

The Weibull distribution has two key parameters:

- Shape parameter α: Controls whether hazard increases ($\alpha > 1$), decreases ($\alpha < 1$), or remains constant ($\alpha = 1$) over time
- Scale parameter λ: Related to the median survival time

Expert knowledge can constrain these parameters based on disease characteristics:

$$h(t) = \frac{\alpha}{\lambda}\left(\frac{t}{\lambda}\right)^{\alpha-1} \tag{9.1}$$

Disease-Specific Shape Constraints

Experts can provide guidance on shape parameters for different conditions:

- **Cancer progression:** $\alpha > 1$ (increasing hazard)
  - Tumor growth accelerates damage over time
  - Metastatic spread increases with disease duration
  - Example: $\alpha \in [1.2, 2.5]$ for breast cancer
- **Infectious disease:** $\alpha < 1$ (decreasing hazard)
  - Highest risk immediately after exposure/infection
  - Immune response decreases hazard over time
  - Example: $\alpha \in [0.5, 0.9]$ for post-surgical infection
- **Chronic conditions:** $\alpha \approx 1$ (constant hazard)
  - Ongoing risk remains relatively stable
  - Random event occurrence pattern
  - Example: $\alpha \in [0.9, 1.1]$ for chronic heart failure

9.6.2 Implementation Approaches for Parameter Constraints

Several techniques can enforce expert-defined parameter constraints:

- **Parameter bounding:** Ensure parameters remain within valid ranges using transformations

$$\alpha = \alpha_{min} + \text{softplus}(w) \tag{9.2}$$

- **Informative priors:** Use probability distributions that encode expert knowledge

$$\alpha \sim \text{Gamma}(a, b) \tag{9.3}$$

- **Penalty terms:** Add regularization that penalizes deviation from expert expectations

$$\mathcal{L}_{penalty} = \lambda(\alpha - \alpha_{prior})^2 \tag{9.4}$$

- **Constrained activation:** Use bounded activation functions to enforce parameter limits

$$\alpha = \alpha_{min} + (\alpha_{max} - \alpha_{min})\sigma(w) \tag{9.5}$$

  where σ is the sigmoid function

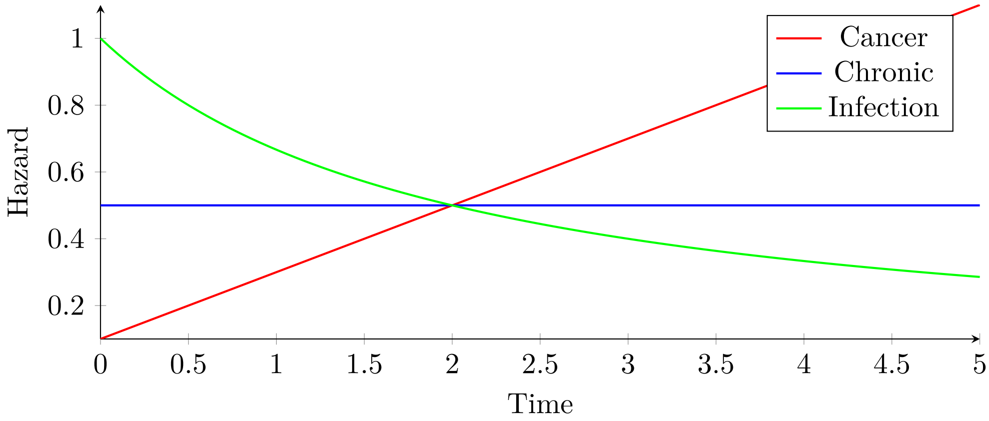


Figure 9.4: Expert-guided hazard shapes for different disease types. Cancer typically shows increasing hazard (red), chronic conditions often have constant hazard (blue), and infectious complications frequently show decreasing hazard over time (green).

9.7 Feature-Level Expert Constraints

Beyond distributional parameters, expert knowledge can inform how individual features relate to survival outcomes.

9.7.1 Knowledge About Risk Factors

Experts typically have well-established understanding of how specific variables influence risk:

- **Sign constraints:** Force coefficients to have expert-expected signs
  - Example: Age coefficient must be positive for most diseases
  - Implementation: $\beta_{age} = \text{softplus}(w)$

- **Relative importance:** Constrain relative magnitudes of feature effects
  - Example: Smoking impact > BMI impact for lung cancer
  - Implementation: $|\beta_{smoking}| > |\beta_{BMI}|$
- **Feature grouping:** Related features should have similar effects
  - Example: Various cholesterol measures
  - Implementation: Group lasso or similarity penalties

9.7.2 Monotonicity Constraints

Many risk factors are known to have consistently increasing or decreasing effects:

- **Monotonic feature effects:** Enforce consistently directional relationships
  - Risk monotonically increases with age
  - Blood pressure has threshold effects
  - Implementation: constrained neural networks
- **Methods for monotonicity:**
  - Positive-weight-only networks
  - Monotonic spline transformations
  - Architectural constraints in layers
  - Gradient penalties during training

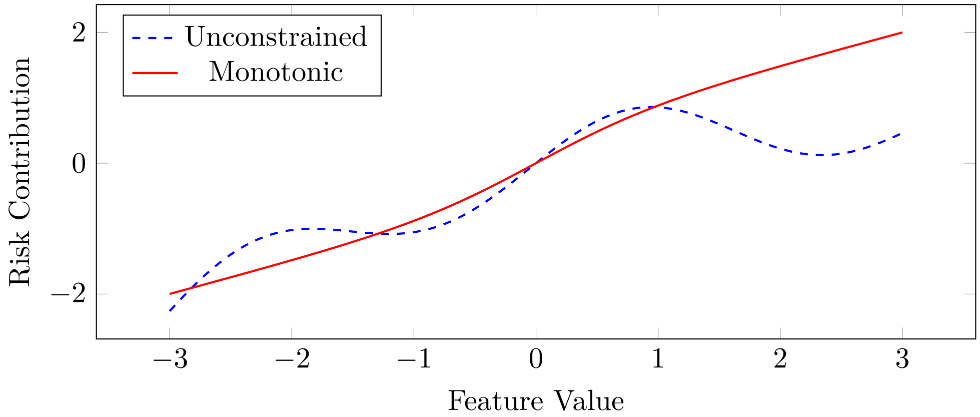


Figure 9.5: Comparison of unconstrained (blue) and monotonically constrained (red) feature effects. The unconstrained function can learn arbitrary patterns, while the monotonic constraint ensures the relationship consistently increases with the feature value, aligning with expert knowledge.

9.8 Expert-Guided Neural Architectures

The architecture of neural survival models can be designed to incorporate expert knowledge directly into the model structure.

9.8.1 Structure Constraints for Networks

Network architecture can reflect expert understanding of feature importance and relationships:

- **Feature importance constraints:**
  - Force known risk factors to have high weights
  - Regularize less well-understood features more heavily
  - $\mathcal{L}_{reg} = \sum_i w_i |\theta_i|$ where w_i is feature-specific regularization
- **Attention mechanisms:**
  - Guide attention to focus on important features
  - Prior attention masks from domain knowledge
  - Semi-supervised attention learning
- **Dependency structures:**
  - Expert-defined event dependency matrix D_{ij} between risks
  - Encode in the MENSA dependency layer architecture
  - $\pi_{jk}(x) = g(W_j \cdot \phi(x) + \sum_i D_{ij} \cdot W_i \cdot \phi(x))$

9.8.2 Dedicated Architecture for Expert Constraints

Custom network architectures can be designed specifically to incorporate expert knowledge at multiple levels:

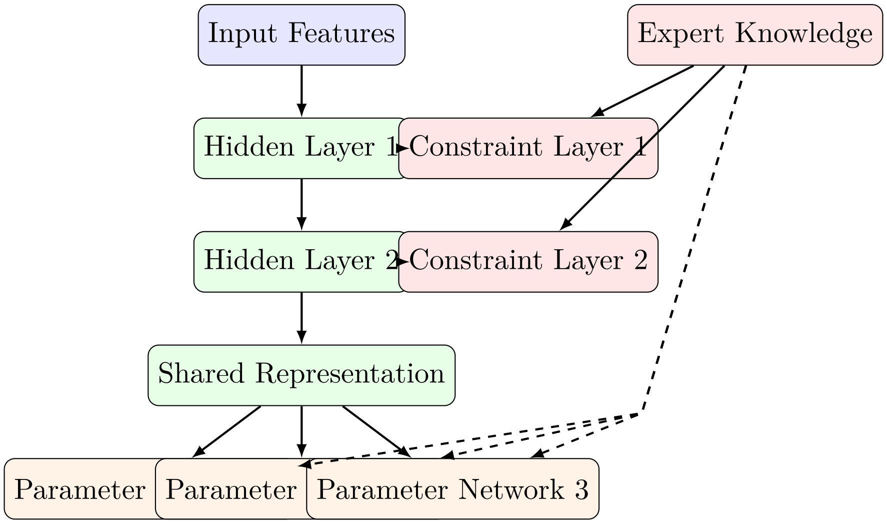


Figure 9.6: Architecture for expert knowledge integration into a neural survival model. Expert knowledge influences the network at multiple levels: constraining intermediate representations, guiding feature learning, and directly shaping distribution parameters.

This architecture incorporates expert knowledge at multiple levels:

- Constraint layers directly modify hidden representations based on expert knowledge
- Direct connections to parameter networks enforce constraints on distribution parameters
- The entire architecture reflects the causal structure understood by domain experts

9.9 Expert Knowledge as Regularization

Rather than strictly constraining the model, expert knowledge can be incorporated as regularization that guides learning while still allowing data to drive the primary optimization.

9.9.1 Training with Expert Priors

Expert knowledge can be formalized as prior distributions and incorporated into the objective function:

Expert-Guided Regularization

Prior distribution on parameters given expert knowledge:

$$p(\theta|\text{expert}) \propto \exp(-\lambda R(\theta, \text{expert})) \tag{9.6}$$

Regularized log-likelihood:

$$\mathcal{L}_{reg} = \mathcal{L}_{data} + \lambda R(\theta, \text{expert}) \tag{9.7}$$

where λ controls the strength of the expert prior relative to the data likelihood.

9.9.2 Common Regularization Forms

Several forms of regularization can encode different types of expert knowledge:

- **Parameter-targeted:** Regularize model parameters directly

  $$R(\theta, \text{expert}) = \|\theta - \theta_{expert}\|^2 \tag{9.8}$$

- **Output-targeted:** Regularize model predictions toward expert expectations

  $$R(\theta, \text{expert}) = \|f_\theta(x) - f_{expert}(x)\|^2 \tag{9.9}$$

- **Structure-targeted:** Regularize intermediate representations to align with expert understanding

  $$R(\theta, \text{expert}) = \|g_\theta(x) - g_{expert}(x)\|^2 \tag{9.10}$$

  where g_θ represents intermediate representations

- **Adaptive weighting:** Adjust regularization strength based on data size

  $$\lambda = \lambda_0 \cdot \frac{n_0}{n_0 + n} \tag{9.11}$$

  where n_0 represents the "equivalent sample size" of expert knowledge

9.10 Knowledge Distillation from Expert Models

Another approach to incorporating expert knowledge is through knowledge distillation from simpler, more interpretable models developed based on expert understanding.

9.10.1 Model Distillation Process

Knowledge distillation transfers insights from expert-developed models to more flexible neural models:

- **Expert models as teachers:**
  - Traditional survival models built by experts
  - Disease-specific simplified models
  - Clinical risk scores and nomograms
  - Established causal models
- **Distillation process:**

$$\mathcal{L}_{distill} = \alpha\mathcal{L}_{data} + (1-\alpha)\mathcal{L}_{expert} \tag{9.12}$$

- **Distillation techniques:**
  - Prediction matching: $\|f_\theta(x) - f_{expert}(x)\|^2$
  - Feature attention matching
  - Gradient similarity enforcement
  - Representation alignment

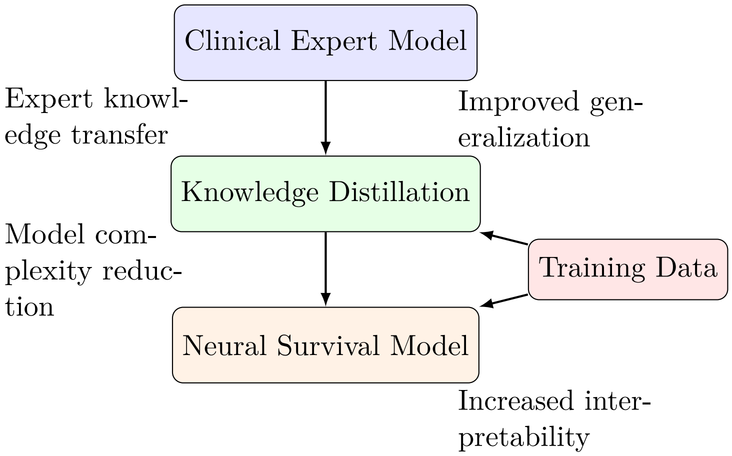


Figure 9.7: Knowledge distillation process. A simple, expert-developed clinical model serves as a teacher for a more complex neural survival model, transferring domain knowledge while maintaining the flexibility of deep learning approaches.

9.11 Expert-Guided Ensemble Methods

Ensemble methods offer another approach to combine the benefits of expert knowledge and data-driven learning.

9.11.1 Combining Models with Expert Weights

Expert knowledge can guide the combination of multiple model types:

- **Expert-weighted ensemble:**

  $$f_{ensemble}(x) = \sum_{m=1}^{M} w_m f_m(x) \tag{9.13}$$

  where w_m are expert-defined model weights

- **Model class weighting:**
  - Combine parametric, ML, and mechanistic models
  - Weights based on expert confidence in model types
  - Example: 0.5 Œ Cox + 0.3 Œ Neural + 0.2 Œ Mechanistic

- **Context-dependent weighting:**
  - Expert rules for when to trust which model
  - Different weights for different patient subgroups
  - Gating network trained with expert guidance

9.12 Case Study: Expert-Guided MENSA for Cardiovascular Disease

To illustrate the practical application of expert knowledge integration, we present a case study of an expert-guided Multi-Event Neural Survival Analysis (MENSA) model for cardiovascular disease prediction.

9.12.1 Problem: Complex Dependencies in Cardiovascular Disease

Cardiovascular disease (CVD) involves multiple interrelated events with complex dependencies:

- Multiple competing events: heart attack, stroke, heart failure
- These events have known physiological dependencies
- Traditional models treat them as independent
- Limited data for some combinations of events
- Patient subgroups have distinct risk profiles

9.12.2 Expert Knowledge Integration

Cardiologists provided structured knowledge to enhance the MENSA model:

- **Event dependency structure:** Cardiologists defined a dependency matrix between events

$$D = \begin{pmatrix} 1.0 & 0.7 & 0.5 \\ 0.7 & 1.0 & 0.3 \\ 0.5 & 0.3 & 1.0 \end{pmatrix} \quad \begin{array}{c} \text{heart attack} \\ \text{stroke} \\ \text{heart failure} \end{array} \tag{9.14}$$

- **Feature importance constraints:** Based on established Framingham risk score
  - Age, blood pressure, cholesterol given stronger weights
  - Novel biomarkers allowed more flexibility
- **Parameter constraints:** Age-dependent hazard shapes for each event

9.12.3 Results and Evaluation

The expert-guided model showed significant improvements over standard MENSA:

Model	C-index	Calibration
Standard MENSA	0.71	0.15
Expert-guided	**0.73**	**0.08**

Key benefits included:

- Better performance with limited data
- Significantly improved calibration
- More realistic risk interdependencies
- Enhanced clinician trust and adoption
- Better generalization to new populations

9.13 Integrating Expert Knowledge in the Workflow

Expert knowledge integration should be a systematic process throughout the model development lifecycle.

Key phases in this workflow include:

- **Knowledge elicitation:** Systematic capture of domain expertise
- **Formalization:** Translation into mathematical constraints
- **Architecture design:** Incorporating expertise into model structure
- **Constrained training:** Parameter learning guided by expert priors
- **Evaluation:** Assessment against both data-driven and expert-defined metrics
- **Feedback loop:** Refinement of expert knowledge based on model performance

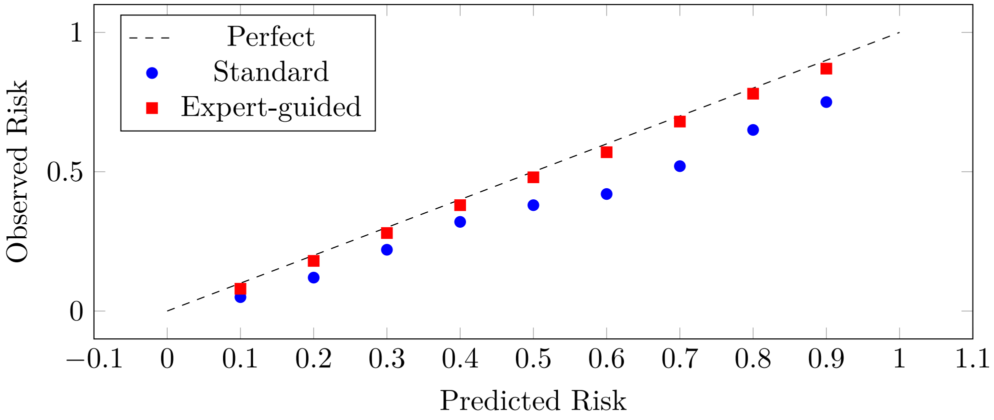


Figure 9.8: Calibration plot comparing standard MENSA (blue) and expert-guided MENSA (red). The expert-guided model shows improved alignment with the perfect calibration line (dashed), indicating better agreement between predicted and observed risks.

9.14 Measuring the Impact of Expert Knowledge

Rigorous evaluation is essential to quantify the value added by expert knowledge integration.

9.14.1 Evaluation Approaches

Several complementary approaches can assess the impact of expert knowledge:

- **Ablation studies:**
  - Compare with and without expert priors
  - Vary strength of expert constraints
  - Measure performance impact by constraint type
- **Out-of-distribution testing:**
  - Test on populations different from training
  - Evaluate performance on rare subgroups
  - Measure robustness to data shifts
- **Expert evaluation:**
  - Clinical review of model predictions
  - Assessment of prediction explanations
  - Trust and adoption metrics

9.15 Challenges and Future Research Directions

While expert knowledge integration offers significant benefits, several challenges and open research questions remain.

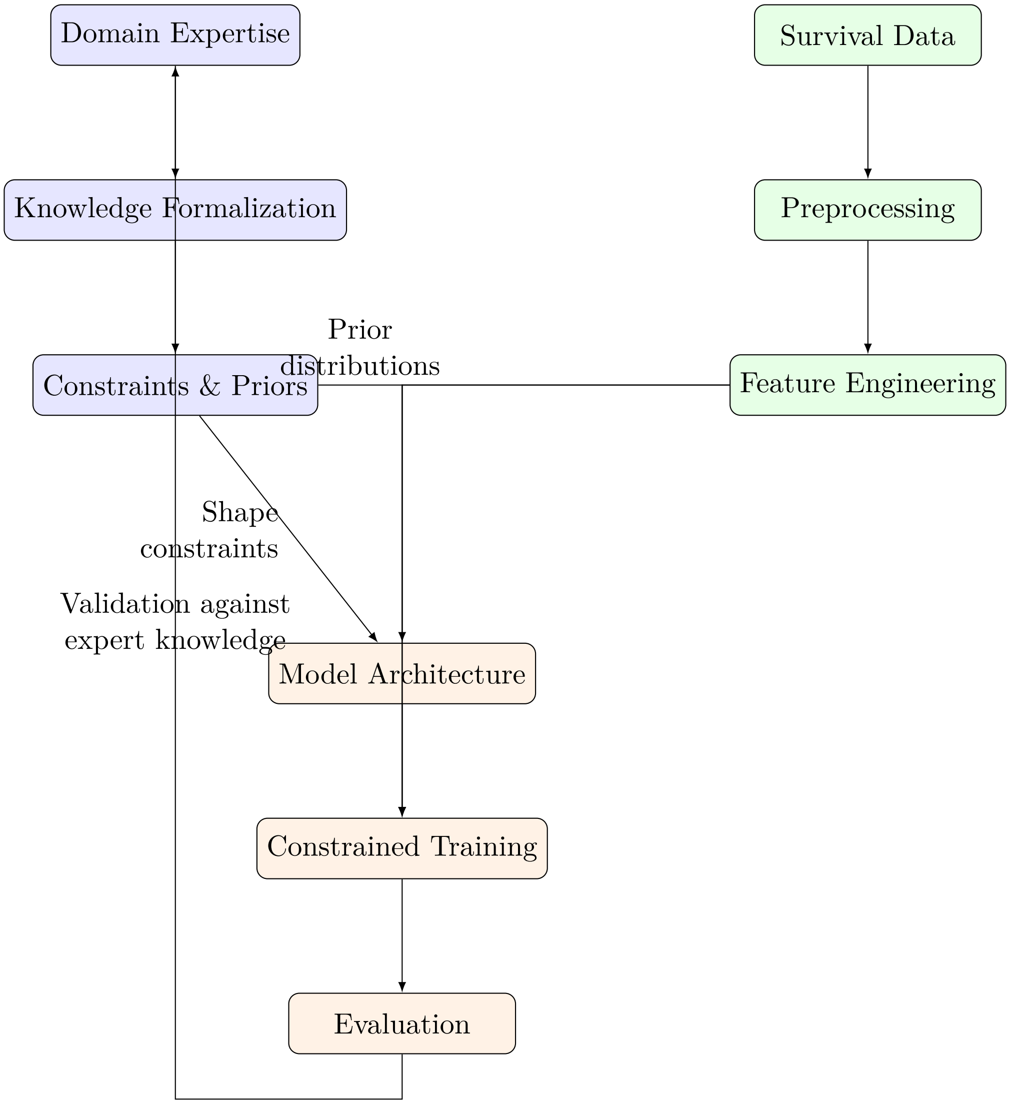


Figure 9.9: Integrated workflow for expert knowledge in survival modeling. Expert knowledge and data flow in parallel through the modeling process, with expertise informing architecture design, parameter constraints, and training regularization.

9.15.1 Current Limitations

Expert knowledge integration faces several practical challenges:

- Experts may disagree or have outdated knowledge
- Difficult to quantify confidence in expert opinions
- Overly strong priors may prevent learning from data
- Domain expertise may not transfer across populations
- Balancing expert knowledge with data-driven discovery
- Computational complexity of some constraint forms

9.15.2 Future Research Directions

Several promising research directions may address these challenges:

- Methods for eliciting quantitative constraints from experts
- Automated validation of expert knowledge against data
- Bayesian approaches to weight expert priors vs. data evidence
- Techniques to reconcile conflicting expert opinions
- Frameworks for expert knowledge transfer across domains
- Interactive systems for expert refinement of models
- Hybrid approaches combining mechanistic and ML models
- Causal structure learning with expert guidance

9.16 Summary: The Value of Expert Knowledge

Expert knowledge integration provides substantial benefits for survival analysis, particularly in high-stakes medical applications:

- Expert knowledge provides crucial guidance for survival models
- Multiple integration approaches: constraints, priors, architecture
- Benefits include better performance, generalization, interpretability
- Important for data-limited, high-stakes medical applications
- Creates bridge between clinical expertise and advanced ML
- Enables more trustworthy and scientifically consistent models
- Remains an active area of research with many open challenges

The integration of expert knowledge with modern machine learning approaches represents a powerful synergy that leverages the strengths of both human understanding and computational methods. This combination is particularly valuable in survival analysis, where domain expertise about disease progression, risk factors, and event dependencies can significantly enhance model performance and trustworthiness.

Looking Ahead

In the next chapter, we will explore how the concepts presented throughout this book can be integrated into practical applications, focusing on implementation considerations, deployment strategies, and real-world impact assessment. We will also discuss emerging trends and future directions in survival analysis that build upon the foundations, deep learning approaches, and expert knowledge integration methods we have examined.

Chapter 10

Conclusion

10.1 Conclusion

Chapter Overview

This chapter covers:

- Summary of the key concepts and approaches presented throughout the book
- Discussion of current challenges in survival analysis with censored data
- Exploration of future research directions in the field
- Practical guidelines for implementing survival models in clinical settings
- Ethical considerations in the development and deployment of survival models

Throughout this book, we have explored the development and application of deep learning approaches to time-to-event prediction with censored data. This journey has taken us from the fundamental statistical foundations of survival analysis to cutting-edge neural architectures that push the boundaries of predictive performance. As we conclude, it is worthwhile to reflect on the key themes, current challenges, and future directions in this rapidly evolving field.

10.1.1 Key Contributions and Insights

The intersection of deep learning and survival analysis has yielded significant advances in both methodology and practical applications. Several key insights have emerged from this integration:

Architectural Innovations

We have examined several innovative model architectures designed specifically for survival analysis:

- **Deep Survival Machines (DSM)** represents a paradigm shift in parametric survival modeling, offering a mixture-based approach that captures complex survival distributions while providing uncertainty quantification.
- **Multi-Event Neural Survival Analysis (MENSA)** extends the DSM framework to handle competing risks with explicit modeling of dependencies between event types, offering a more nuanced view of complex disease progression.

- **Non-parametric approaches** like DeepHit provide flexible alternatives that make minimal assumptions about the underlying survival distributions, allowing the data to speak for itself.

These architectures demonstrate that the rigid assumptions of traditional survival models can be relaxed while maintaining interpretability and adding the representational power of deep neural networks.

Loss Function Innovations

The development of specialized loss functions for censored data has been crucial to the success of deep survival models:

- **Ranking-based losses** like RankNet, SOAP, and ListMLE directly optimize for discrimination, aligning model training with the concordance index evaluation metric that is central to survival analysis.
- **Likelihood-based losses** adapted for neural networks enable proper probabilistic modeling of survival times, capturing uncertainty and producing calibrated predictions.
- **Multi-task losses** combine different objectives, allowing models to simultaneously optimize for time prediction, ranking, and calibration.

These loss functions demonstrate that the challenges of censored data can be addressed through careful formulation of the learning objective, enabling effective training even with incomplete observations.

Incorporation of Domain Expertise

A recurring theme has been the importance of domain knowledge in survival modeling:

- **Parameter constraints** encode known biological and clinical relationships, improving model generalization and robustness.
- **Distribution selection** informed by domain knowledge leads to more appropriate modeling choices for specific disease contexts.
- **Feature engineering** guided by clinical understanding enhances model interpretability and performance.

This integration of domain expertise with data-driven learning represents a balanced approach that leverages the strengths of both traditional statistical methods and modern deep learning techniques.

10.1.2 Current Challenges

Despite the significant progress in deep survival analysis, several challenges remain that warrant ongoing research and development efforts:

Interpretability and Explainability

While deep learning models offer superior predictive performance, their interpretability often lags behind traditional statistical approaches:

- **Black-box nature** of deep networks can limit their acceptance in clinical settings where decisions must be understood and justified.
- **Post-hoc explanation methods** like SHAP or LIME struggle with the complex temporal dependencies in survival data.
- **Balancing interpretability and performance** remains a persistent challenge, with inherent trade-offs between model complexity and transparency.

Developing models that are both highly performant and interpretable is crucial for the clinical adoption of deep survival analysis techniques.

Data Limitations

The effectiveness of deep learning approaches is often constrained by data availability and quality:

- **Sample size limitations** in many medical datasets make it difficult to fully leverage the representational capacity of deep models.
- **Selection bias** in observational data can lead to models that perpetuate or amplify existing disparities in healthcare.
- **Informative censoring** violates assumptions of many models and can lead to biased estimates if not properly addressed.

Developing techniques that are robust to these data limitations, such as transfer learning, data augmentation, and appropriate handling of missing data, remains an important area of research.

Evaluation Metrics and Benchmarking

The evaluation of survival models presents unique challenges:

- **Multiple competing metrics** (concordance, calibration, Brier score) capture different aspects of performance, making model comparison difficult.
- **Lack of standardized benchmarks** with consistent preprocessing and evaluation protocols hampers reproducible comparison of methods.
- **Clinical relevance** of statistical metrics is not always clear, leading to a gap between methodological advances and practical utility.

Establishing comprehensive evaluation frameworks that align with clinical decision-making is essential for meaningful progress in the field.

10.1.3 Future Directions

The field of deep survival analysis is rapidly evolving, with several promising directions for future research:

Integration with Multi-modal Data

Future survival models will increasingly leverage diverse data types:

- **Imaging data** integration with tabular clinical data can capture complex spatial patterns relevant to disease progression.
- **Genomic and molecular data** incorporation can identify biological mechanisms underlying disease trajectories.
- **Longitudinal measurements** and time series data can track disease evolution and response to interventions.

Developing architectures that effectively combine these heterogeneous data sources while respecting their unique characteristics represents a significant opportunity for improving predictive performance.

Causal Inference for Survival Outcomes

Moving beyond prediction to causal understanding is a crucial frontier:

- **Treatment effect estimation** with censored outcomes requires specialized approaches that combine causal inference with survival analysis.
- **Counterfactual reasoning** about survival times can inform personalized treatment decisions and policy interventions.
- **Mediation analysis** can identify mechanisms through which risk factors influence survival outcomes.

The integration of causal inference techniques with deep survival models offers the potential to move from predictive to prescriptive analytics in healthcare.

Federated and Privacy-Preserving Learning

Addressing privacy concerns while leveraging distributed data will be increasingly important:

- **Federated learning** approaches can train models across institutions without sharing raw patient data.
- **Differential privacy** techniques can provide formal guarantees against re-identification of individuals.
- **Synthetic data generation** can enable sharing of realistic but non-identifiable datasets for benchmarking and method development.

These approaches will be essential for scaling deep survival analysis to large, diverse populations while maintaining patient privacy and data security.

10.1.4 Practical Implementation Guidelines

For practitioners looking to implement deep survival analysis methods, we offer the following guidelines:

Model Selection

When choosing a survival model for a specific application:

- **Consider the primary objective** (e.g., risk stratification, time prediction, understanding risk factors) to guide the choice of model architecture and loss function.
- **Evaluate data characteristics** including sample size, censoring rate, and feature dimensionality to determine the appropriate model complexity.
- **Balance interpretability and performance** based on the specific requirements of the application context.

There is no one-size-fits-all approach; the best model depends on the specific context, data, and objectives of the analysis.

Implementation Best Practices

To ensure robust implementation and evaluation:

- **Use proper cross-validation** techniques that account for censoring and preserve the temporal structure of the data.
- **Evaluate multiple metrics** capturing different aspects of performance (discrimination, calibration, accuracy).
- **Conduct thorough sensitivity analyses** to assess model robustness to different assumptions and hyperparameter choices.
- **Compare against appropriate baselines**, including both traditional statistical methods and simpler machine learning approaches.

Rigorous methodology is essential for developing reliable and trustworthy survival models.

Deployment Considerations

When deploying survival models in clinical settings:

- **Develop clear visualization and explanation tools** that communicate model predictions and uncertainty to clinicians.
- **Implement monitoring systems** to detect distribution shifts and performance degradation over time.
- **Establish updating protocols** for retraining models as new data becomes available.
- **Integrate with existing clinical workflows** to minimize disruption and maximize adoption.

Thoughtful deployment strategies are as important as model development for realizing the potential benefits of survival analysis in practice.

10.1.5 Ethical Considerations

The development and deployment of survival models raise important ethical considerations:

Fairness and Equity

Ensuring that survival models do not perpetuate or amplify existing health disparities:

- **Evaluate model performance across demographic groups** to identify potential disparities in predictive accuracy.
- **Consider the representativeness of training data** and potential biases in data collection processes.
- **Implement fairness constraints or objectives** during model training when appropriate.

Models should be developed with an explicit commitment to promoting health equity rather than reinforcing existing disparities.

Transparency and Accountability

Fostering trust through transparent modeling practices:

- **Document model development decisions** including data preprocessing, feature selection, and hyperparameter choices.
- **Disclose limitations and uncertainties** in model predictions to end-users.
- **Establish clear lines of responsibility** for model outcomes when used in clinical decision-making.

Transparent reporting of model development and limitations is essential for responsible implementation.

Patient Autonomy and Shared Decision-Making

Respecting patient agency in the use of predictive models:

- **Present survival predictions in a way that supports informed decision-making** rather than dictating courses of action.
- **Acknowledge the probabilistic nature of predictions** and the importance of individual preferences in healthcare decisions.
- **Involve patients in the design and evaluation** of systems that will use survival predictions.

Survival models should enhance rather than replace the patient-provider relationship and shared decision-making processes.

10.1.6 Concluding Remarks

The integration of deep learning with survival analysis has opened new frontiers in predicting and understanding time-to-event outcomes with censored data (C. Lee et al., 2018; Nagpal, Li, and Dubrawski, 2021; Kvamme, Borgan, and Scheel, 2019). By combining the flexibility and representational power of neural networks with the statistical rigor of survival analysis, researchers have developed methods that push the boundaries of predictive performance while maintaining the ability to handle censoring and other complexities of survival data (Y. Zhong et al., 2021; Chapfuwa et al., 2018; Nagpal, Yadlowsky, et al., 2021).

As the field continues to evolve, the focus will increasingly shift from methodological innovations to practical implementation and impact. The true measure of success for deep survival analysis will be its ability to improve clinical decision-making, enhance patient outcomes, and advance our understanding of disease progression and treatment effects.

By addressing the challenges of interpretability, data limitations, and evaluation, while embracing new opportunities in multi-modal data integration, causal inference, and privacy-preserving learning, the field of deep survival analysis is poised to make significant contributions to healthcare and beyond. The journey from statistical foundations to deep learning innovations has been productive, but the most impactful work likely lies ahead as these methods mature and find their place in clinical practice and research.

Bibliography

Antolini, L., P. Boracchi, and E. Biganzoli (2005). “A time-dependent discrimination index for survival data”. In: *Statistics in Medicine* 24.24, pp. 3927–3944. DOI: `10.1002/sim.2427`.

Austin, P. C., D. S. Lee, and J. P. Fine (2016). “Introduction to the Analysis of Survival Data in the Presence of Competing Risks”. In: *Circulation* 133.6, pp. 601–609. DOI: `10.1161/CIRCULATIONAHA.115.017719`.

Biganzoli, E. et al. (2001). “Feed forward neural networks for the analysis of censored survival data: a partial logistic regression approach”. In: *Statistics in Medicine* 17.10, pp. 1169–1186. DOI: `10.1002/(SICI)1097-0258(19980530)17:10<1169::AID-SIM796>3.0.CO;2-D`.

Bishop, C. M. (2006). “Pattern Recognition and Machine Learning”. In: *Springer*.

Burges, C. et al. (2005). “Learning to rank using gradient descent”. In: *Proceedings of the 22nd International Conference on Machine Learning*, pp. 89–96. DOI: `10.1145/1102351.1102363`.

Chapfuwa, P. et al. (2018). “Adversarial time-to-event modeling”. In: *Proceedings of the 35th International Conference on Machine Learning* 80, pp. 735–744.

Cox, D. R. (1972). “Regression models and life-tables”. In: *Journal of the Royal Statistical Society, Series B* 34.2, pp. 187–220. DOI: `10.1111/j.2517-6161.1972.tb00899.x`.

Fine, J. P. and R. J. Gray (1999). “A proportional hazards model for the subdistribution of a competing risk”. In: *Journal of the American Statistical Association* 94.446, pp. 496–509. DOI: `10.1080/01621459.1999.10474144`.

Fotso, S. (2018). “Deep Neural Networks for Survival Analysis Based on a Multi-Task Framework”. In: *arXiv preprint arXiv:1801.05512*.

Gensheimer, M.F. and B. Narasimhan (2019). “A scalable discrete-time survival model for neural networks”. In: *PeerJ* 7, e6257. DOI: `10.7717/peerj.6257`.

Ghassemi, M., L. Oakden-Rayner, and A.L. Beam (2022). “The false hope of current approaches to explainable artificial intelligence in health care”. In: *The Lancet Digital Health* 3.11, e745–e750. DOI: `10.1016/S2589-7500(21)00208-9`.

Goldberg, D. (1991). “What every computer scientist should know about floating-point arithmetic”. In: *ACM Computing Surveys* 23.1, pp. 5–48. DOI: `10.1145/103162.103163`.

Graf, E. et al. (1999). “Assessment and comparison of prognostic classification schemes for survival data”. In: *Statistics in Medicine* 18.17-18, pp. 2529–2545. DOI: `10.1002/(SICI)1097-0258(19990915/30)18:17/18<2529::AID-SIM274>3.0.CO;2-5`.

Hanin, B. and D. Rolnick (2022). “Deep ReLU Networks Have Surprisingly Few Activation Patterns”. In: *Advances in Neural Information Processing Systems* 32, pp. 359–368.

Harrell, F. E. et al. (1982). “Evaluating the yield of medical tests”. In: *JAMA* 247.18, pp. 2543–2546. DOI: `10.1001/jama.1982.03320430047030`.

He, K. et al. (2020). “Momentum Contrast for Unsupervised Visual Representation Learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738. DOI: `10.1109/CVPR42600.2020.00975`.

Ibrahim, J.G., M.H. Chen, and D. Sinha (2001). “Bayesian Survival Analysis”. In: *Springer Series in Statistics*. doi: 10.1007/978-1-4757-3447-8.

Kaplan, E. L. and P. Meier (1958). “Nonparametric estimation from incomplete observations”. In: *Journal of the American Statistical Association* 53.282, pp. 457–481. doi: 10.1080/01621459.1958.10501452.

Karimi, B., B. Schölkopf, and I. Valera (2021). “Algorithmic recourse: from counterfactual explanations to interventions”. In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 353–362.

Katzman, J. L. et al. (2018). “DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network”. In: *BMC Medical Research Methodology* 18.1, pp. 1–12. doi: 10.1186/s12874-018-0482-1.

Kendall, A., Y. Gal, and R. Cipolla (2018). “Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7482–7491. doi: 10.1109/CVPR.2018.00781.

Kleinbaum, D. G. and M. Klein (2012). “Survival Analysis: A Self-Learning Text”. In: *Springer Science & Business Media*.

Koenker, R. and O. Geling (2001). “Reappraising Medfly Longevity: A Quantile Regression Survival Analysis”. In: *Journal of the American Statistical Association* 96.454, pp. 458–468. doi: 10.1198/016214501753168172.

Koller, M. T. et al. (2012). “Competing risks and the clinical community: irrelevance or ignorance?” In: *Statistics in Medicine* 31.11-12, pp. 1089–1097. doi: 10.1002/sim.4384.

Kuo, Y. et al. (2020). “Interpretable survival analysis with very deep networks”. In: *IEEE International Conference on Big Data*, pp. 2091–2100.

Kvamme, H. (2021). “Continuous and Discrete-Time Survival Prediction with Neural Networks”. In: *IEEE Transactions on Neural Networks and Learning Systems* 33.9, pp. 4963–4977. doi: 10.1109/TNNLS.2021.3082212.

Kvamme, H., Ø. Borgan, and I. Scheel (2019). “Time-to-event prediction with neural networks and Cox regression”. In: *Journal of Machine Learning Research* 20.129, pp. 1–30.

Lee, C. et al. (2018). “DeepHit: A deep learning approach to survival analysis with competing risks”. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 2314–2321.

Lin, T. Y. et al. (2017). “Focal Loss for Dense Object Detection”. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988. doi: 10.1109/ICCV.2017.324.

McLachlan, G. J. and K. E. Basford (1988). “Mixture Models: Inference and Applications to Clustering”. In: *Marcel Dekker*.

Nagpal, C., X. Li, and A. Dubrawski (2021). “Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks”. In: *IEEE Journal of Biomedical and Health Informatics* 25.8, pp. 3163–3175. doi: 10.1109/JBHI.2021.3053137.

Nagpal, C., S. Yadlowsky, et al. (2021). “Deep parametric time-to-event regression with time-varying covariates”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.10, pp. 9346–9354.

Prentice, R. L. et al. (1978). “The analysis of failure times in the presence of competing risks”. In: *Biometrics* 34.4, pp. 541–554. doi: 10.2307/2530374.

Radfar, S. et al. (2022). “A Survey on Expert Knowledge-Guided Neural Networks: Applications, Challenges, and Research Directions”. In: *arXiv preprint arXiv:2207.04974*.

Ranganath, R. et al. (2016). “Deep Exponential Families for Survival Analysis”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 30.1. doi: 10.1609/aaai.v30i1.10186.

Rudin, C. (2019). “Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead”. In: *Nature Machine Intelligence* 1.5, pp. 206–215. DOI: 10.1038/s42256-019-0048-x.

Steiner, L. M. et al. (2021). “Using deep learning for focal loss to predict mortality in patients with multiple chronic diseases”. In: *BMC Medical Informatics and Decision Making* 21.1, pp. 1–14. DOI: 10.1186/s12911-021-01636-1.

Tagasovska, N. and D. Lopez-Paz (2019). “Single-Model Uncertainties for Deep Learning”. In: *Advances in Neural Information Processing Systems* 32, pp. 6417–6428.

Wiegrebe, S. et al. (2023). “Deep Learning-based Competing Risk Survival Analysis Using Cross-Sectional Imaging Data”. In: *Radiology: Artificial Intelligence* 5.2, e220159. DOI: 10.1148/ryai.220159.

Xia, F. et al. (2008). “ListMLE: A consistent list-wise approach to learning to rank”. In: *Proceedings of the 31st Annual International ACM SIGIR Conference*, pp. 703–704. DOI: 10.1145/1390334.1390458.

Zhong, X. and J.H. Jeong (2020). “Estimation of median survival time for COVID-19 patients: A pseudo-value approach”. In: *Science Advances* 6.27, eabd4900. DOI: 10.1126/sciadv.abd4900.

Zhong, Y. et al. (2021). “MENSA: Multi-Event Neural Survival Analysis”. In: *arXiv preprint arXiv:2112.09823*.