Deep Learning Methods for Survival Analysis

Ying Ding, Lang Zeng, Na Bo

Department of Biostatistics and Health Data Science

School of Public Health

University of Pittsburgh

Outline

- Part 1 – Deep Learning for Survival Predictions

  - Neural networks for right-censored survival data with **time-independent** or **time-dependent** covariates
  Case Study 1: Prediction of Progression of AMD (Age-related Macular Degeneration)

  - Neural networks for interval-censored (and left truncated) survival data
  Case Study 2: Prediction of Development of AD (Alzheimer's Disease)

Outline

- Part 2 – Deep Learning for Causal Survival Analysis

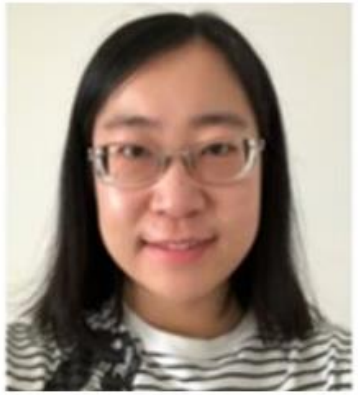
  - CATE (conditional average treatment effect) for survival outcomes

  - Deep learning approaches for estimating CATE with survival outcomes

  - *Case Study 3: Individualized Treatment Effects of AREDS formula in delaying Progression to late-AMD*

Acknowledgement

Lang Zeng



Na Bo

Haoling Wang

Zhiyu Sui

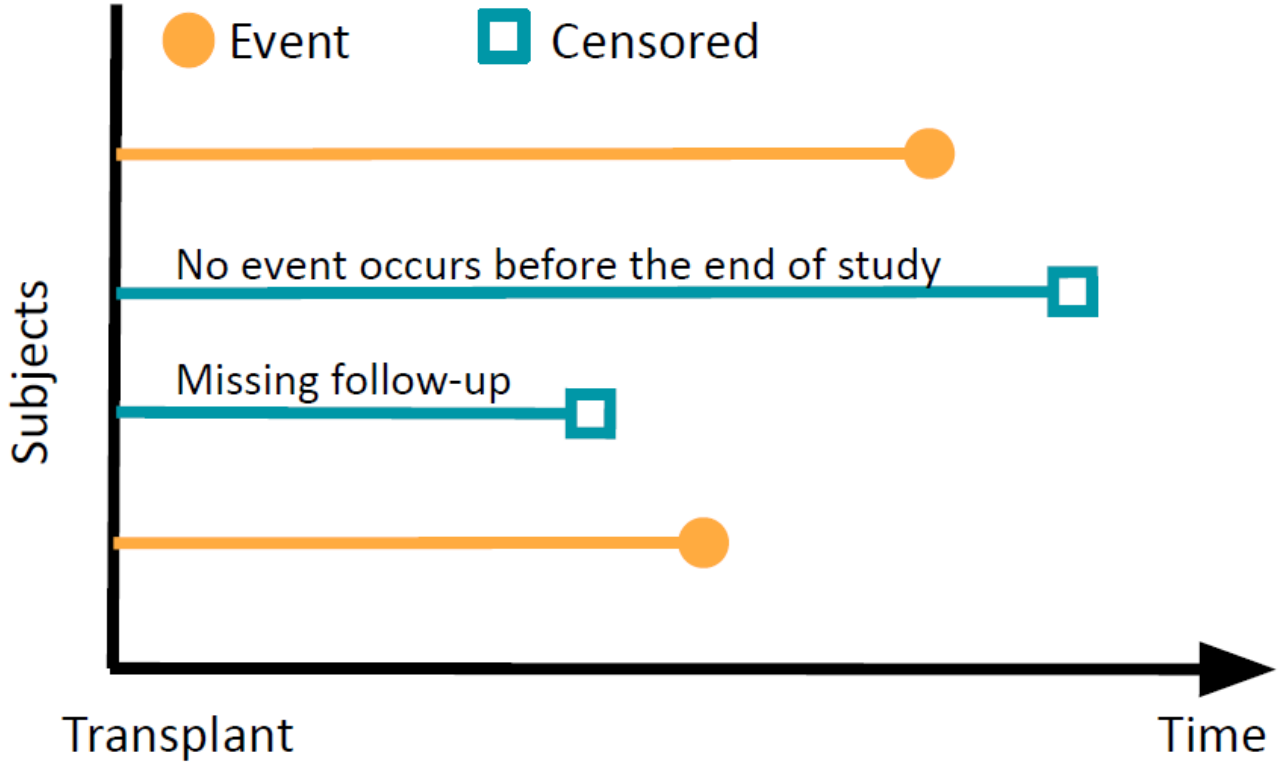
Tao Sun

Yue (Luna) Wei

Part I: Deep Learning for Survival Predictions

1.1 Neural networks for right-censored survival data

Survival Data



Goal: Covariates X ⟷ Association ⟷ Outcome: Time to Event T

Censoring: the events of interest are not observed.

Right-censored data: $Y = \min\{T, C\}, \Delta = 1(T \leq C), X$

Survival Prediction (with time-independent covariates)

$$\Pr(T > t|X)$$

Dynamic Survival Prediction (with time-dependent covariates)

For $t > s$ $\Pr(T > t|\{X(u), 0 \leq u \leq s\}, T > s)$

Useful Functions in Survival Analysis

$$\Pr(T > t|X = x) = S(t|X = x) = \exp\big(-\Lambda(t|x)\big) = \exp\left(-\int_0^t \lambda\,(u|x)\mathrm{d}u\right)$$

Survival function: $S(t|X = x)$

Hazard function: $\lambda(t|X = x)$

Cumulative hazard function: $\Lambda(t|X = x)$

CoxPH: Cox Proportional Hazards Model

$$\Pr(T > t|X = x) = \exp\big(-\Lambda(t|x)\big) = \exp\left(-\int_0^t \lambda\,(u|x)\mathrm{d}u\right)$$

$$= \exp\left(-\int_0^t \lambda_0(u)\exp(\beta x)\mathrm{d}u\right)$$

Proportional Hazards Assumption: $\lambda(u|x) = \lambda_0(u)\exp(\beta x)$

Partial Likelihood for CoxPH Model

$$\Pr(T > t | X = x) = \exp\left(-\int_0^t \lambda_0(u)\exp(\beta x)\mathrm{d}u\right)$$

Right-censored data: $Y = \min\{T, C\}, \Delta = 1(T \leq C), X$

β can be optimized by minimizing the $l(\beta) = -\log pL(Y, \Delta, X)$

$$l(\beta) \coloneqq -\frac{1}{n}\sum_{i:\Delta_i=1}[\beta x_i - \log\sum_{j\in R(Y_i)}\exp(\beta x_j)]$$

Survival Probability Prediction under CoxPH

$$\Pr(T > t|X = x) = \exp\left(-\int_0^t \lambda_0(u)\exp(\beta x)\mathrm{d}u\right)=\exp(-\Lambda_0(t)\exp(\beta x))$$

After solving $\hat{\beta}$ by minimizing $l(\beta)$, we also obtain

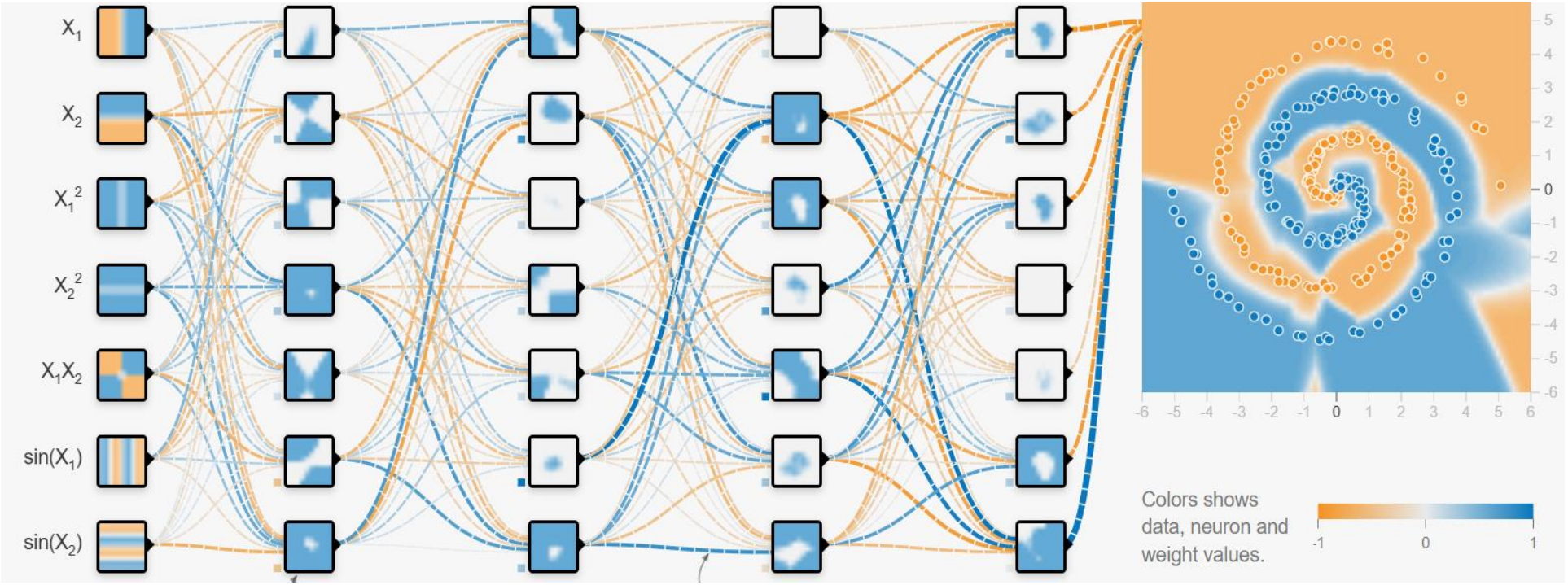
$$\widehat{\Lambda}_0(\cdot) = \widehat{\Lambda}_0(\cdot\,|\hat{\beta})$$

Then

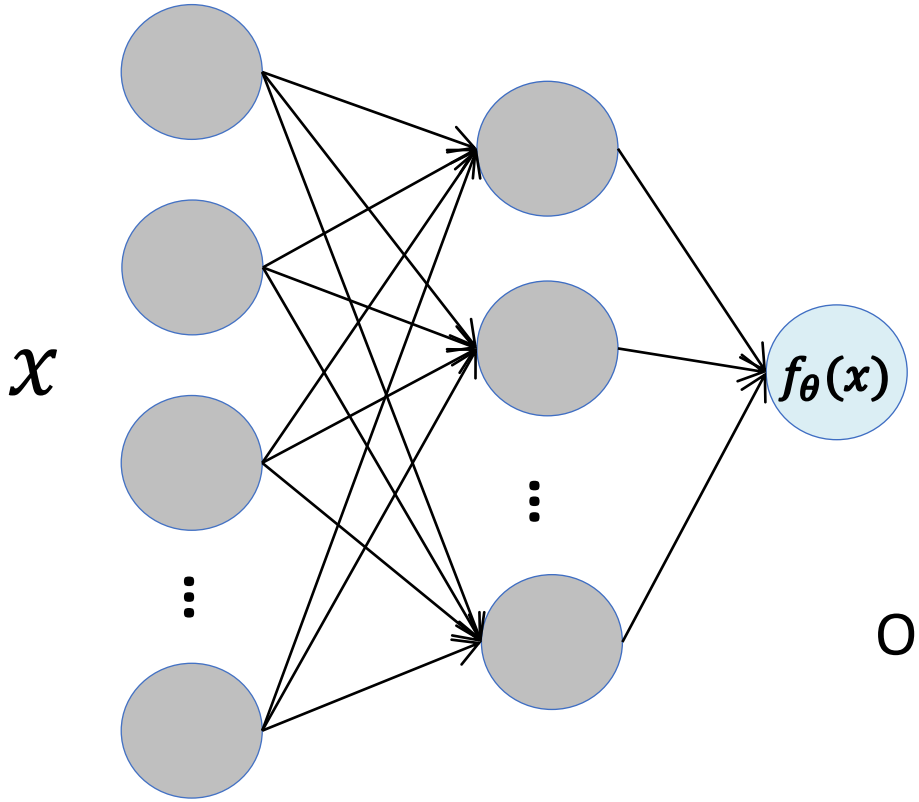
$$\hat{\Pr}(T > t|X = x) = \exp\left(-\widehat{\Lambda}_0(t)\exp(\hat{\beta}x)\right)$$

Deep Learning

- Neural-network-based machine learning method.
- A neural network with multiple hidden layers is called a “deep” network, hence the name “deep learning” (LeCun et al. 2015).
- A deep/wide neural network can theoretically approximate any continuous function (Cybenko 1989).



A simple Neural Network (NN) (for continuous Y)



- Observed data: (**x**, y)
- Input nodes: of NN: **x**
- Output node of NN: $f_\theta(x)$
- Parameters to be optimized: θ

Optimize θ by minimizing a loss function $L(\theta)$

$$L(\theta) = (f_\theta(x) - y)^2$$

Optimization Algorithms for Training NN

- Gradient Descent (GD): computes the gradient of the loss function with respect to the model parameters using the **entire** training dataset in each iteration.
  - Stable training

- Stochastic Gradient Descent (SGD): computes the gradient using **a small subset** of examples in each iteration.
  - Memory efficient
  - Introduces **noise** that can help escape local minima or saddle points.

Evaluation Metrics

- C-index

$$\frac{\sum_{i,j} \Delta_i I(T_i < T_j) I(\hat{T}_i < \hat{T}_j)}{\sum_{i,j} \Delta_i I(T_i < T_j)}$$

- Mean Square Error (MSE): (Only for simulation when true survival function S_0 is known)

$$\frac{1}{n} \sum_i \frac{1}{T_i} \int_0^{T_i} \left\{ S_0(u|X_i) - \hat{S}(u|X_i) \right\}^2 \mathrm{d}u$$

- Integrated Brier Score (IBS) score:

$$\frac{1}{n} \sum_i \frac{1}{T_i} \int_0^{T_i} \left\{ I(u < T_i) - \hat{S}(u|X_i) \right\}^2 \mathrm{d}u$$

Deep Learning Models for Survival Data

- CoxPH+NN

Continuous-Time Models:

Method	Description	Example
CoxTime	Cox-Time is a relative risk model that extends Cox regression beyond the proportional hazards [1].	notebook
CoxCC	Cox-CC is a proportional version of the Cox-Time model [1].	notebook
CoxPH (DeepSurv)	CoxPH is a Cox proportional hazards model also referred to as DeepSurv [2].	notebook
PCHazard	The Piecewise Constant Hazard (PC-Hazard) model [12] assumes that the continuous-time hazard function is constant in predefined intervals. It is similar to the Piecewise Exponential Models [11] and PEANN [14], but with a softplus activation instead of the exponential function.	notebook

[Katzman, J. 2018]

- SODEN [Tang, W. 2022]: Survival ODE+NN
- SuMo-net [Rindt, D. 2022]: Monotone NN

Deep Learning Models for Survival Data

Discrete-Time Models:

Method	Description	Example
LogisticHazard (Nnet-survival)	The Logistic-Hazard method parametrize the discrete hazards and optimize the survival likelihood [12] [7]. It is also called Partial Logistic Regression [13] and Nnet-survival [8].	notebook
PMF	The PMF method parametrize the probability mass function (PMF) and optimize the survival likelihood [12]. It is the foundation of methods such as DeepHit and MTLR.	notebook
DeepHit, DeepHitSingle	DeepHit is a PMF method with a loss for improved ranking that can handle competing risks [3].	single competing
MTLR (N-MTLR)	The (Neural) Multi-Task Logistic Regression is a PMF methods proposed by [9] and [10].	notebook
BCESurv	A method representing a set of binary classifiers that remove individuals as they are censored [15]. The loss is the binary cross entropy of the survival estimates at a set of discrete times, with targets that are indicators of surviving each time.	bs_example

DeepHit [Lee, C. 2018]

Dr. Haavard Kvamme: https://github.com/havakv/pycox

pycox

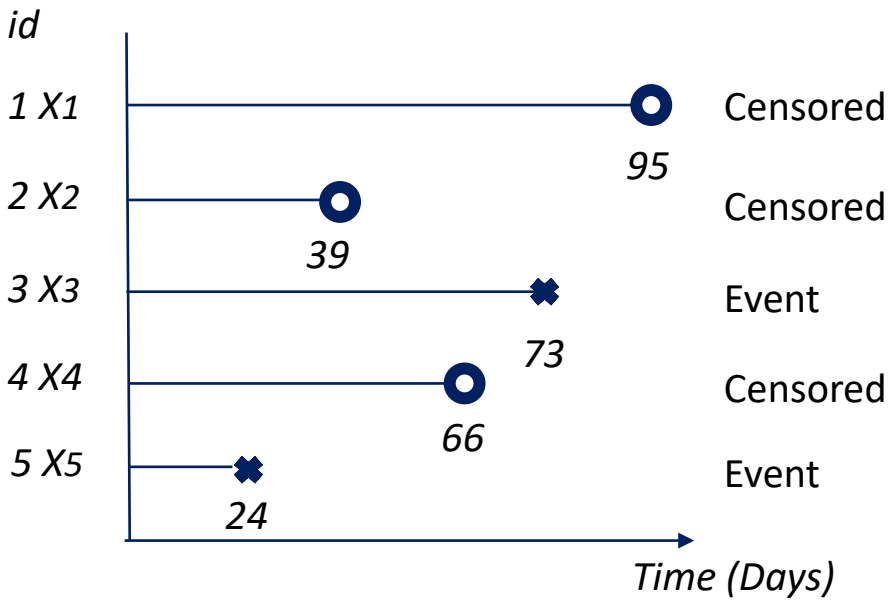
Time-to-event prediction with PyTorch

Get Started • Methods • Evaluation Criteria • Datasets • Installation • References

pycox is a python package for survival analysis and time-to-event prediction with PyTorch, built on the torchtuples package for training PyTorch models. An R version of this package is available at survivalmodels.

The package contains implementations of various survival models, some useful evaluation metrics, and a collection of event-time datasets. In addition, some useful preprocessing tools are available in the `pycox.preprocessing` module.

Data Format for Continuous-time Models:

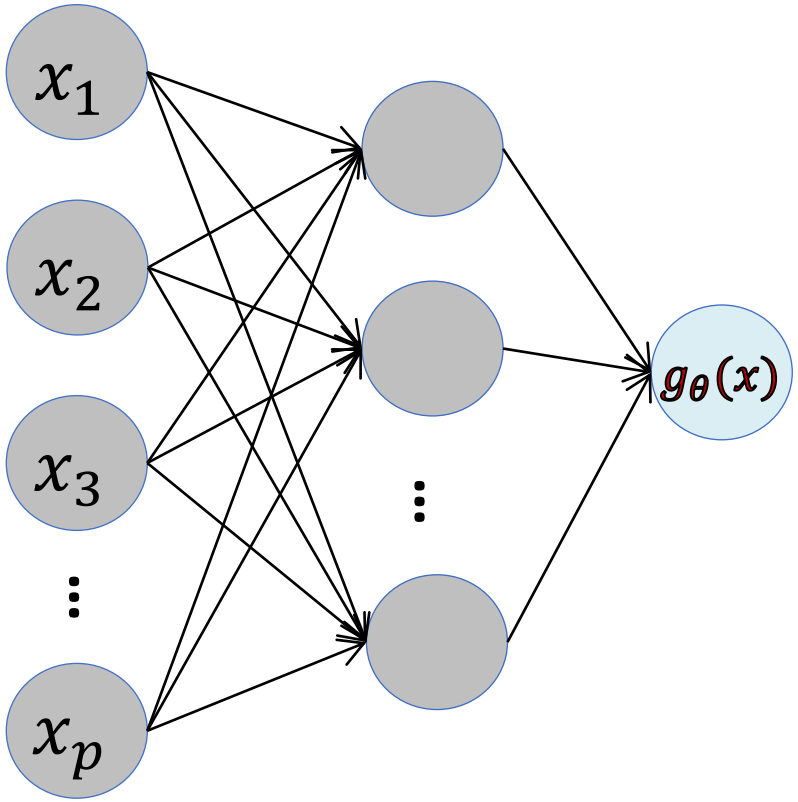


Subject	Time	Event	X
1	95	0	…
2	39	0	…
3	73	1	…
4	66	0	…
5	24	1	…
…	…	…	…

DeepSurv: CoxPH + NN

$$\Pr(T > t|X = x) = \exp\left(-\int_0^t \lambda_0(u)\exp(g_\theta(x))\mathrm{d}u\right)$$

Risk Score



- Observed data: (Y, Δ,X)
- Input nodes of NN: x
- Output node of NN: $g_\theta(x)$
- Parameters to be optimized: θ

$$l(\theta) := -\frac{1}{n_E}\sum_{i:\Delta_i=1}\left[g_\theta(x_i) - \log\sum_{j\in R(Y_i)}\exp\left(g_\theta(x_j)\right)\right]$$

$$\widehat{\Pr}(T > t|X = x) = \exp\left(-\widehat{\Lambda}_0(t|g_{\widehat{\theta}})\exp(g_{\widehat{\theta}}(x)\right)$$

DeepSurv: Theoretical Guarantee

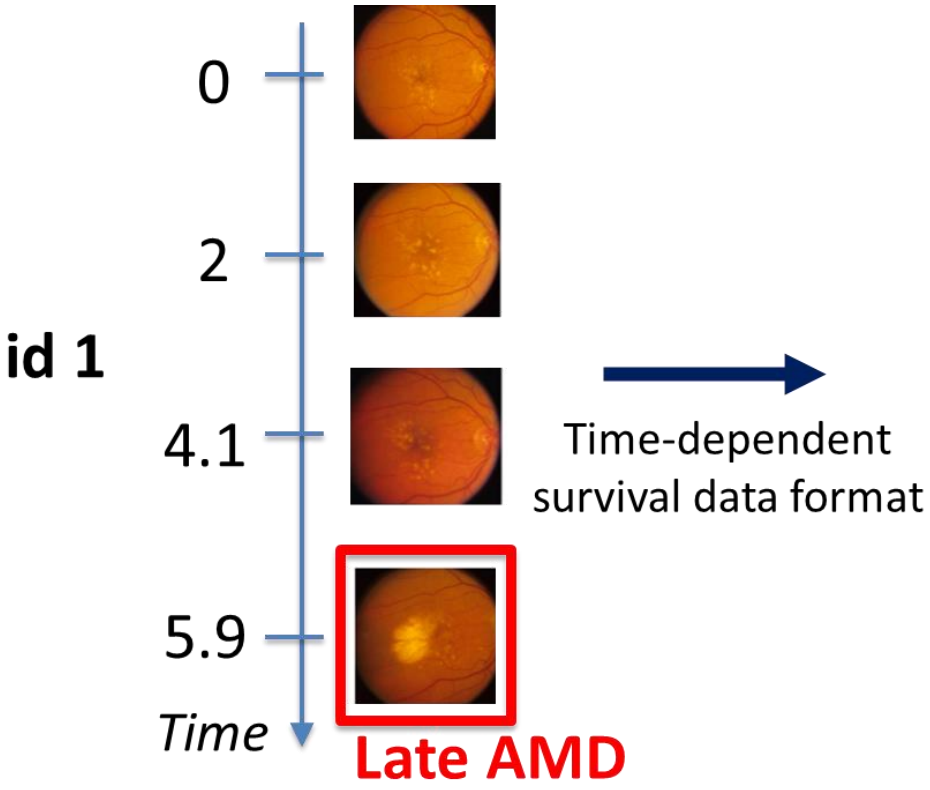
$$\Pr(T > t|X = x) = \exp\left(-\int_0^t \lambda_0(u)\exp(g_\theta(x))\mathrm{d}u\right)$$

Risk Score

[Zhong, Q. 2022] established the statistical foundations for DeepSurv

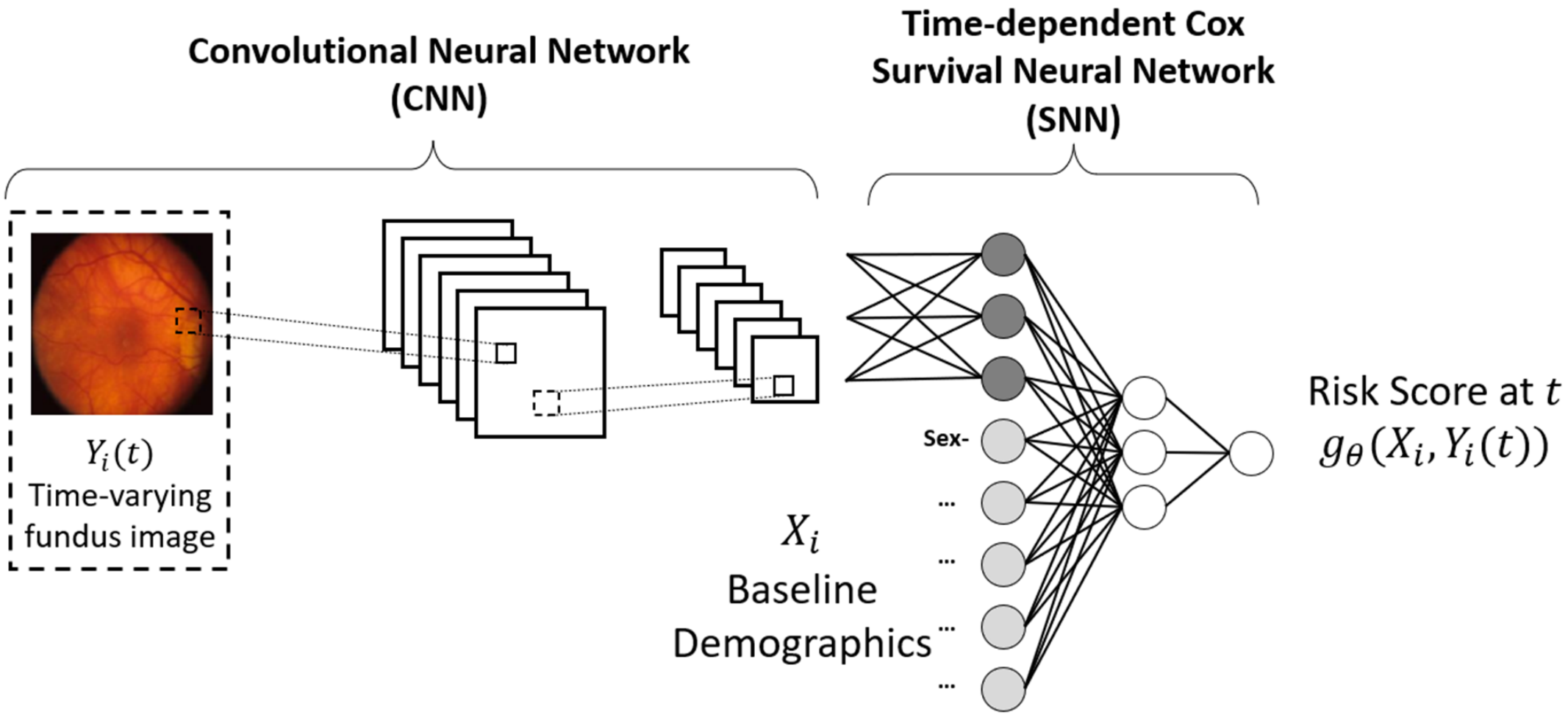
[Zeng, L. 2025+] established the statistical properties and offered practical guidelines when using mini-batch estimator via SGD to solve DeepSurv

tdCoxSNN: Time-dependent Cox Survival Neural Network



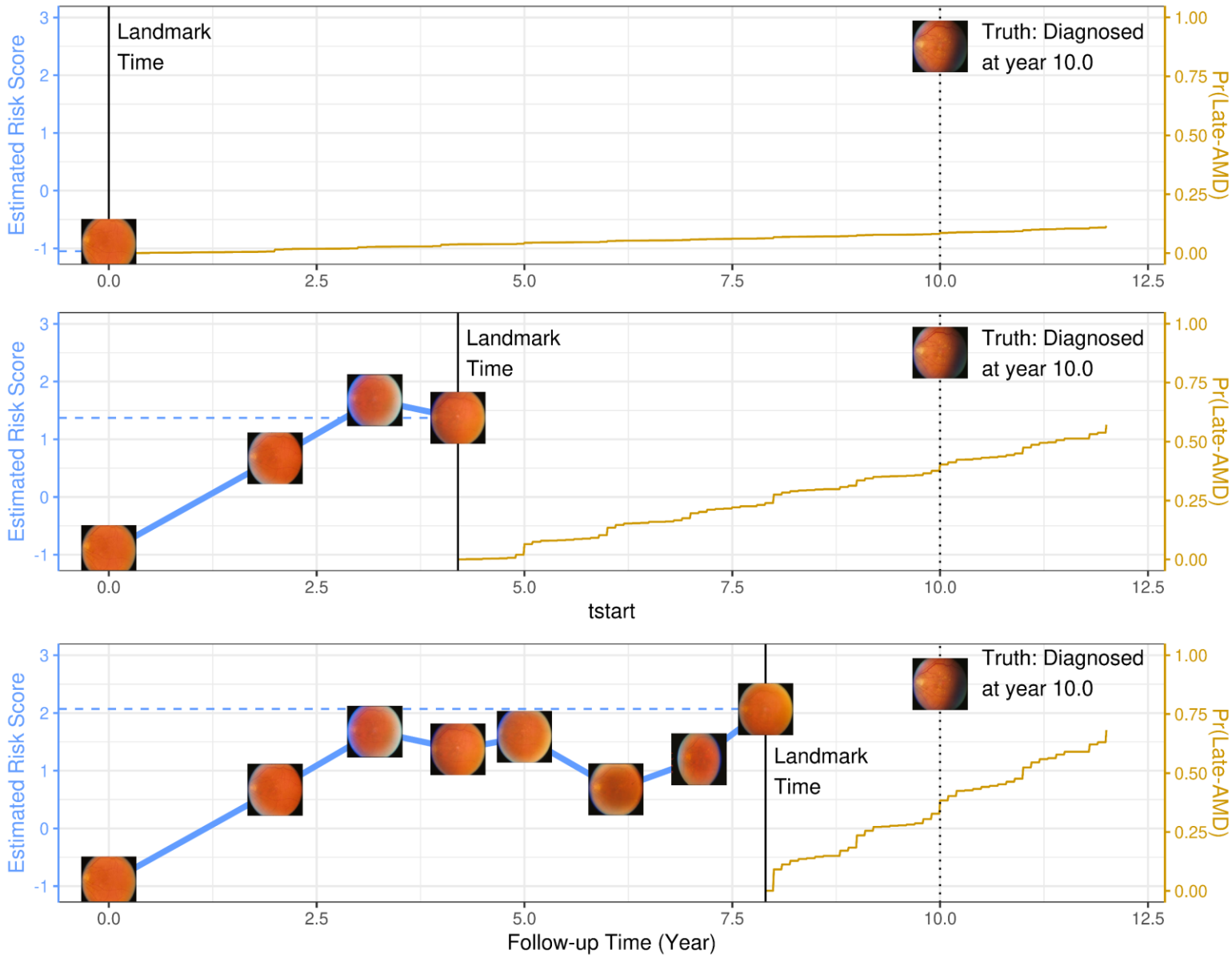
id	tstart	tstop	Event	Fundus Image	Sex	…
1	0	2	0		Female	…
1	2	4.1	0		Female	…
1	4.1	5.9	1		Female	…
2	…	…	…	…	…	…

tdCoxSNN



The PyTorch, Tensorflow, and R-Tensorflow version can be found at https://github.com/langzeng/tdCoxSNN.

Dynamic Prediction with Longitudinal Fundus Images



Relax the Proportional Hazards assumption:

SuMo-net:

$$\Pr(T > t|X = x) = \exp\big(-\Lambda(t, x)\big)$$

Directly model the cumulative hazard function through Monotone-NN

SODEN:

$$\Pr(T > t|X = x) = \exp\left(-\int_0^t \lambda(u, x)\,\mathrm{d}u\right)$$

Directly model the hazard function through NN with positive output

Fit the NN by maximizing the full likelihood: $\max \sum_i \Delta_i \log[\lambda(Y_i|X_i)] - \Lambda(Y_i|X_i)$

Survival ODE

$\Lambda(t|X_i)$ is the solution of the following Ordinary Differential Equation (ODE):

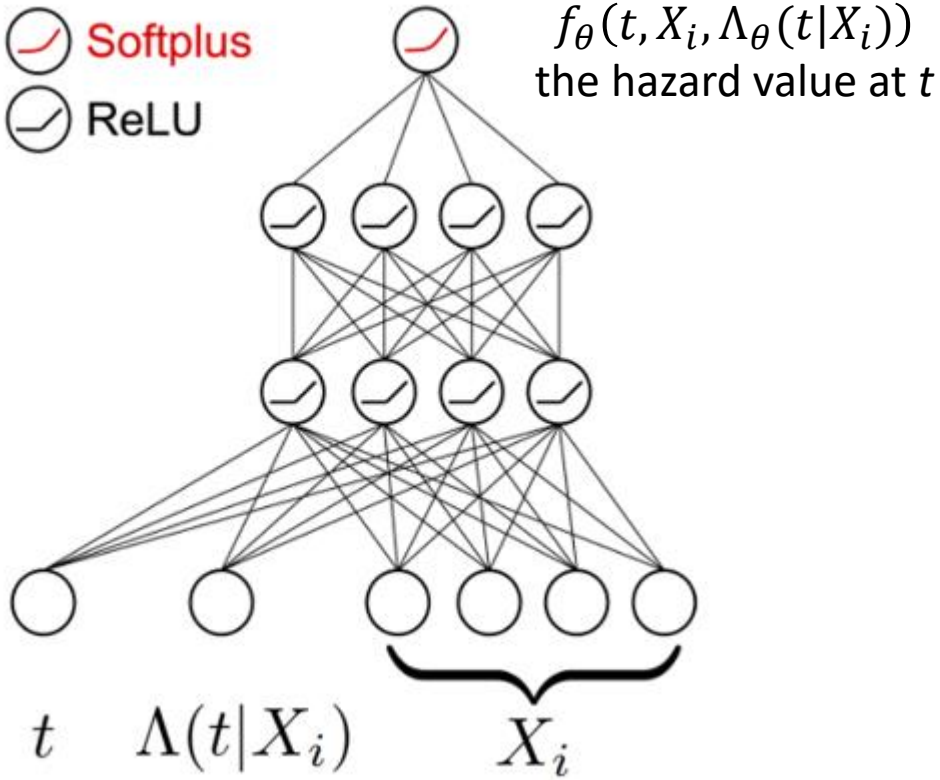
$$\begin{cases} \dfrac{\mathrm{d}\Lambda(t|X_i)}{\mathrm{d}t} = \lambda(t|X_i) \\ \Lambda(0|X_i) = 0 \end{cases}$$

- Obtain $\Lambda(Y_i|X_i)$ through numerical approximation.
- Note that ODE constraints are for every individual.

SODEN: Survival ODE+NN

Model $\lambda(t|X_i)$ through a neural network $f_\theta(t, X_i, \Lambda_\theta(t|X_i))$

$$\begin{cases} \dfrac{\mathrm{d}\Lambda_\theta(t|X_i)}{\mathrm{d}t} = f_\theta(t, X_i, \Lambda_\theta(t|X_i)) \\ \Lambda_\theta(0|X_i) = 0 \end{cases}$$



- Observed data: (Y, Δ, X)
- Input nodes of NN: t, X, Λ_θ
- Output node of NN: λ at t
- Parameters to be optimized: θ

Gradient-Based Optimization Algorithms for SODEN

- Loss function: $l(\theta) = \sum_i \left\{ -\Delta_i \log \left[\frac{\mathrm{d}\Lambda_\theta(t|X_i)}{\mathrm{d}t} |_{t=Y_i} \right] + \Lambda_\theta(Y_i|X_i) \right\}$

- Training procedure:
  - Step 1: Given current parameter θ, compute $\Lambda_\theta(Y_i|X_i)$ and its gradient w.r.t. to θ by solving*

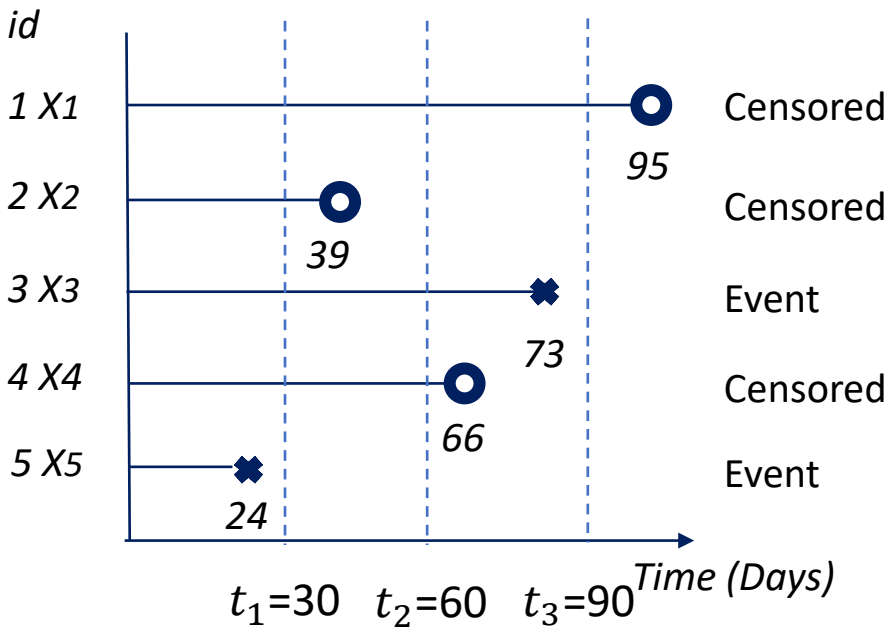
$$\begin{cases} \dfrac{\mathrm{d}\Lambda_\theta(t|X_i)}{\mathrm{d}t} = f_\theta(t, X_i, \Lambda_\theta(t|X_i)) \\ \Lambda_\theta(0|X_i) = 0 \end{cases}$$

  - Step 2: Update the parameter θ using the quantities calculated in step 1.

- Prediction: $\widehat{\Pr}(T > t|X = x) = \exp\left(-\widehat{\Lambda}_\theta(t|X_i)\right)$

*The ODE is solved using package "*torchdiffeq*"

Data Format for Discrete-time Models:



Subject	Time	Event	x
1	90	0	…
2	30	0	…
3	90	1	…
4	60	0	…
5	30	1	…
…	…	…	…

DeepHit: Discrete-time Survival Model + NN

$$t^* \in \{0, t_1, t_2, \ldots, t_m\}$$

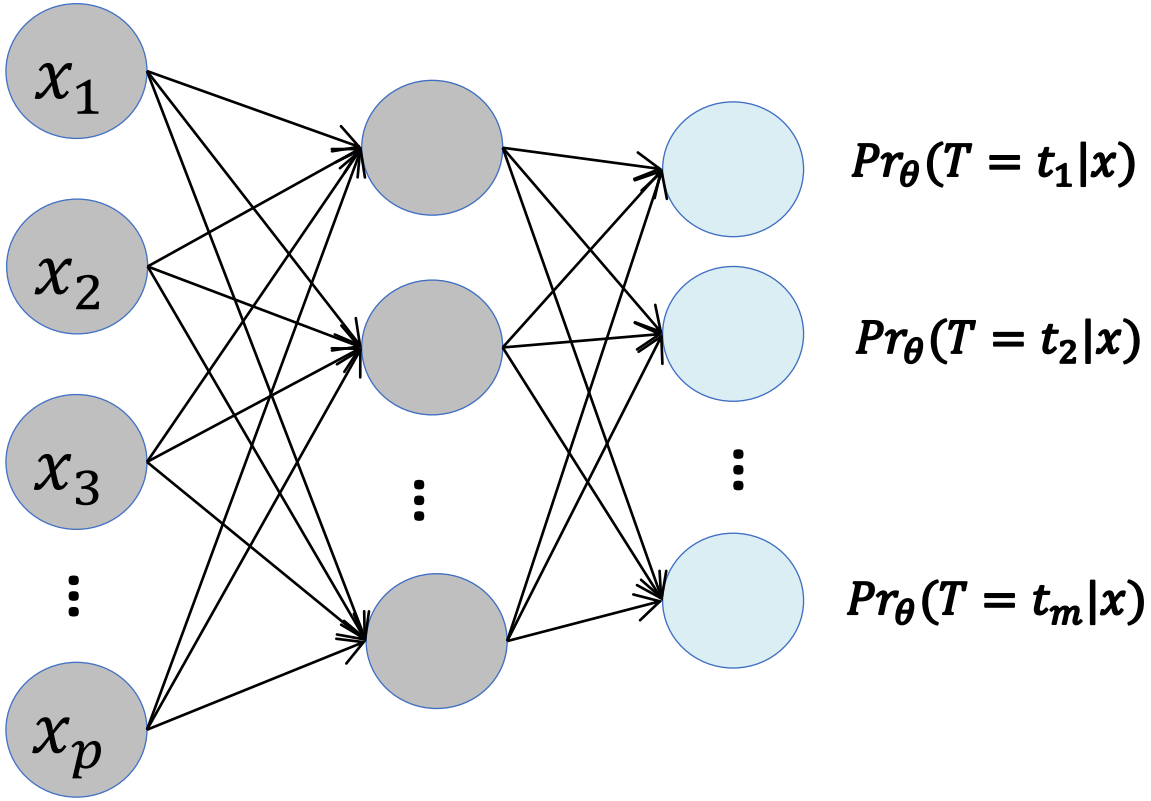
$$\Pr(T > t^*|X) = 1 - \Pr(T \leq t^*|X) = 1 - \sum_{s=t_1}^{t^*} \Pr(T = s|X)$$

Discretize the time:

- m: number of time points
- Discretization strategy

$$\Pr(T_d = t_j|X) := \Pr(t_{j-1} < T_c \leq t_j \ |X)$$

DeepHit: Discrete-time Survival Model + NN



- Observed data: (Y, Δ,X)
- Input nodes of NN: x
- Output nodes of NN:

  $(Pr_\theta(t_1|x), \dots, Pr_\theta(t_m|x))$
- Parameters to be optimized: θ

DeepHit: Loss Function

$$l(\theta) = L_1 + \alpha L_2$$

$$L_1 : -\log Likelihood(Y, \Delta, X)$$
$$= -\sum \{I(\Delta_i = 1) \log[\Pr_\theta(T = Y_i | X_i)] + I(\Delta_i = 0) \log[\Pr_\theta(T > Y_i | X_i)]\}$$

$$L_2 : Pairwise\ disconcordance\ (Ordering\ Error)$$
$$= \sum_{\text{all pairs } (i,j)} I\left(\Delta_i = 1, Y_i < Y_j\right) \exp\left(-\frac{\Pr_\theta(T \leq Y_i | X_i) - \Pr_\theta\left(T \leq Y_j | X_j\right)}{\sigma}\right)$$

A patient who died at time *T* should have higher risk at time *T* than a patient who survived longer than *T*.

Example with Codes

Primary Biliary Cirrhosis (PBC) disease
Dataset: “pbc2” (Mayo Clinic)

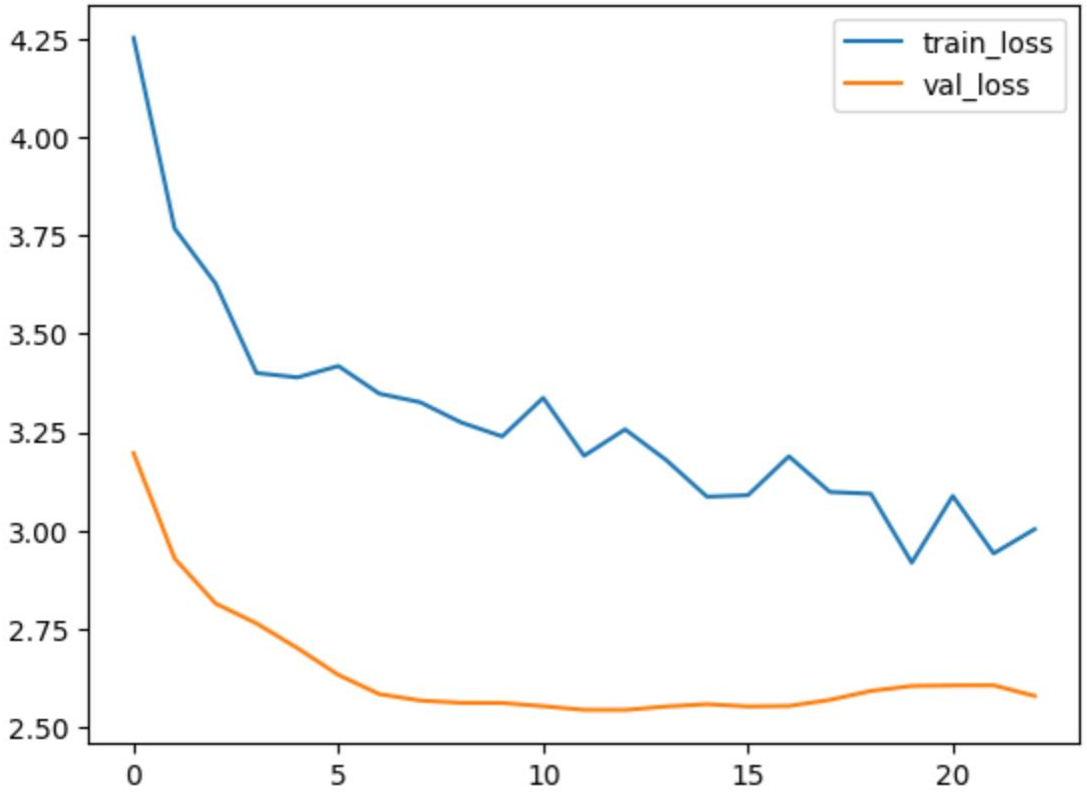
Consists of 312 subjects
Outcome of interest: time-to-liver transplant
Predictors:

- 12 lab variables (7 continuous lab tests, e.g., albumin; 5 categorical, e.g., liver enlargement)
- 3 other variables (sex, age at start-of-study, and treatment indicator)

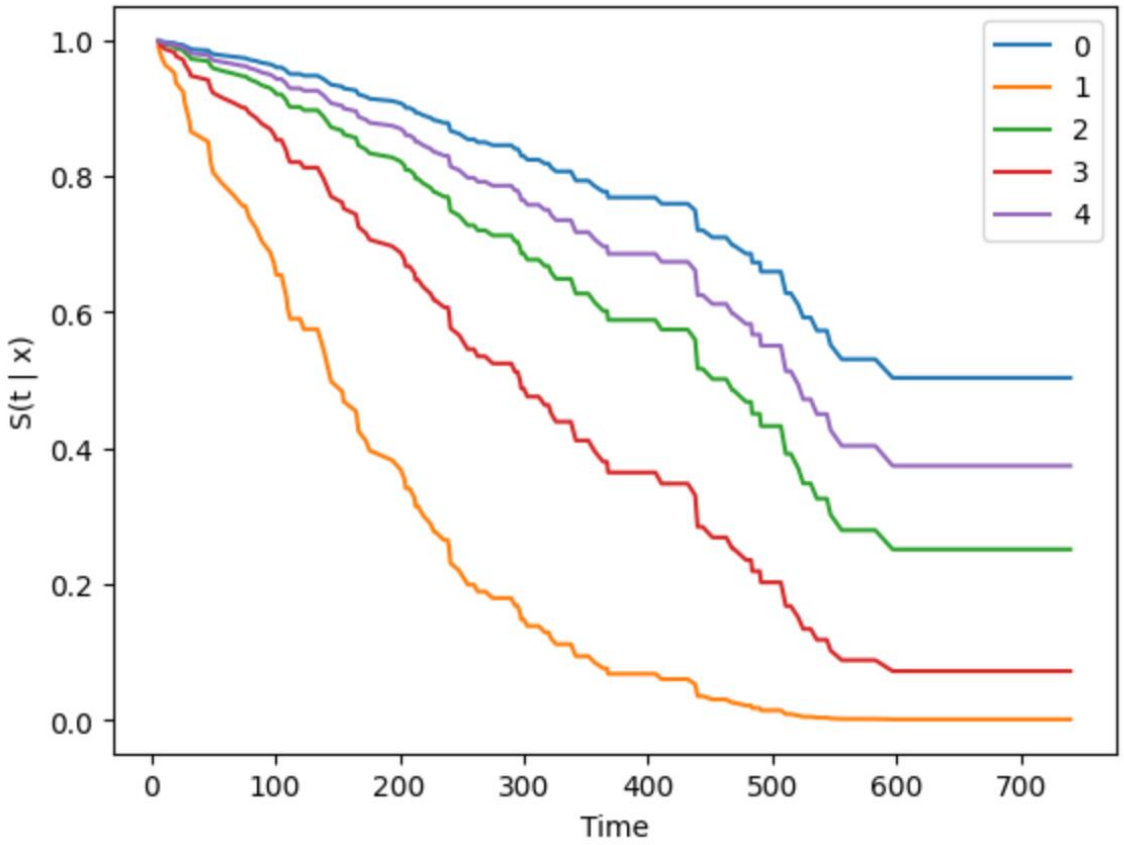
“DeepSurv_DeepHit_PBC.ipynb”

DeepSurv Result

Training



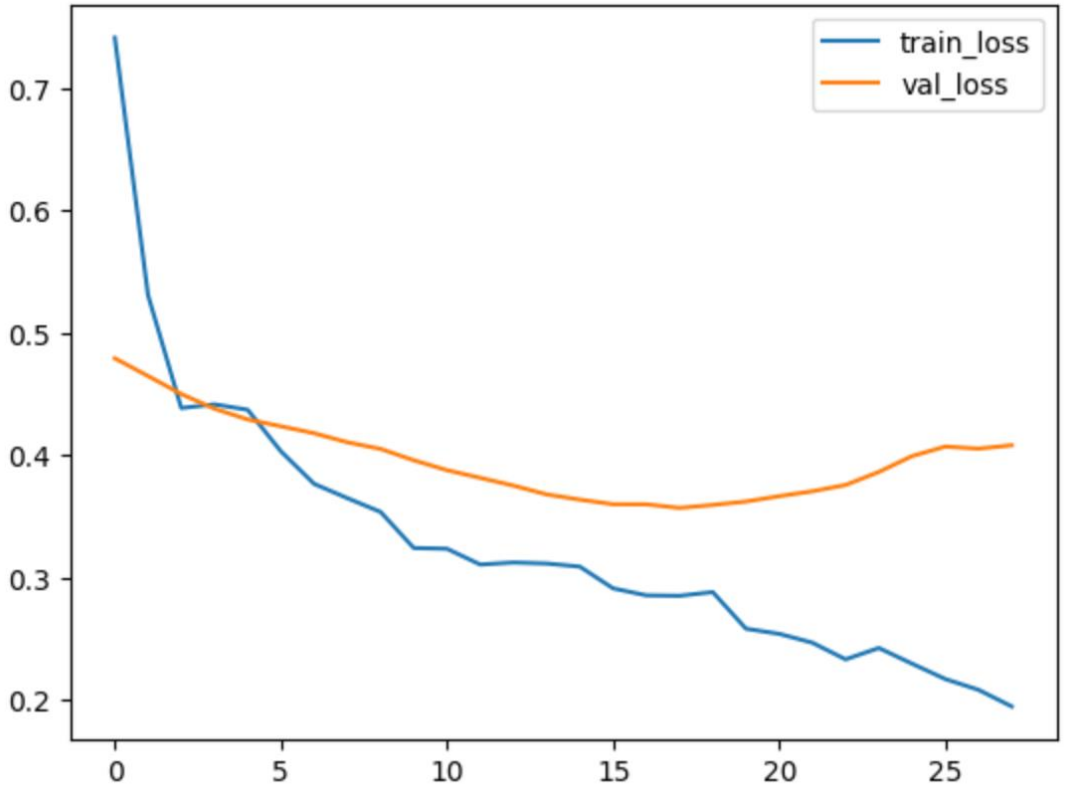
Prediction



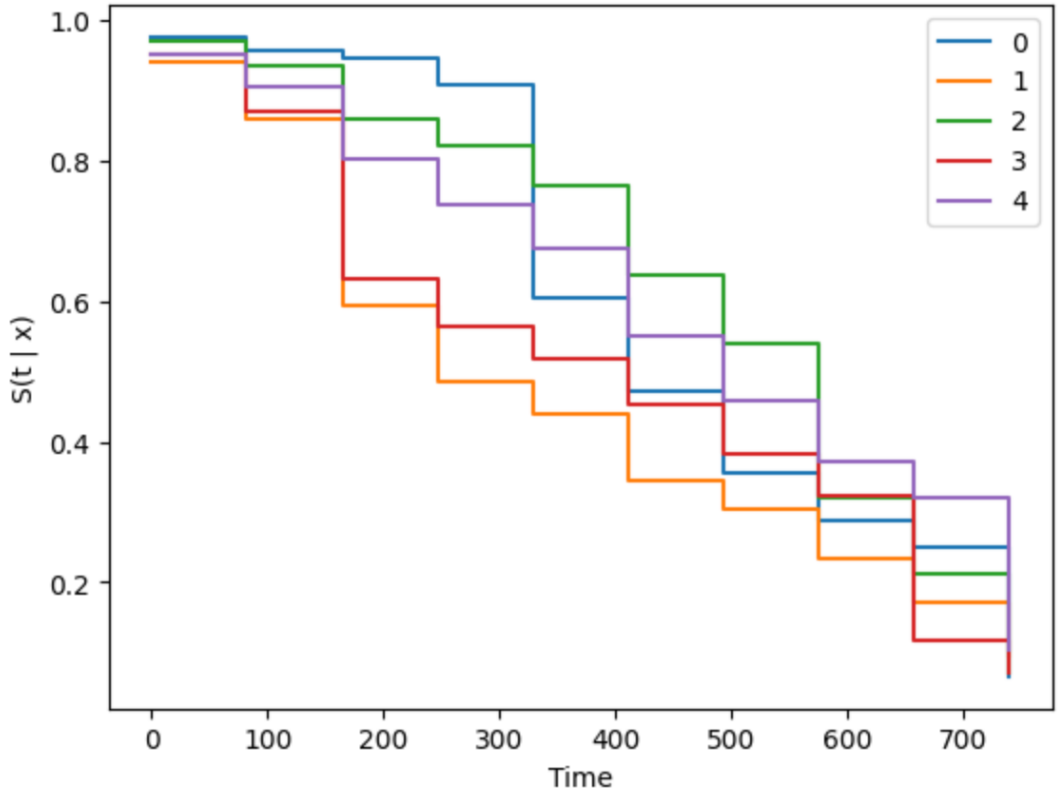
C-index (on test data) = 0.87; IBS (on test data) = 0.10

DeepHit Result

Training



Prediction



C-index (on test data) = 0.79; IBS (on test data) = 0.17

Part I

Case Study 1: Prediction of Progression of AMD (Age-related Macular Degeneration)

Age-related Macular Degeneration (AMD)

HEALTHY EYE CONDITION

EARLY AMD

LATE AMD

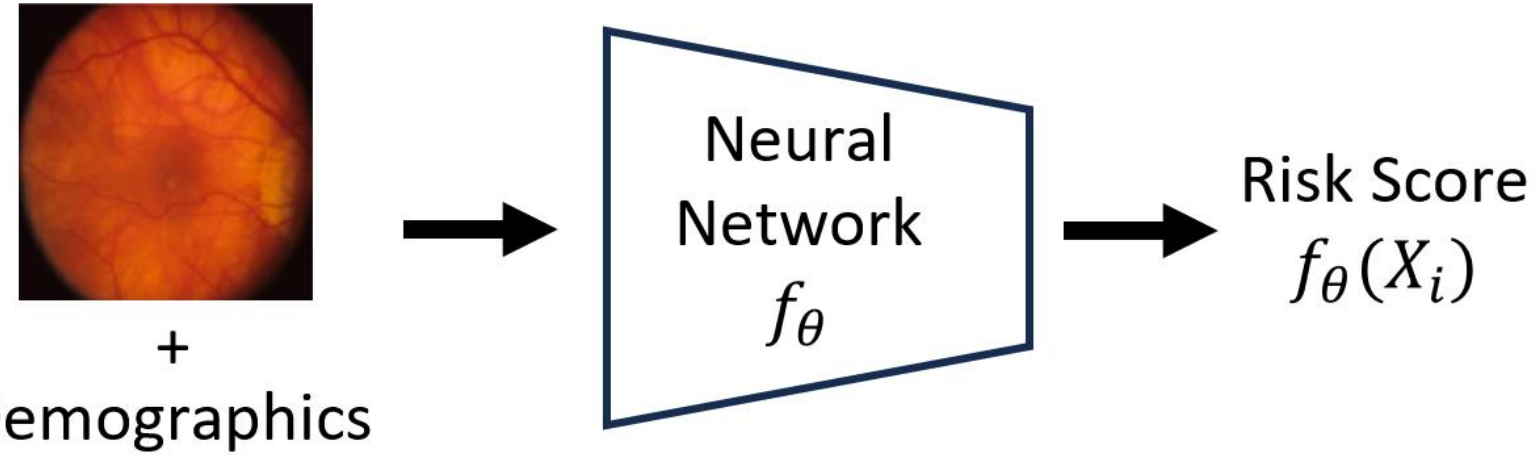
- AMD, a prognostic chronic disease, is a leading cause of vision loss in the developed world.
- The late phase (late-AMD) is associated with severe visual loss.
- Predicting the progression of AMD provides the best chance of preserving the vision.

Outcome: Time-to-late-AMD
Target: Predict AMD progression with fundus images

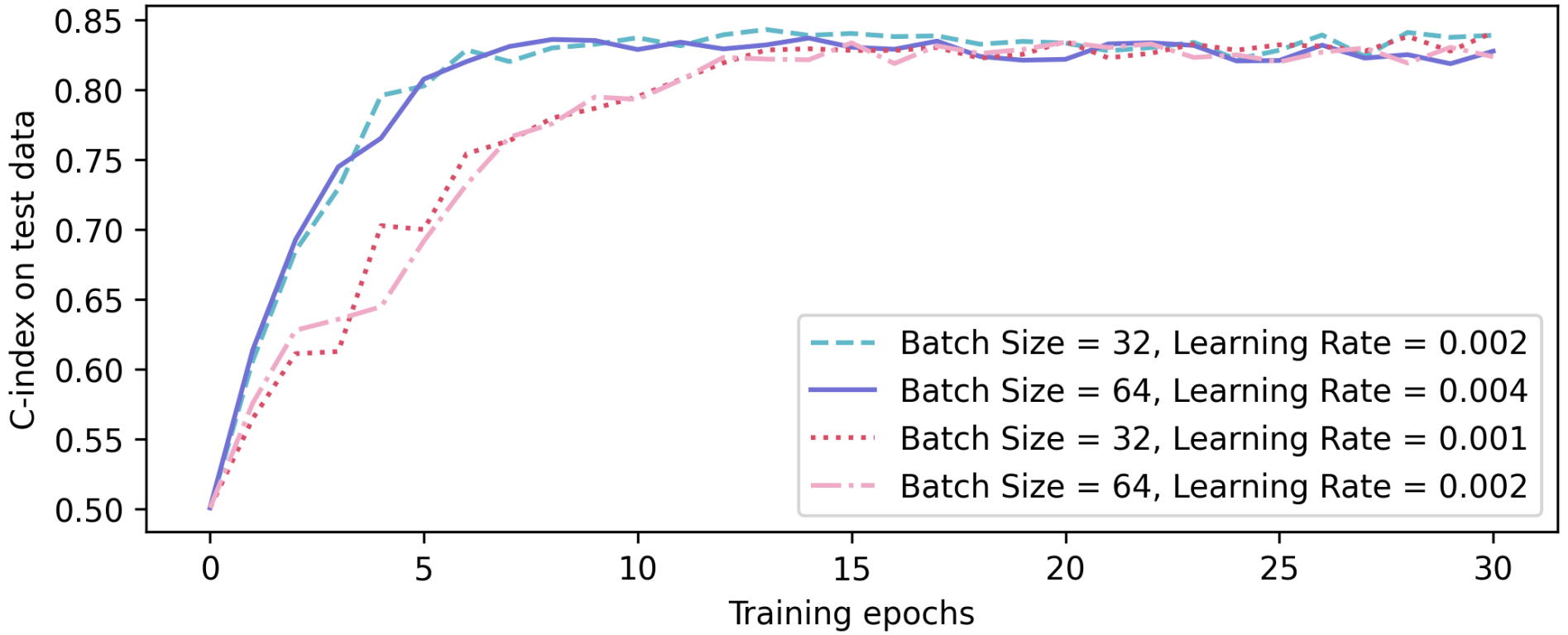
Survival Prediction with High-dimensional Covariates

- Medical Image Predictor: fundus images ($3 \times 224 \times 224$)
- Outcome of interest: Time-to-late-AMD

Cox Neural Network: $\lambda(t|X) = \lambda_0(t)\exp\{f_\theta(X)\}$

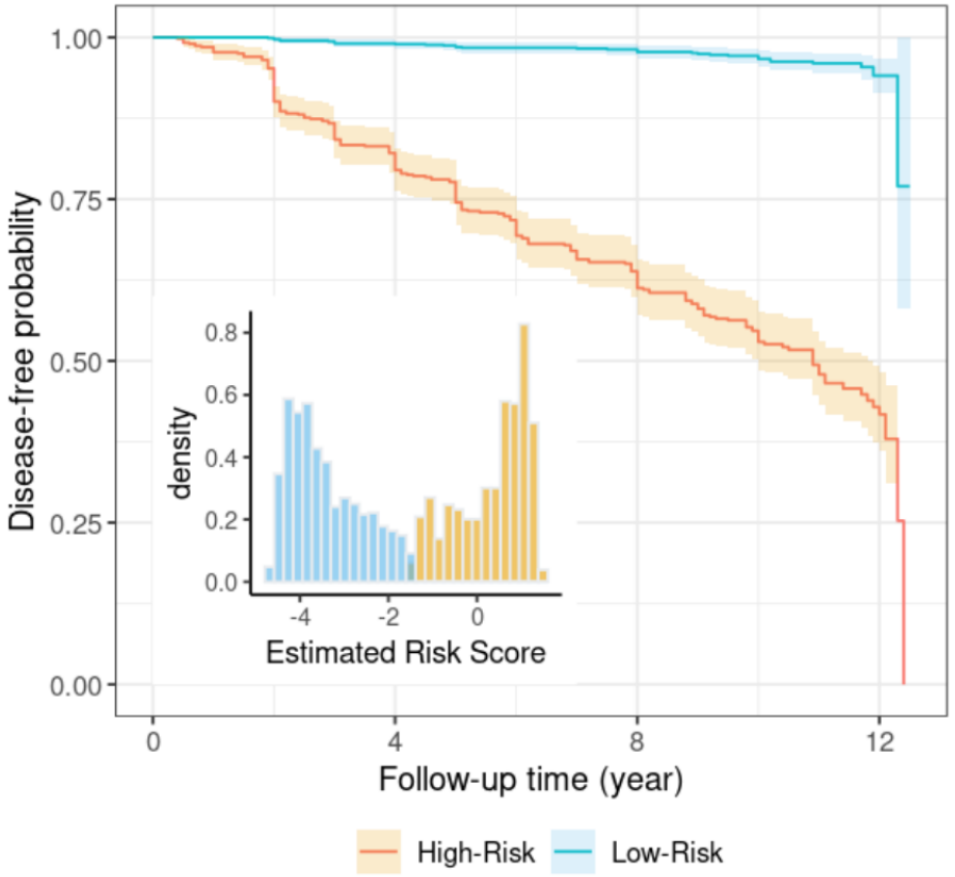


Training of NN

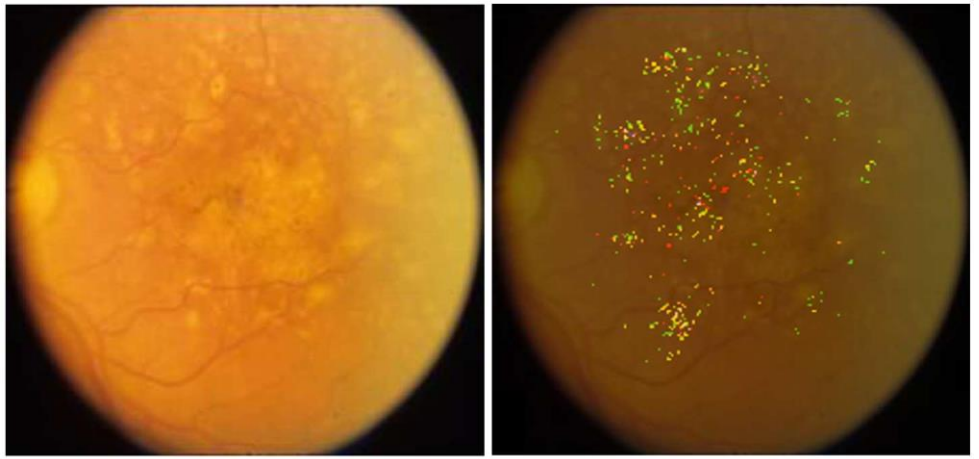


- A final model was fitted through hyper-parameter tuning with C-index 0.85 on a hold-out test data.

Subgroup Identification and Saliency Map



High-Risk				
At Risk	607	465	271	38
Events	0	122	211	261
Low-Risk				
At Risk	975	925	723	122
Events	0	10	20	33



Original Image **Saliency Map**

Prediction Risk Subgroup Characteristics

Table 2: Characteristics of high-risk and low-risk groups in test data

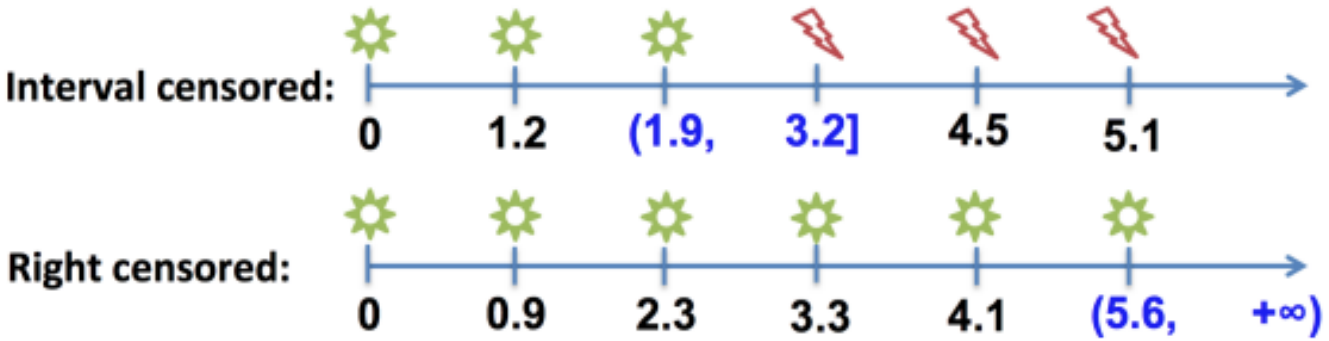
Eye-level variables Baseline manually extracted image features	High-Risk (N = 607)	Low-Risk (N = 975)	p-value
Maximum Drusen Size (mean ± s.d.)	3.83 ± 1.18	2.41 ± 1.02	<0.001
Pigmentary Abnormalities (N, %)	161 (26.5)	64 (6.6)	<0.001
Soft Drusen (N, %)	502 (82.7)	308 (31.6)	<0.001
Calcified Drusen (N, %)	18 (3.0)	0 (0)	<0.001
Reticular Drusen (N, %)	21 (3.5)	2 (0.2)	<0.001
Drusen area (mean ± s.d.)	4.39 ± 2.25	1.36 ± 1.36	<0.001
Increased Pigment (N, %)	311 (51.2)	108 (11.1)	<0.001
Subject-level variable Exclude subjects with two eye of disparate risk	High-Risk (N = 283)	Low-Risk (N = 440)	p-value
Baseline age, year (mean ± s.d.)	70.5 ± 5.6	68.4 ± 4.4	<0.001
Female (N, %)	150 (53.0)	261 (59.3)	0.110
Educational level at least highschool (N, %)	167 (59.0)	305 (69.3)	0.006
Baseline smoking status (N, %)			0.516
Never smoked	114 (40.3)	196 (44.5)	
Former smoker	146 (51.6)	209 (47.5)	
Current smoker	23 (8.1)	35 (8.0)	

* Two-sample t-test for continuous features, Pearson's Chi-squared test for categorical features

Part I

1.2 Neural networks for interval-censored survival data

Interval-Censored Data



Observed data: $D_i = (L_i, R_i; X_i)$

$L_i < T_i \leq R_i$; when $R_i = \infty$, right censoring

Likelihood: $\Pr(L_i < T_i \leq R_i) = S(L_i|X_i) - S(R_i|X_i)$

Modeling $S(t|X)$

Semiparametric transformation model:

$$S(t|X) = \exp(-G\{\exp(X^T\beta)\,\Lambda(\mathrm{t})\})$$

$G(\cdot)$ is an increasing transformation function

- $G(x) = x$: proportional hazards (PH) model
- $G(x) = \log(1 + x)$: proportional odds (PO) model
- In practice, transformation can be assumed unknown and get 'estimated' by the data

$\Lambda(\cdot)$ is an unspecified function

- Can be estimated through a sieve maximum likelihood estimation approach

Sieve Estimation

- Build a sieve space using Bernstein polynomials
- Approximate $\Lambda(\cdot)$ by a sieve estimator $\Lambda_n(\cdot)$

$$\Lambda_n(t) = \sum_{k=0}^{m_n} \phi_k B_k(t, m_n, c, u),$$

- Obtain sieve MLEs $(\hat{\beta}_n, \hat{\Lambda}_n)$

Loss Function

- The loss function (based on the full log-likelihood): $-l(\theta, \Lambda; X) + \lambda \left|\left|\theta\right|\right|_1$

$$l(\theta, \Lambda; X) = \frac{1}{n} \sum_{\{i=1\}}^{n} \log[\exp\{-\Lambda(L_i) e^{g(X_i;\theta)}\} - \exp\{-\Lambda(R_i) e^{g(X_i;\theta)}\}]$$

- The goal is to estimate $\Lambda(\cdot)$ and $g(\cdot; \theta)$ through neural networks.

BPNet-IC: two sub NNs

- For the estimation of $g(X_i;\theta)$, use a standard feed-forward neural network
- For the estimation of $\big(\Lambda(L_i),\Lambda(R_i)\big)$, use the neural network constructed by Bernstein polynomials (BPNet)

$$\Lambda_n(t) = f^{out}\Big(\sum_{k=0}^{m_n} w_k f_k(t)\Big) = \sum_{k=0}^{m_n} w_k f_k(t) = \sum_{k=0}^{m_n} w_k B_k(t, m_n, c, u),$$

- t: input nodes
- one hidden layer with $(m_n + 1)$ nodes with $f_k = B_k(t, m_n, c, u)$ as hidden node activation function
- Simultaneously solve the two NNs using one loss function

ODE-NN-IC

- The previous method of BPNet-IC depends on the PH assumption
- Relax PH by directly modeling $\Lambda(t|X)$
- Full log-likelihood:

$$\sum_{\{i=1\}}^{n} \log[\exp\{-\Lambda(L_i|X_i)\} - \exp\{-\Lambda(R_i|X_i)\}]$$

ODE-NN-IC: ODE+NN

$$\lambda(t|X_i) = f_\theta(t, X_i, \Lambda(t|X_i))$$

- The NN takes $\Lambda(t|X_i)$ as its input
- Note the relationship: $\Lambda(t|X_i)$ is the solution of the following ODE:

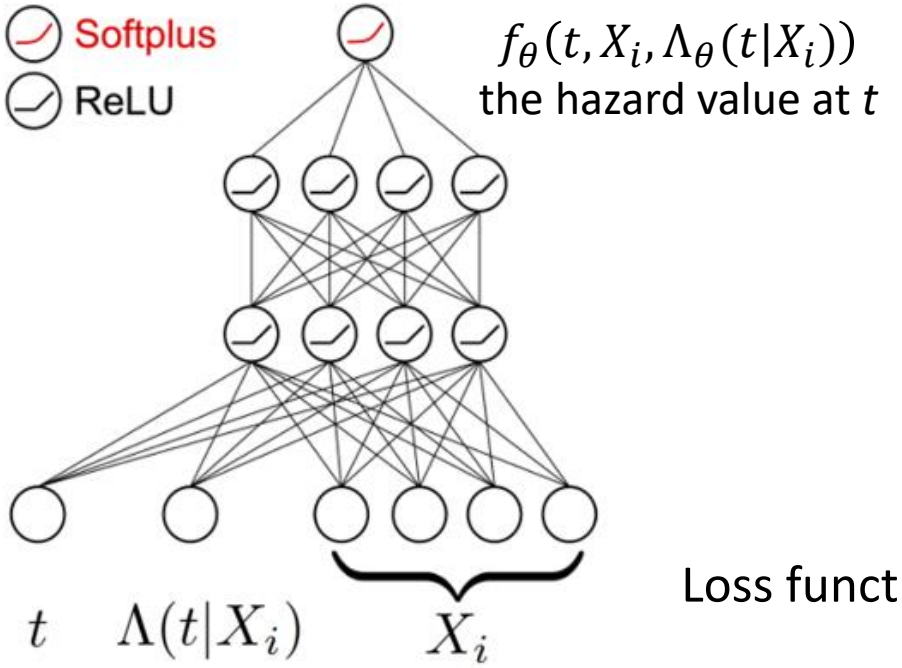
$$\begin{cases} \frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t|X_i, \theta) = f_\theta(t, X_i, \Lambda(t|X_i)) \\ \Lambda(0|X_i, \theta) = 0 \end{cases}$$

- Iteratively solve the ODE, get both $\Lambda(t|X_i)$ and $\lambda(t|X_i)$, and fit the neural network

ODE-NN-IC: NN part

- The full log-likelihood in IC without PH:

$$\sum_{\{i=1\}}^{n} \log[\exp\{-\Lambda_\theta(L_i|X_i)\} - \exp\{-\Lambda_\theta(R_i|X_i)\}]$$



$f_\theta(t, X_i, \Lambda_\theta(t|X_i))$
the hazard value at t

Model $\lambda(t|X_i)$ through NN

- Observed data: (L, R, X)
- Input nodes of NN: x, t, $\Lambda(t|X_i)$
- Output node of NN: $\lambda_\theta(x)$
- Parameters to be optimized: θ

Loss function: $-\sum_i \log[\exp\{-\Lambda(L_i|X_i)\} - \exp\{-\Lambda(R_i|X_i)\}] + \alpha||\theta||_1$

Example with Codes – ODE-NN-IC

Simulated IC dataset with 20 covariates

Dataset: "ODE_NN_IC_Simu_trainData" (training) and "ODE_NN_IC_Simu_testData" (test set)

Consists of 1000 subjects in both training set and test set

Predictors:

- 4 continuous, 4 binary, 12 multinomial
- Non-PH for one binary variable; nonlinear effects for selected continuous covariates

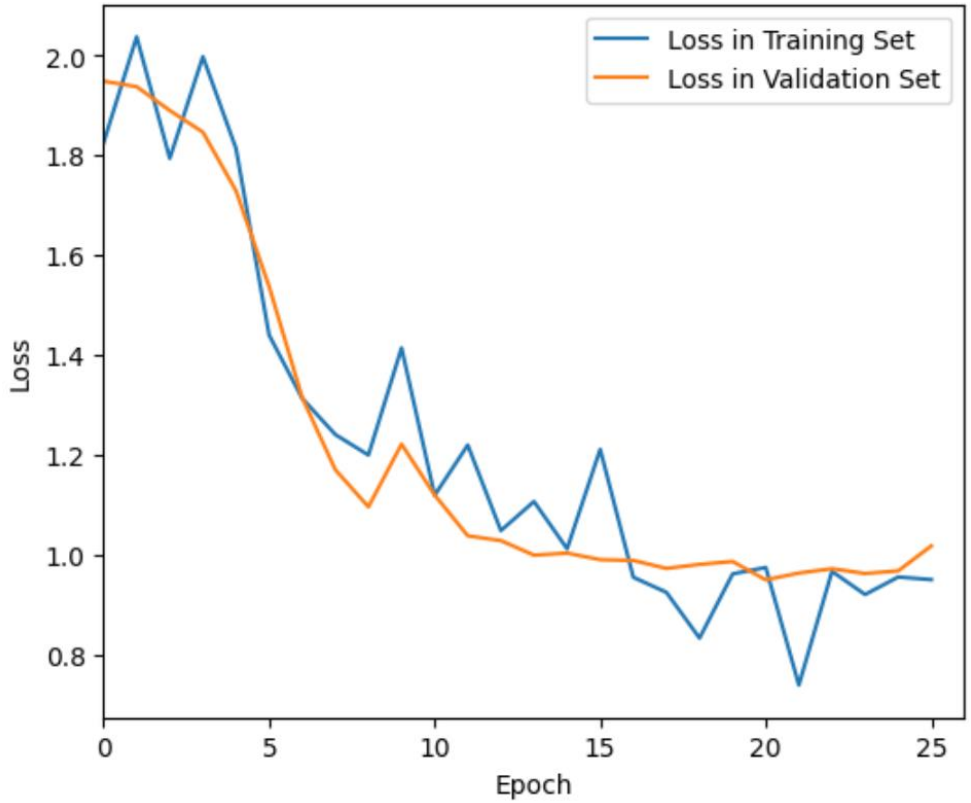
Outcome of interest: event time generate from following:

$$S(t|X_i) = \left[e^{-0.01t^{10}} I(X_{i5} = 0) + e^{-0.01t^{5}} I(X_{i5} = 1)\right]^{\exp(\sum_j \beta_j X_{ij} + X_{i1}^2 + X_{i2}^2 + X_{i3}X_{i4})}$$

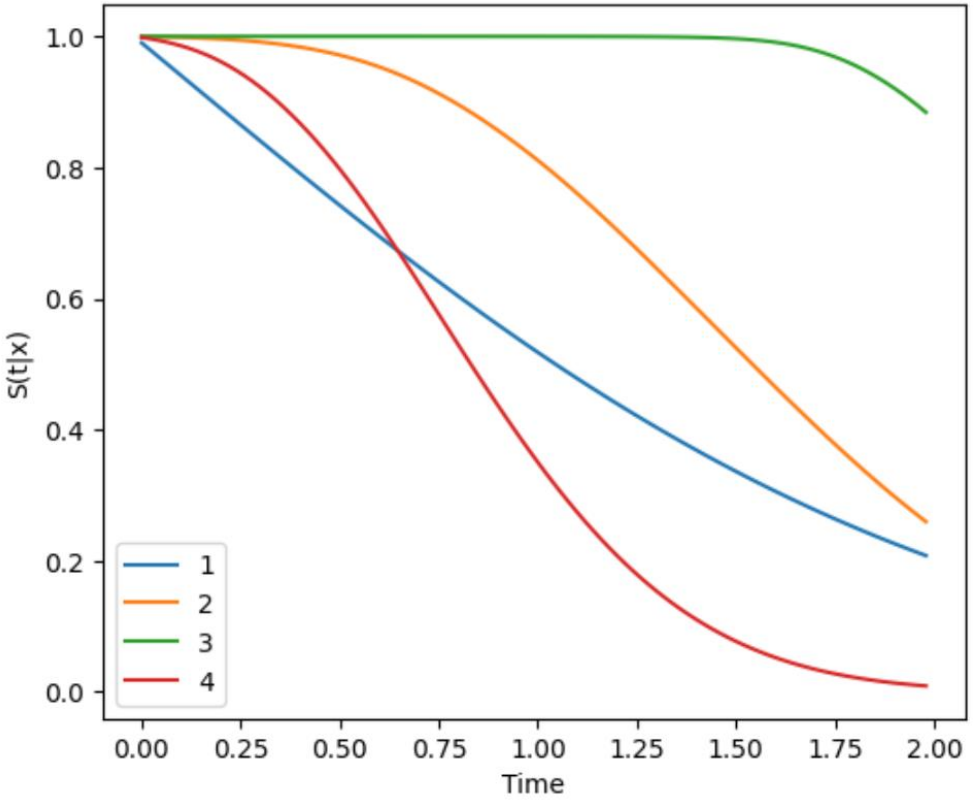
"ODE_NN_IC_Simu.ipynb"

ODE-NN-IC Result

Training



Prediction



MSE (on test data) = 0.033

Part I

Case study 2: Prediction of Development of AD (Alzheimer's Disease)

Tao Sun, Ying Ding. Neural network on interval-censored data with application to the prediction of Alzheimer's disease. (2022). *Biometrics*. doi: 10.1111/biom.13734. PMID: 35960189

Alzheimer's Disease Neuroimaging Initiative (ADNI)

- Estimated heritability of AD is up to 79%
- Desirable to make early predictions for AD using genetic data
- ADNI: longitudinal study for AD since 2004
  - investigating clinical, imaging, and genetic biomarkers for AD
  - four consecutive phases (ADNI1, ADNIGO, ADNI2, ADNI3)
  - 2,300 individuals recruited with CN, MCI, or AD
- Analysis plan
  - 1,740 Caucasian individuals with genetic data
  - event of interest: time-to-AD
  - start time: age 55
  - AD rarely occurs before 55; ADNI recruited subjects with age ≥ 55

ADNI Data

- Attribute 1: time-to-AD is subject to general interval censoring
  - interval-censoring if progressed to AD during study
  - right-censoring if free of AD at the last assessment
  - left-censoring if enrolled with AD

- Attribute 2: time-to-AD is subject to left truncation in ADNIGO
  - subjects with AD are not recruited in ADNIG

- Attribute 3: high-dimensional GWAS data
  - 7.7 million SNPs

Joint Likelihood

Likelihood function for all ADNI phases

C: conditional F: full (unconditional)

$$L_n(\beta, \Lambda | D_1, D_2) = L^C_{n_1}(\beta, \Lambda | D_1) \times L^F_{n_2}(\beta, \Lambda | D_2),$$

where

$$L^C_{n_1}(\beta, \Lambda | D_1) = \prod_{i=1}^{n_1} \frac{\exp[-G\{\exp(\widetilde{Z}_i^T \beta)\Lambda(\widetilde{L}_i)\}] - \exp[-G\{\exp(\widetilde{Z}_i^T \beta)\Lambda(\widetilde{R}_i)\}]}{\exp[-G\{\exp(\widetilde{Z}_i^T \beta)\Lambda(\widetilde{A}_i)\}]},$$

A_i: truncation time

$$L^F_{n_2}(\beta, \Lambda | D_2) = \prod_{i=1}^{n_2} (\exp[-G\{\exp(Z_i^T \beta)\Lambda(L_i)\}] - \exp[-G\{\exp(Z_i^T \beta)\Lambda(R_i)\}]).$$

Analysis of ADNI Data

- Randomly split ADNI into Data1 and Data2 by a ratio 3:1
  - Data1 for GWAS, training, and internal validation
  - Data2 for external validation

Table: Characteristics of the complete ADNI data, Data1, and Data2.

	Complete (n=1740)	Data1 (n=1305)	Data2 (n=435)
Age			
Mean(SD)	73.5 (7.1)	73.4 (7.2)	74.0 (6.9)
Median (range)	73.5 (55.0-91.4)	73.5 (55.0-91.4)	73.5 (55.0-90.3)
Gender (n, %)			
Female	780 (44.8%)	587 (45.0%)	193 (44.4%)
Male	960 (55.2%)	718 (55.0%)	242 (55.6%)
Education			
Mean(SD)	16.1 (2.8)	16.1 (2.8)	15.9 (2.7)
Median (range)	16 (4-20)	16 (4-20)	16 (7-20)
APOE (n, %)			
0 allele	941 (54.1%)	697 (53.4%)	244 (56.1%)
1 allele	637 (36.6%)	487 (37.3%)	150 (34.5%)
2 alleles	162 (9.3%)	121 (9.3%)	41 (9.4%)
Censoring types (n, %)			
Left-censored	300 (17.2%)	222 (17.0%)	78 (17.9%)
Interval-censored	342 (19.7%)	257 (19.7%)	85 (19.6%)
Right-censored	1098 (63.1%)	826 (63.3%)	272 (62.5%)
Study (n, %)			
ADNI1	703 (40.4%)	537 (41.2%)	166 (38.2%)
ADNIGO	114 (6.5%)	84 (6.4%)	30 (6.9%)
ADNI2	619 (35.6%)	462 (35.4%)	157 (36.1%)
ADNI3	304 (17.5%)	222 (17.0%)	82 (18.8%)
Follow-up years			
Mean(SD)	3.1 (3.2)	3.1 (3.2)	3.0 (3.3)
Median (range)	2.0 (0.0-15.0)	2.0 (0.0-14.7)	2.0 (0.0-15.0)

GWAS Findings on Time-to-AD

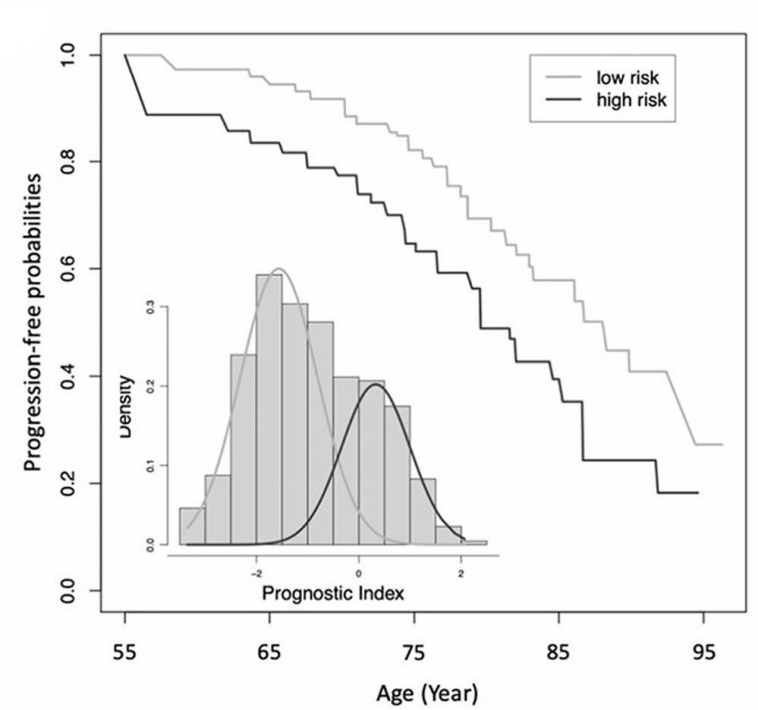
- Perform two sets of genome-wide score tests w/ wt APOE
  - PVRL2–TOMM40–APOE–APOC1 (chr19) and CHRNA4 (chr20) reach the significance level ($p < 5 \times 10^{-8}$)
  - Novel discovery: CDKN2AIP (chr4) by both GWAS ($p < 1 \times 10^{-5}$)
- Representative SNPs as predictors: 71, 371, and 623 SNPs at the p-value thresholds of 1×10^{-5}, 1×10^{-4}, and 2×10^{-4}

Prediction Performance Comparisons

Scenario	SNPs	APOE	IcenReg	BPNet-NN	ODE-NN-IC
IBS Score					
1	71	0.085 (0.011)	0.068 (0.016)	0.065 (0.002)	0.061 (0.004)
2	371	0.085 (0.011)	0.064 (0.018)	0.053 (0.006)	0.060 (0.002)
3	623	0.085 (0.011)	-	0.050 (0.006)	0.068 (0.007)
p_{out}					
1	71	0.41 (0.04)	0.32 (0.04)	0.33 (0.04)	0.35 (0.03)
2	371	0.41 (0.04)	0.44 (0.04)	0.30 (0.05)	0.32 (0.04)
3	623	0.41 (0.04)	-	0.32 (0.04)	0.32 (0.02)

Subgroup Identification

- Low- and high-risk subgroups by BPnet-NN
  - Gaussian Mixture modeling on $\hat{g}(Z_i; \hat{\theta})$



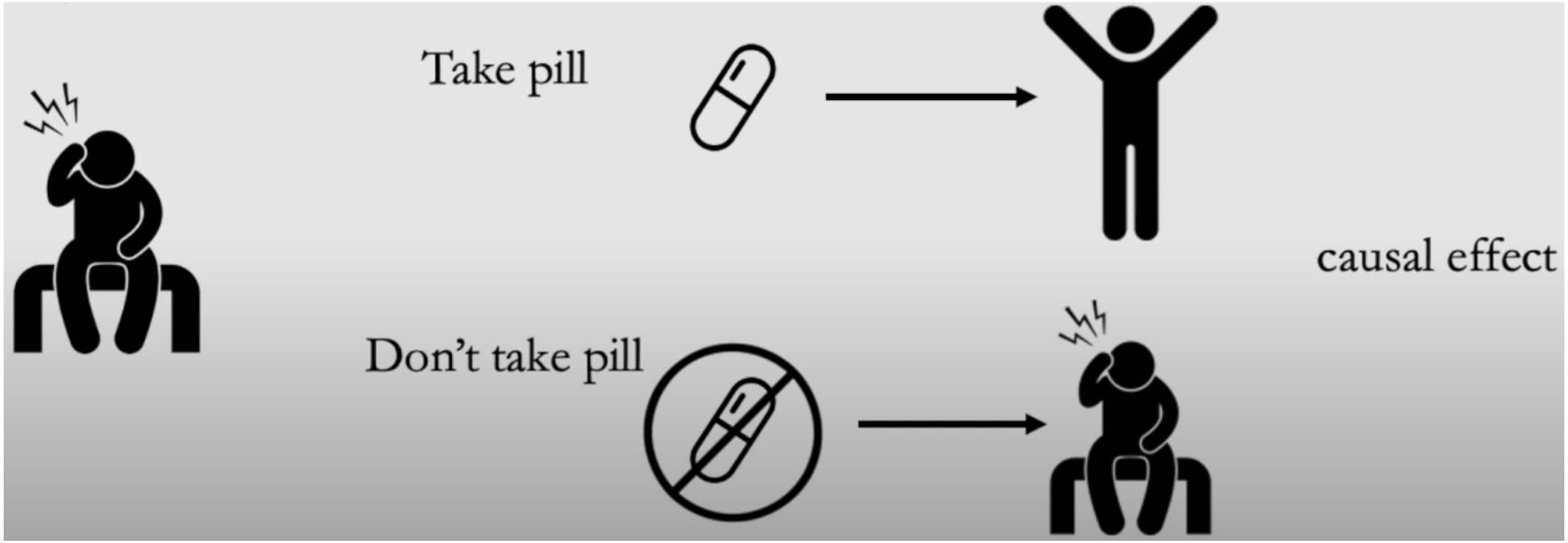
  - Helpful for early prevention and clinical management

Part II: Deep Learning for Causal Survival Analysis

2.1 CATE (conditional average treatment effect) for survival outcomes

Causal Inference

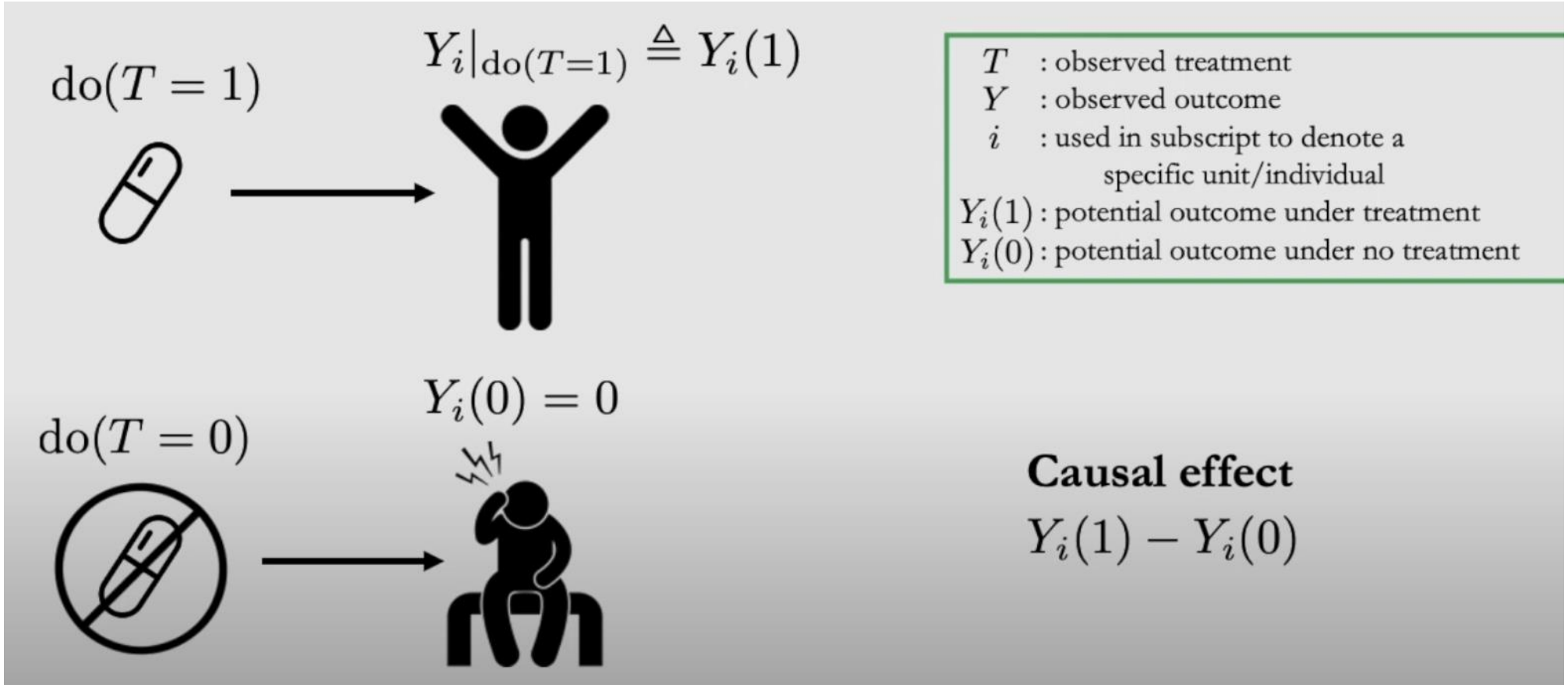
- Causal inference: inferring the effect of a treatment (policy) on some outcome



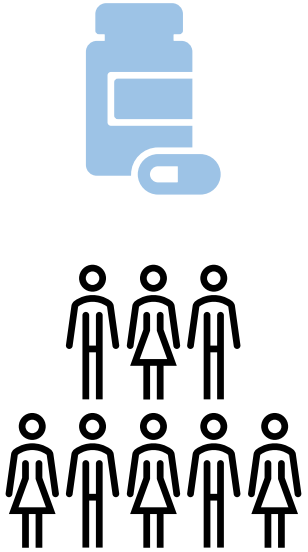
Credit to Brady Neal (Causal Inference online class)

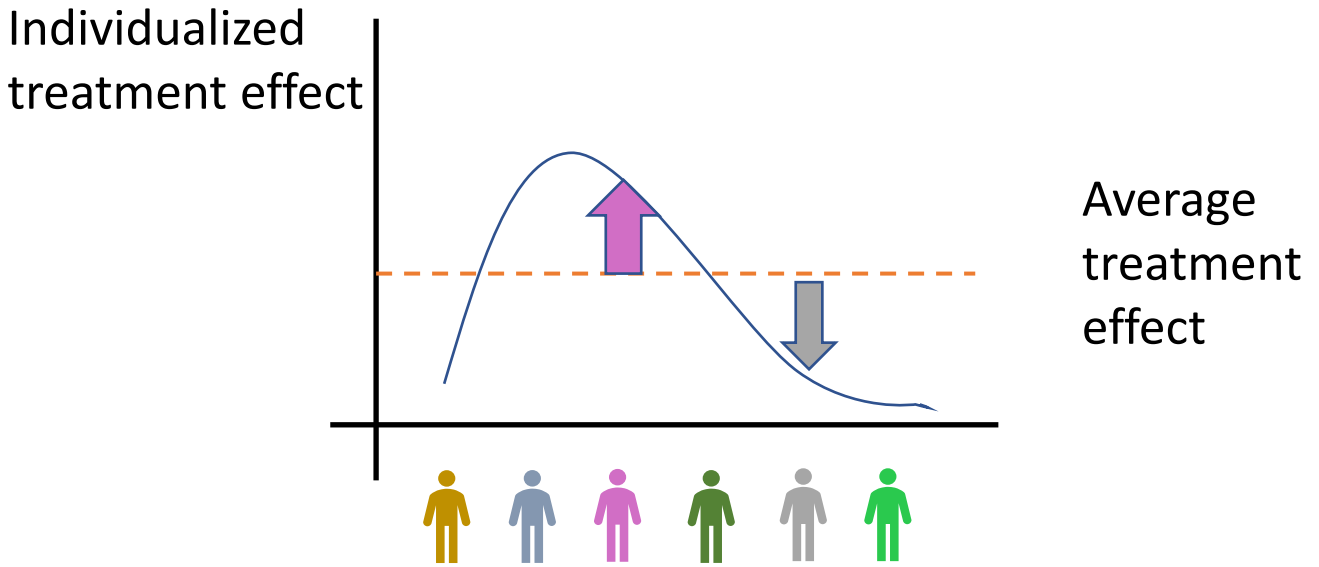
Potential Outcome Framework

- Also called counterfactual outcome framework
- ITE (Individual Treatment Effects): $\tau_i = Y_i(1) - Y_i(0)$
- CATE (Conditional Average Treatment Effects): $\tau(x) = E[Y_i(1) - Y_i(0)|X_i = x]$



Heterogeneous Treatment Effects (HTE)





Study population in a clinical trial or an observational study

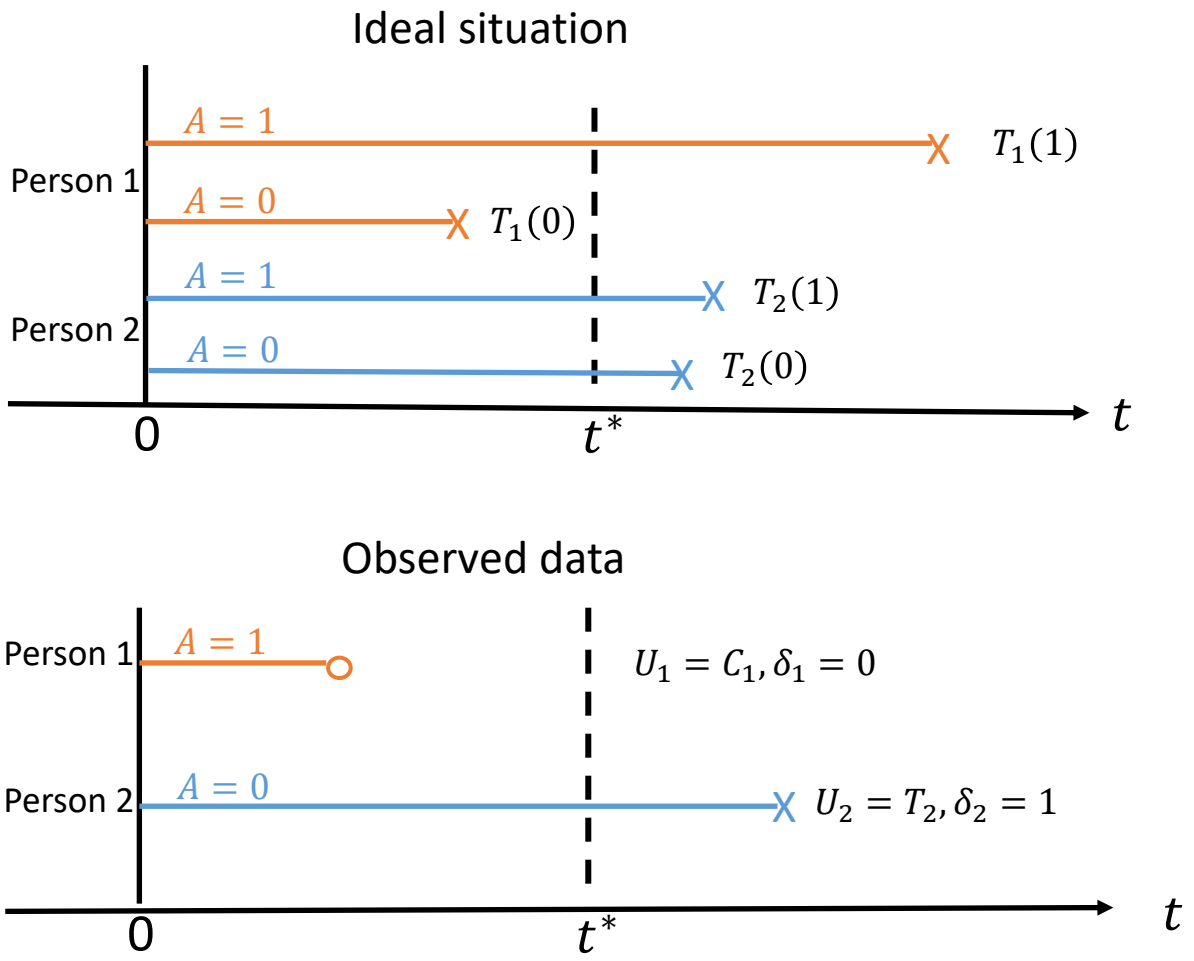
Questions of interests for **survival outcomes**:

- How much increase of the survival probability after the heart surgery?
- How much increase of the life expectancy after the chemotherapy for the lung cancer?

Notation

- A: binary treatment, $A \in \{0,1\}$
- T: survival time; $T(1), T(0)$: counterfactual survival times
- C: censoring time
- $U = \min\{T, C\}$: observed time
- $\delta = I(T < C)$: event indicator
- X: covariates
- i subscript: denote individual i

Problem Setup with Survival Outcomes



CATE Definition for Survival Outcomes

- On survival probability scale:

$$\tau(x) = \mathbb{E}[I(T(1) > t^*) - I(T(0) > t^*)|X = x]$$
$$= \mathbb{E}[I(T > t^*)|X = x, A = 1] - \mathbb{E}[I(T > t^*)|X = x, A = 0]$$

- On restricted mean survival time (RMST) scale:

$$\tau(x) = \mathbb{E}[\min(T(1) \wedge t^*) - \min(T(0) \wedge t^*)|X = x]$$
$$= \mathbb{E}[\min(T, t^*)|X = x, A = 1] - \mathbb{E}[\min(T, t^*)|X = x, A = 0]$$

Assumptions

1. Consistency: $T = AT(1) + (1 - A)T(0)$,

$$C = AC(1) + (1 - A)C(0);$$

2. Unconfoundedness: $(T(0), T(1)) \perp\!\!\!\perp A|X$, $(\mathrm{C}(0), C(1)) \perp\!\!\!\perp A|X$
3. Population Overlap: $e(\boldsymbol{x}) := P(A = 1|X = x) \in (0,1)$;
4. Non-informative censoring: $T(a) \perp C(a)|\{X, A = a\}$

Part II

2.2 Deep learning approaches for estimating CATE with survival outcomes

Two Perspectives for Estimating CATE

$$\tau(x) = \mathbb{E}[I(T(1) > t^*) - I(T(0) > t^*)|X = x]$$
$$= \mathbb{E}[I(T > t^*)|X = x, A = 1] - \mathbb{E}[I(T > t^*)|X = x, A = 0]$$
$$= S_1(t^*|x) - S_0(t^*|x)$$

Indirectly target on $\tau(x)$

Estimate individualized survival function under each treatment arm, then calculate the difference.

Deep learning can be used for learning the shared covariate space well between two treatment arms.

Directly target on $\tau(x)$

$I(T_i(1) > t^*) - I(T_i(0) > t^*)$: individualized treatment effect (ITE).

$$\mathcal{M}(I(T_i(1) > t^*) - I(T_i(0) > t^*) \sim X)$$

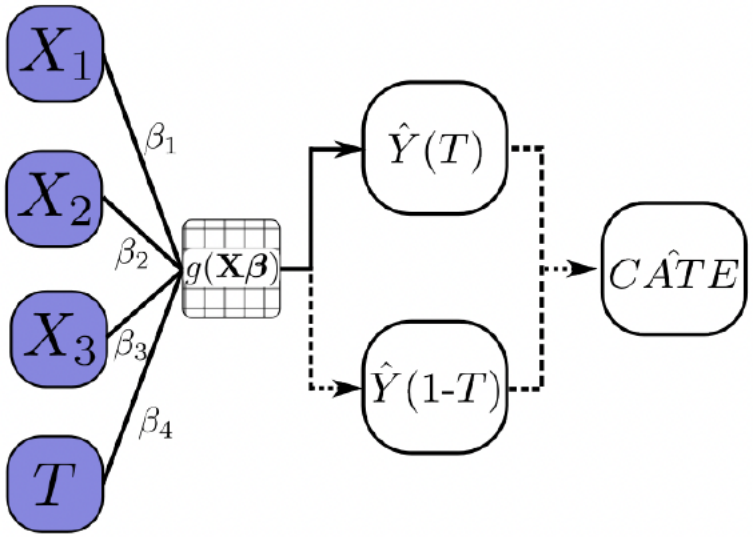
Deep learning can be used for estimating conditional survival probabilities.

(Conditional survival probabilities are nuisance parameters.)

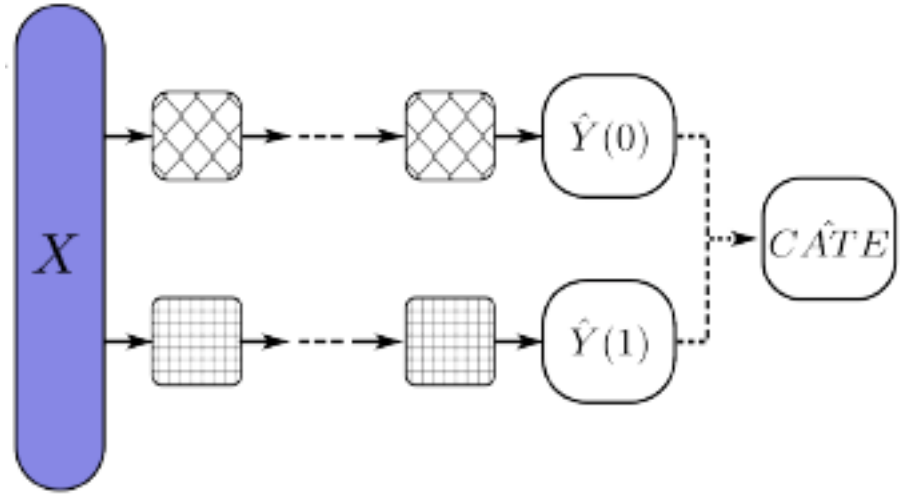
Indirectly Target on $\tau(x)$: Representation Learning

Outcome modeling approach:

- A regression network without hidden layers; corresponds to generalized linear models (GLM).



- Deep outcome modeling

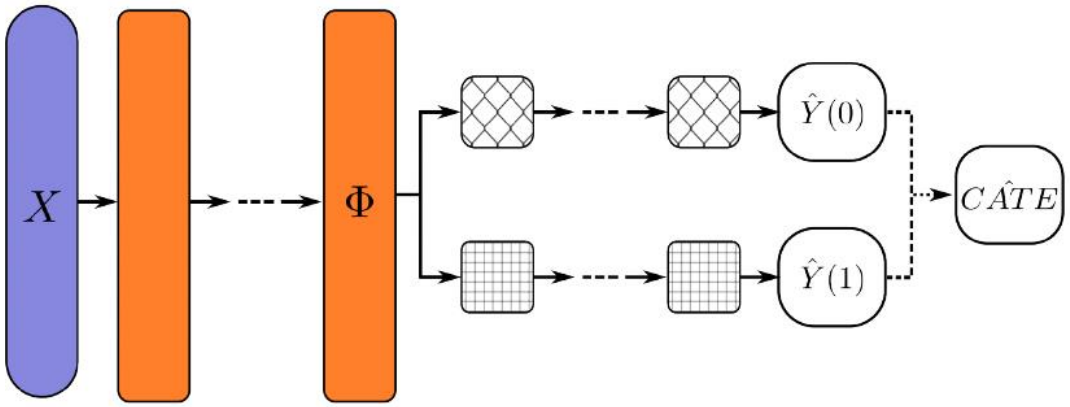


Loss function: $L\big(Y, h(X, A)\big) = MSE(A_i, h_1(X_i, 1) + (1 - A_i)(Y_i, h_0(X_i, 0)))$

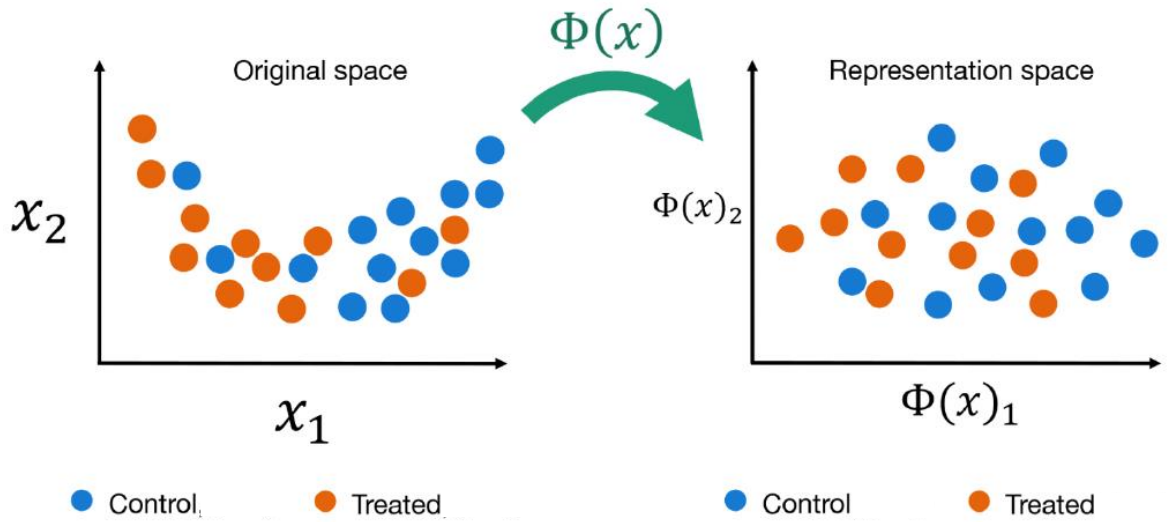
where $h_a(X, A)$: outcome regression model on treatment $A = a$

Koch, B. J., Sainburg, T., Geraldo Bastías, P., Jiang, S., Sun, Y., & Foster, J. G. (2024). A Primer on Deep Learning for Causal Inference. *Sociological Methods & Research*, *54*(2), 397-447. https://doi.org/10.1177/00491241241234866(Original work published 2025)

Representation learning approach/multi-task learning:



representation of the covariate space



$$\arg\min_{h,\Phi} \frac{1}{N}\sum_{i=1}^{N}(Y_i - (T_i(Y_i, \underbrace{h_1(\Phi(X_i),1)}_{\hat{Y}_i(1)}) + (1-T_i)(Y_i, \underbrace{h_0(\Phi(X_i),0)}_{\hat{Y}_i(0)})^2 + \lambda \underbrace{\mathcal{R}(h)}_{L_2}$$

where $h_a(X, A)$: outcome regression model on treatment $A = a$

Example of Representation Learning

- Aim to bound PEHE (Precision in Estimation of Heterogeneous Effect):

$$\epsilon_{PEHE}(f) = \int_{\mathcal{X}} (\hat{\tau}_f(x) - \tau(x))^2 \, p(x) \, dx,$$

- Factual loss and counterfactual loss:

$$\epsilon_F(h, \Phi) = \int_{\mathcal{X} \times \{0,1\}} l_{\{h,\Phi\}}(x, a) p(x, a) dx da$$

$$\epsilon_{CF}(h, \Phi) = \int_{\mathcal{X} \times \{0,1\}} l_{\{h,\Phi\}}(x, a) p(x, 1-a) dx da$$

- Bound counterfactual loss:

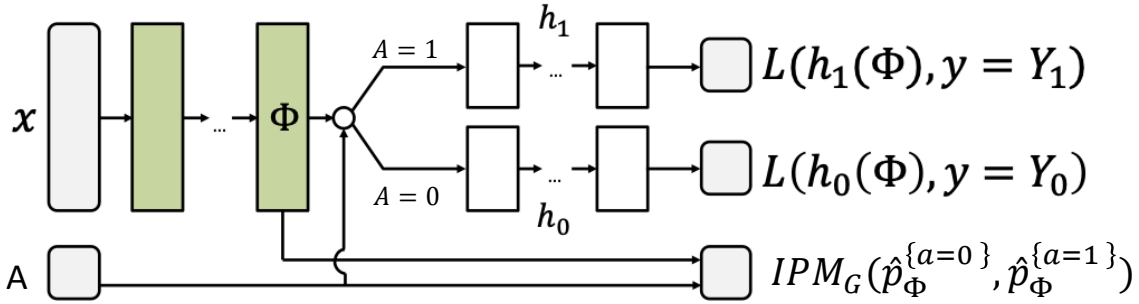
$$\epsilon_{\{CF\}(h,\Phi)} \leq (1-\mu)\epsilon_F^{\{a=1\}}(h, \Phi) + \mu\epsilon_F^{\{a=0\}}(h, \Phi) + B_\Phi \cdot IPM_G(p_\Phi^{\{a=1\}}, p_\Phi^{\{a=0\}})$$

Estimating individual treatment effect: generalization bounds and algorithms

---

Uri Shalit [*1] Fredrik D. Johansson [*2] David Sontag [2 3]

Proceedings of the 34th *International Conference on Machine Learning 2017*



$$\text{IPM}_\text{G}(p, q) := \sup_{g \in \text{G}} \left| \int_{\mathcal{S}} g(s)(p(s) - q(s)) \, ds \right|$$

Representation Learning for Survival Outcomes

- survITE (Curth et. al, 2021)

SurvITE: Learning Heterogeneous Treatment Effects from Time-to-Event Data

Alicia Curth*
University of Cambridge
amc253@cam.ac.uk

Changhee Lee*
Chung-Ang University
changheelee@cau.ac.kr

Mihaela van der Schaar
University of Cambridge
University of California, Los Angeles
The Alan Turing Institute
mv472@cam.ac.uk

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

- BITES (Schrod et. al, 2022)

Bioinformatics, 38, 2022, i60–i67
https://doi.org/10.1093/bioinformatics/btac221
ISCB/ISMB 2022

OXFORD

BITES: balanced individual treatment effect for survival data

S. Schrod[1,*], A. Schäfer[2], S. Solbrig[2], R. Lohmayer[3], W. Gronwald[4], P. J. Oefner[4], T. Beißbarth[1], R. Spang[5], H. U. Zacharias[6,7], and M. Altenbuchinger[1,*]

[1]Department of Medical Bioinformatics, University Medical Center Göttingen, Göttingen 37077, Germany, [2]Department of Physics, Institute of Theoretical Physics, University of Regensburg, Regensburg 93051, Germany, [3]Leibniz Institute for Immunotherapy, Regensburg 93053, Germany, [4]Institute of Functional Genomics, University of Regensburg, Regensburg 93053, Germany, [5]Department of Statistical Bioinformatics, Institute of Functional Genomics, University of Regensburg, Regensburg 93053, Germany, [6]Department of Internal Medicine I, University Medical Center Schleswig-Holstein, Campus Kiel, Kiel 24105, Germany and [7]Institute of Clinical Molecular Biology, Kiel University and University Medical Center Schleswig-Holstein, Campus Kiel, Kiel 24105, Germany

survITE (on discretized time)

- Discretize survival times: $T_d = \{1, \dots, T_{max}\}$
- Target parameters:
  - Hazard function: $\lambda^a(t^*|x) = P(T = t^*|T \geq t, do(A = a), X = x)$
  - Survival function: $S^a(t^*|x) = P(T > t^*|do(A = a), X = x) = \prod_{t \leq t^*}[1 - \lambda^a(t^*|x)]$
- Define counting process $N_T(t)$ and $N_C(t)$
  - $N_T(t) = I(U \leq t, \delta = 1), N_C(t) = I(U \leq t, \delta = 0)$
  - Let $N_T(0) = N_C(0) = 0$
  - $Y(t) = I(N_T(t) = 1 \cap N_T(t-1) = 0)$: an event occurs exactly at time t
- Conditional hazard function:

$$\lambda(t^*|a, x) = P(U = t^*, \delta = 1|U \geq t^*, A = a, X = x)$$
$$= P(Y(t^*) = 1|N_T(t^* - 1) = N_C(t^* - 1) = 0, A = a, X = x)$$

Three Types of Covariates Shifts

- Confounding/treatment selection bias
- Censoring bias
- Event-induced shifts

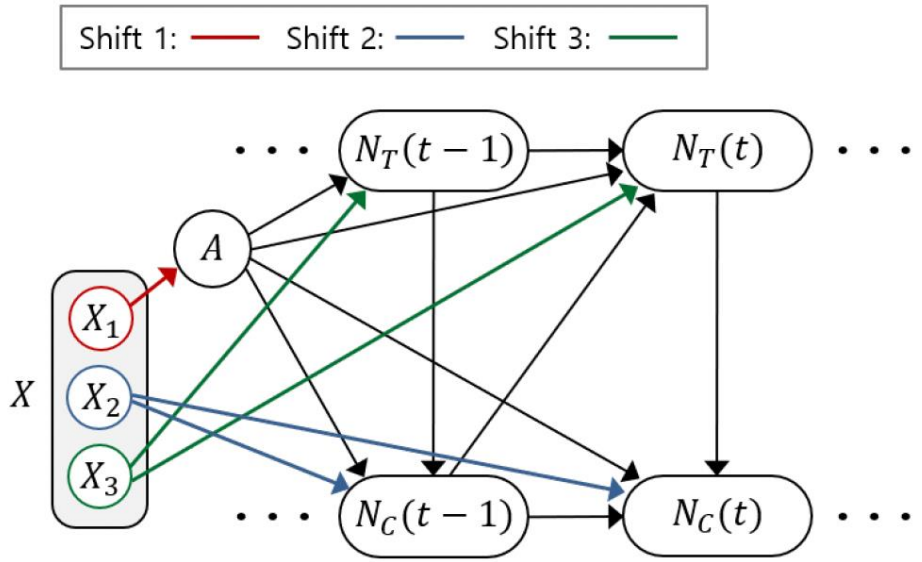


Figure 1: The assumed underlying DAG. Covariates X can be split into (possibly overlapping) subsets X_1, X_2 and X_3, determining treatment selection, informative censoring, and event times, respectively.

SurvITE Architecture

$$\mathcal{L}_{target}(\theta_\phi, \theta_h) = \mathcal{L}_{risk}(\theta_\phi, \theta_h) + \beta \mathcal{L}_{ipm}(\theta_\phi) \tag{6}$$

where $\theta_h = \{\theta_{h_{a,\tau}}\}_{a\in\{0,1\},\tau\in\mathcal{T}}$, and $\beta > 0$ is a hyper-parameter. The pseudo-code of SurvITE, the details of how to obtain $Wass(\cdot,\cdot)$ and how we set β can be found in Appendix D.

$$\mathcal{L}_{risk}(\theta_\phi, \theta_h) = \frac{1}{t_{\max}} \sum_{t=1}^{t_{\max}} \sum_{i:\tilde{\tau}_i \geq t} n_{1,t}^{-1} a_i \ell\big(y_i(t), h_{1,t}(\Phi(x_i))\big) + n_{0,t}^{-1}(1-a_i)\ell\big(y_i(t), h_{0,t}(\Phi(x_i))\big),$$

$$\mathcal{L}_{ipm}(\theta_\phi) = \sum_{a\in\{0,1\}} \sum_{t=1}^{t_{\max}} Wass\big(\{\Phi(x_i)\}_{i=1}^{n}, \{\Phi(x_i)\}_{i:\tilde{\tau}_i \geq t, a_i = a}\big),$$

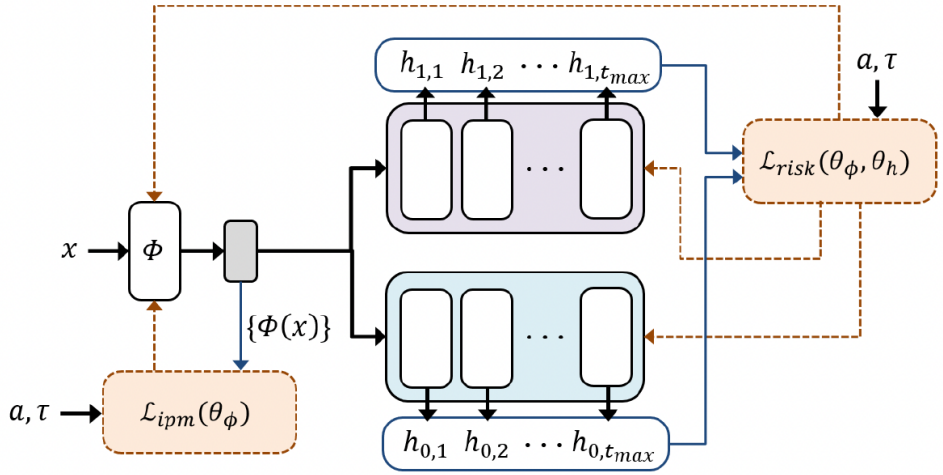
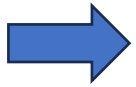


Figure 2: SurvITE architecture.

Data Format for survITE

X	A	U	δ
x_1	1	3	0
x_2	1	2	1
x_3	0	1	1
x_4	0	1	0
x_5	1	3	1
x_6	0	2	1



$\hat{\lambda}^a(1|x)$

X	A	$Y(1)$
x_1	1	0
x_2	1	0
x_3	0	1
x_4	0	0
x_5	1	0
x_6	0	0

$\hat{\lambda}^a(2|x)$

X	A	$Y(2)$
x_1	1	0
x_2	1	1
x_3	-	-
x_4	-	-
x_5	1	0
x_6	0	1

$\hat{\lambda}^a(3|x)$

X	A	$Y(3)$
x_1	1	0
x_2	-	-
x_3	-	-
x_4	-	-
x_5	1	1
x_6	-	-

Directly Target on $\tau(x)$: Meta-learners

- Two major steps
  - Step 1: Estimate nuisance parameters and construct ITEs or pseudo ITEs
  - Step 2: Regress constructed ITEs or pseudo ITEs on the covariates

- Two popular approaches for estimating CATE in continuous or binary outcomes
  - X-learner (Kunzel et. al, 2019, *PNAS*)

  - R-learner (Nie and Wager, 2021, *Biometrika*)

X-learner

- For a continuous outcome Y

Step 1:

$$\hat{h}_0 = \mathcal{M}_0(Y \sim X)$$
$$\hat{h}_1 = \mathcal{M}_1(Y \sim X)$$

← Outcome regression

$$Y_1^* = Y - \hat{h}_0(X), A = 1$$
$$Y_0^* = \hat{h}_1(X) - Y, A = 0$$

← Create Pseudo ITE

Y_a^* has ITE interpretation: Difference between the observed outcome and predicted counterfactual.

Step 2:

$$\hat{\tau}_1(x) = \mathcal{M}_1^{\tau}(Y_1^* \sim X)$$
$$\hat{\tau}_0(x) = \mathcal{M}_0^{\tau}(Y_0^* \sim X)$$

← Group-specific CATE estimator

$$\hat{\tau}(x) = e(x)\,\hat{\tau}_0(x) + [1 - e(x)]\hat{\tau}_1$$

← Final CATE estimator

- For survival outcomes (e.g., survival probability scale)

Step 1:

$$\hat{S}_0(t^*|x) = \mathcal{M}_0((U,\delta)\sim X)$$
$$\hat{S}_1(t^*|x) = \mathcal{M}_1((U,\delta)\sim X)$$

← $\mathcal{M}_a$ can be a NN-based survival model

$$Y_1^* = I(T > t^*) - \hat{S}_0(t^*|x), A = 1$$
$$Y_0^* = \hat{S}_1(t^*|x) - I(T > t^*), A = 0$$

← Pseudo ITE

Step 2:

$$\hat{\tau}_1(x) = \mathcal{M}_1^{\tau}(Y_1^*\sim X)$$
$$\hat{\tau}_0(x) = \mathcal{M}_0^{\tau}(Y_0^*\sim X)$$

$$\hat{\tau}(x) = e(x)\,\hat{\tau}_0(x) + [1 - e(x)]\hat{\tau}_1$$

- For survival outcomes (e.g., survival probability scale)

Step 1:

$$\hat{S}_0(t^*|x) = \mathcal{M}_0((U,\delta)\sim X)$$
$$\hat{S}_1(t^*|x) = \mathcal{M}_1((U,\delta)\sim X)$$

← $\mathcal{M}_a$ can be a NN-based survival model

$$Y_1^* = I(T > t^*) - \hat{S}_0(t^*|x), A = 1$$
$$Y_0^* = \hat{S}_1(t^*|x) - I(T > t^*), A = 0$$

← Pseudo ITE

$I(T > t^*)$ cannot be observed for those who censored before t^*.

Step 2:

$$\hat{\tau}_1(x) = \mathcal{M}_1^{\tau}(Y_1^*\sim X)$$
$$\hat{\tau}_0(x) = \mathcal{M}_0^{\tau}(Y_0^*\sim X)$$

$$\hat{\tau}(x) = e(x)\,\hat{\tau}_0(x) + [1 - e(x)]\hat{\tau}_1$$

$\mathcal{M}_a^{\tau}$: weighted pseudo ITE regression $\frac{1}{n_a^O}\sum_{i=1}^{n_a^O} w_i^C [Y_a^* - \tau_a(x)]2$

w_i^C: inverse probability censoring weighting $\frac{1}{P(C>\min(T,t^*)|X=x)}$, which needs to be estimated in step 1.

Bo,N.#, Wei, Y.#, Zeng, L., Kang C., Ding Y. A Meta-Learner Framework to Estimate Individualized Treatment Effects for Survival Outcomes. *Journal of Data Science* **22**(2024), no. 4, 505-523, DOI 10.6339/24-JDS1119

R-learner

- For continuous outcome Y:
  - Motivated by Robinson decomposition: $Y_i - E[Y_i|X_i = x] = [A_i - e(X_i)]\tau(X_i) + \epsilon_i$, where $E[\epsilon_i|X_i, A_i] = 0$.
  - CATE can be solved by minimizing the objective function:

$$\min_{\tau} \frac{1}{n}\sum_{i=1}^{n}\{[Y_i - h(X_i)] - [A_i - e(X_i)]\tau(X_i)\}^2$$

  - Re-arrange the objective function:

$$\min_{\tau} \frac{1}{n}\sum_{i=1}^{n}\underbrace{[A_i - e(X_i)]^2}_{w^A}\{[Y_i - h(X_i)]/[A_i - e(X_i)] - \tau(X_i)\}^2$$

  - Step 1:
    - $\hat{h}(X) = \mathcal{M}(Y \sim X)$ ← Outcome regression
    - $Y^* = \frac{[Y - \hat{h}(X)]}{[A - \hat{e}(X)]}$ ← Create Pseudo outcome
    - $w^A = [A - \hat{e}(X)]^2$ ← Method-specific weight
  - Step 2:

$$\min_{\tau} \frac{1}{n}\sum_{i=1}^{n}[A_i - \hat{e}(X_i)]^2\{[Y_i - \hat{h}(X_i)]/[A_i - \hat{e}(X_i)] - \tau(X_i)\}^2$$

- For survival outcome:

  - Step 1:
    - $\hat{S}(X) = \mathcal{M}\big((U, \delta) \sim X\big)$ ← $\mathcal{M}$ can be a NN-based survival model

      or $\hat{S}(X) = \hat{e}(X)\hat{S}_1(X) + [1 - \hat{e}(X)]\hat{S}_0(X)$

    - $Y^* = [I(T > t^*) - \hat{S}(X)]/[A - \hat{e}(X)]$
    - $w^A = [A - \hat{e}(X)]^2$

  - Step 2:

$$\min_{\tau} \frac{1}{n^o} \sum_{i=1}^{n^o} w_i^C w_i^A \{[I(T > t^*) - \hat{S}(X_i)]/[A_i - \hat{e}(X_i)] - \tau(X_i)\}^2$$

inverse probability censoring weighting $\frac{1}{P(C > \min(T, t^*) | X = x)}$

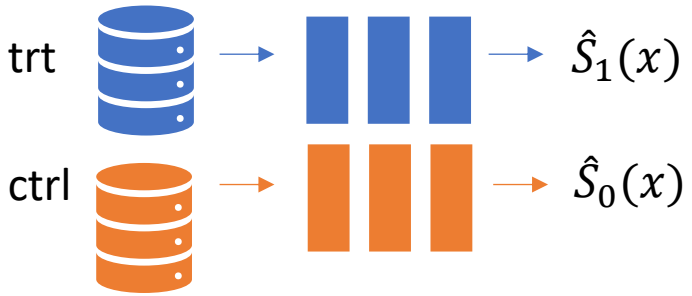
Remark: Using NN to learn conditional survival functions can yield better performance in predicting CATE.

Bo, N., Jeong, J.-H., Forno, E. and Ding, Y. (2025), Evaluating Meta-Learners to Analyze Treatment Heterogeneity in Survival Data: Application to Electronic Health Records of Pediatric Asthma Care in COVID-19 Pandemic. *Statistics in Medicine*, 44: e10333. https://doi.org/10.1002/sim.10333

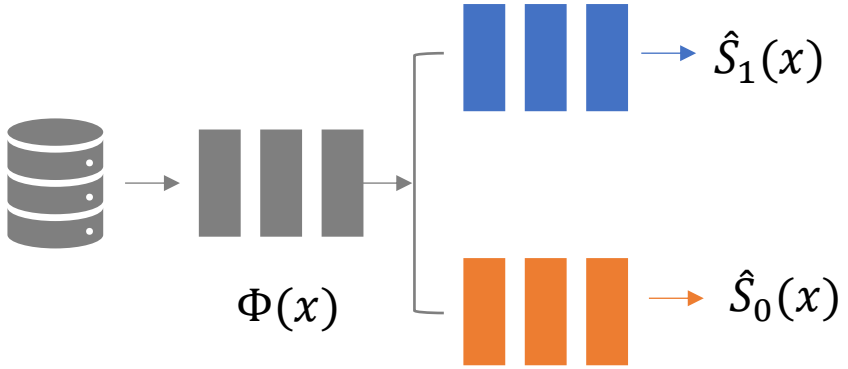
More Complicated NN Blocks

- What we did so far:

  Learning outcome regression models separately for each treatment group



- Adding shared blocks to learn representations of X:



Example Codes

- Xlearner_simu.ipynb
- Rlearner_simu.ipynb

- survITE_simu_TF2.ipynb

Part II

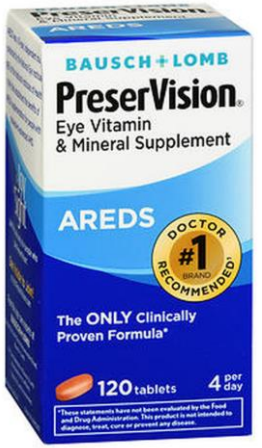
Case Study 3: Individualized Treatment Effects of AREDS formula in delaying Progression to late-AMD

https://doi.org/10.6339/24-JDS1119

Application on AREDS

AREDS:

- Treatment Arm:
  - ***Placebo***
  - Antioxidants (Vitamin C + Vitamin E + beta-carotene)
  - Zinc (zinc oxide + cupric oxide)
  - ***Antioxidants + Zinc (AREDS®)***
- ***X***: age, smoking status, sex, education, baseline AMD severity scale, **686 SNPs**
  - 46 SNPs identified to be associated with treatment efficacy (**CE4 SNPs*)***
  - 640 SNPs reported to be associated with AMD progression (**prognostic SNPs**).
- N = 806 (balanced design)
- $\tau(x; t^* = 5)$: difference of progression-free probability at 5 years

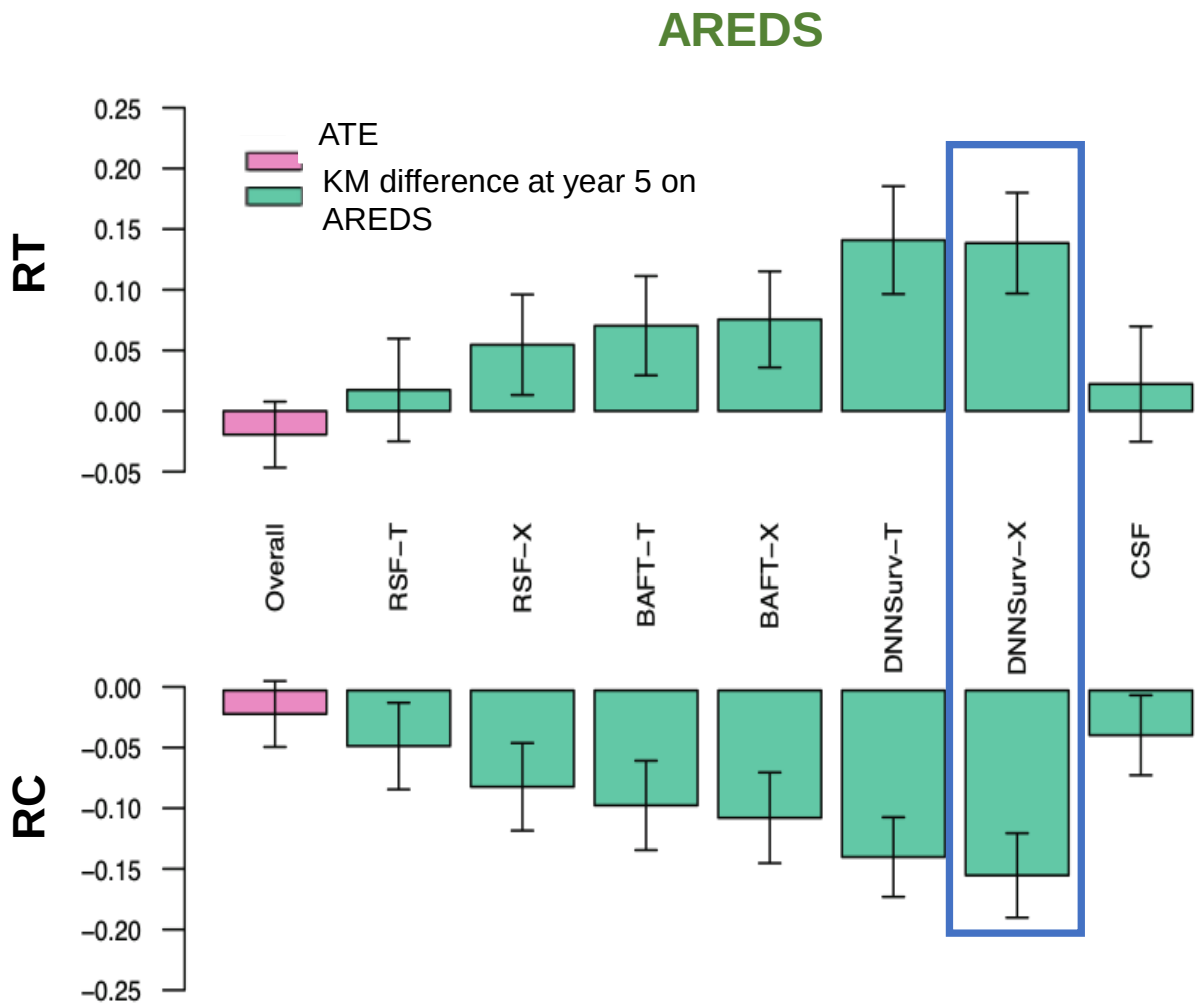


AREDS2 (external validation data):

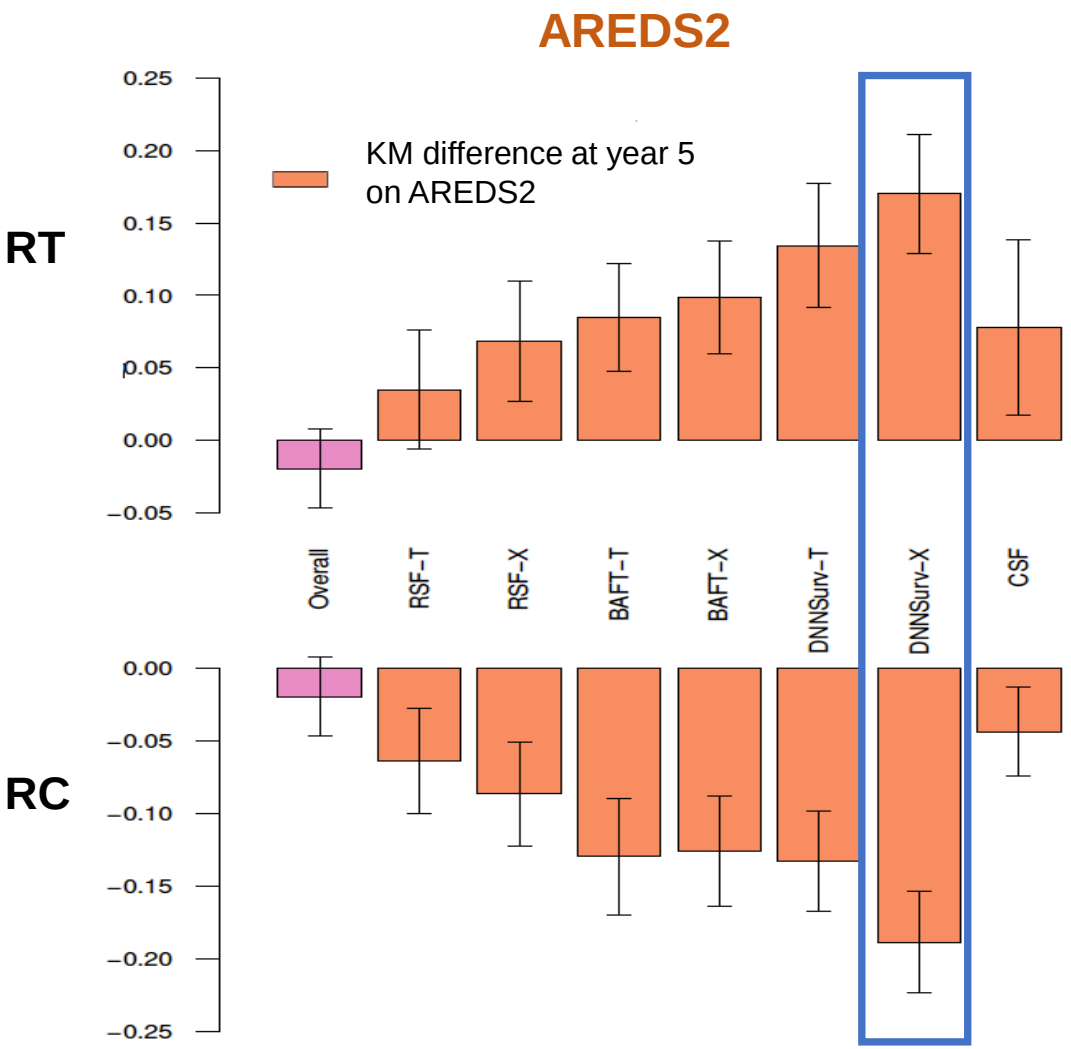
- Treatment Arm:
  - *AREDS®*
  - Modification 1 of *AREDS®*
  - Modification 2 of *AREDS®*
  - *AREDS2*
- N = 110 (AREDS arm only)

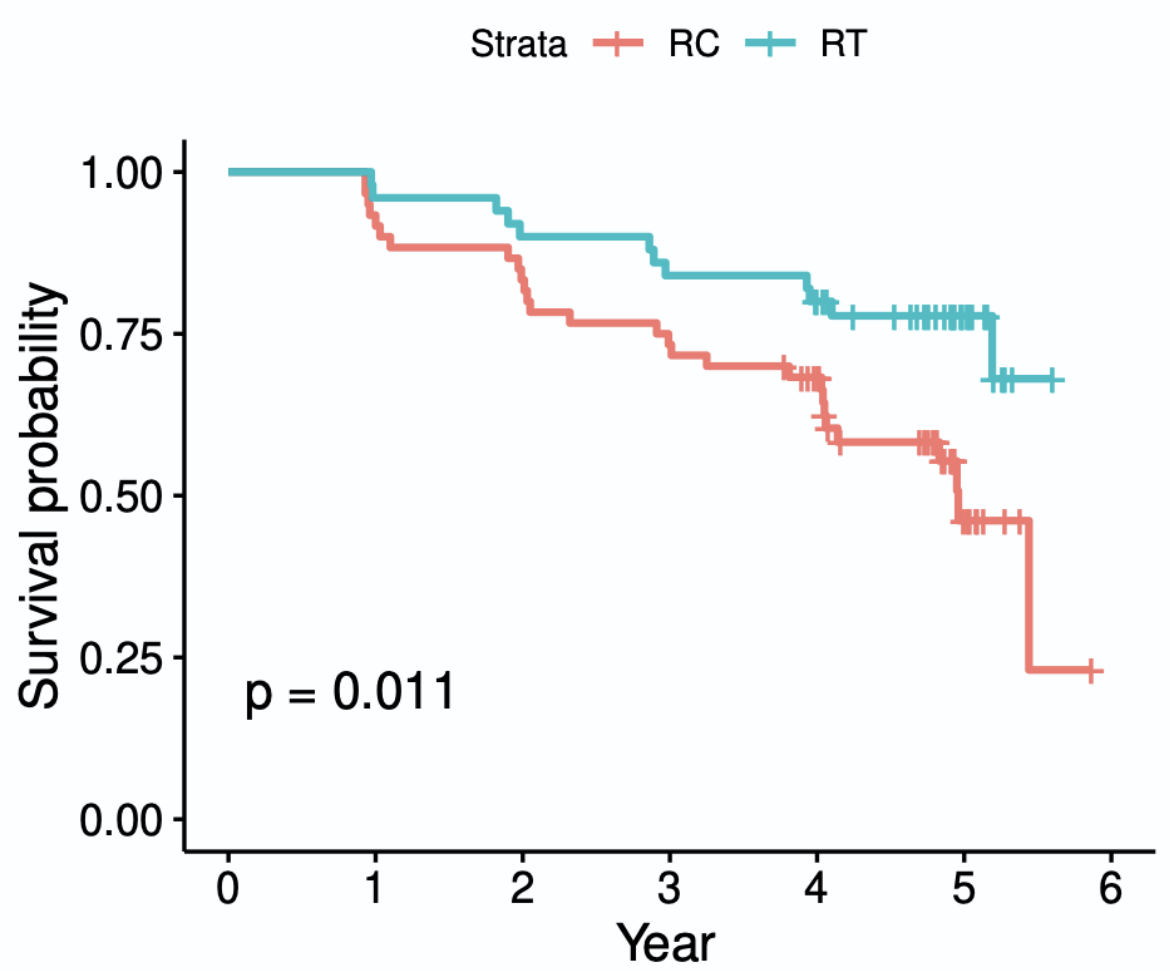
Estimated ITEs in RT and RC Groups

$\hat{\tau}(\boldsymbol{x}; t^*) > 0$: recommended for treatment (**RT**).
$\hat{\tau}(\boldsymbol{x}; t^*) \leq 0$: recommended for placebo (**RC**).
Calculate **mean treatment difference** using Kaplan Meier (KM) estimator between two arms at year 5 in RT and RC separately.



External Validation on AREDS2

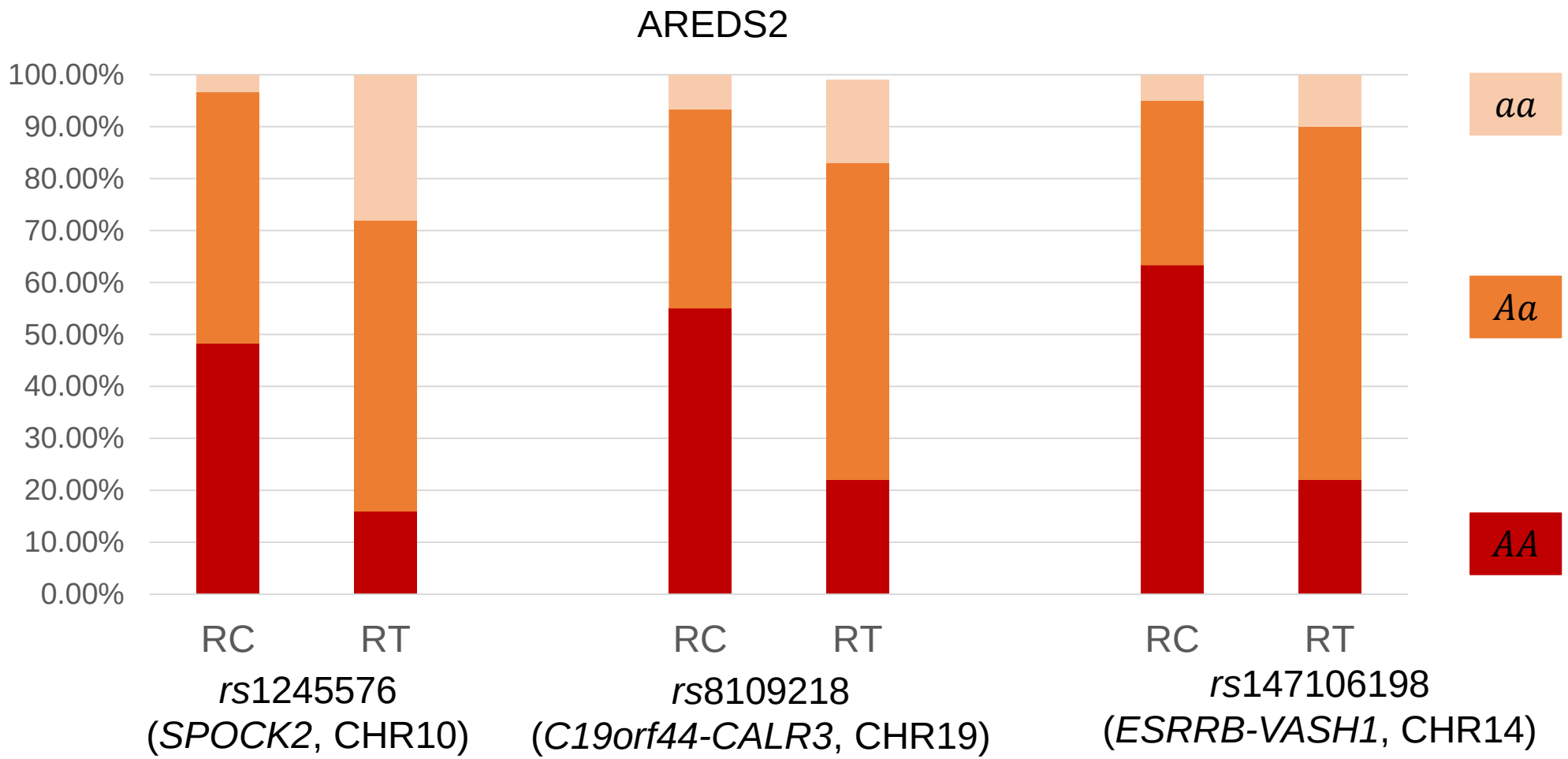




The Kaplan Meier curves of RT and RC cohorts in AREDS2 where the treatment recommendation rule is based on the D-X result on AREDS data.

Importance Features

- Applied a post-hoc Boruta algorithm after dichotomizing predicted ITEs using D-X.
- Identified 17 SNPs, mainly from CHR10, 14 and 19.



Thank You!

Email: yingding@pitt.edu
Lang.zeng@pitt.edu
nab177@pitt.edu