Survival, Quantile regression

Hongtu Zhu

The University of North Carolina at Chapel Hill

June 15, 2024

Table of contents

Overview

- Survival analysis is a significant area of interest within the statistics community.
- While it may not be as prominent as fields like computer vision and natural language processing, there has been considerable work on deep learning-based survival analysis methods.
- Most of these methods, although rooted in statistical approaches, provide greater flexibility by eliminating the need for variable selection and offering enhanced expressiveness in handling non-linear relationships.

Notations

- Let T denote the time from origin (e.g., birth) until the event we are studying.
- Because of censoring(right) we do not always observe T. Instead we observe $\{x_i, t_i, \delta_i\}$
  - x_i: Explanatory variables
  - δ_i: Whether the event was right censored($\delta = 1$ is not censored)
  - t_i: The follow-up time, equals to T_i if $\delta_i = 1$
- Survival function: $S(t) = \mathbb{P}(T > t)$
- Hazard function: $\lambda(t) = \lim_{\delta \to 0} \frac{\mathbb{P}(t \leq T \leq T + \delta | T \geq t)}{\delta}$

Statistical methods

- Non-Parametric Methods: Kaplan-Meier, Nelson-Aalen
- Semi-Parametric Methods: Cox Regression
- Parametric Methods: Accelerated Failure Time(AFT)

DeepSurv - Background

- DeepSurv [Kat+18] model is based on Cox proportion hazard model.

Cox-PH models the hazard function $\lambda(t)$ given covariates X:

$$\lambda(t \mid X) = \lambda_0(t) \exp(\beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p),$$

where β is the vector of regression coefficients. Cox-PH model estimate β by maximizing the partial log-likelihood,

$$\ell(\beta) = \sum_{i:\delta_i=1} \left[\beta' X_i - \log \left(\sum_{j \in R(t_i)} \exp(\beta' X_j) \right) \right].$$

$R(t_i)$ is the risk set at time t_i, which includes all individuals who are at risk before time t_i.

DeepSurv - model

The DeepSurv model replaces the linear estimator $\beta' X_i$ with a feedforward neural network, denoted as $h(X_i)$. The parameters of this neural network are optimized by minimizing the negative partial log-likelihood.

$$-\ell(\beta) = -\sum_{i:\delta_i=1} \left[h(X_i) - \log \left(\sum_{j \in R(t_i)} \exp(h(X_j)) \right) \right].$$

DeepSurv - evaluation

The Concordance Index (C-index) is a widely used metric for assessing the predictive accuracy of survival models. It measures the proportion of concordant pairs. DeepSurv has demonstrated superior performance compared to the Cox Proportional Hazards (Cox-PH) model across various datasets.

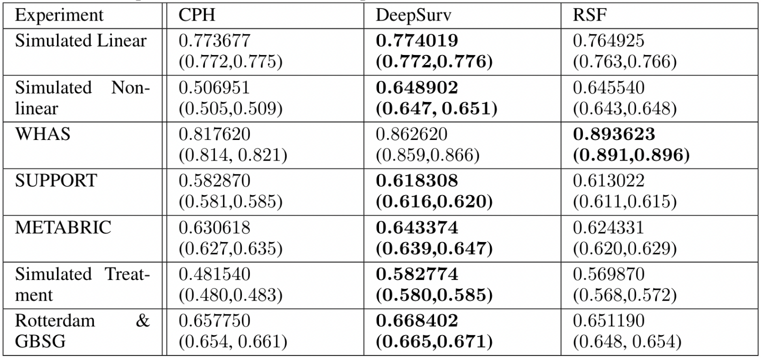


Table 1: Experimental Results for All Experiments: C-index (95% Confidence Interval)

Experiment	CPH	DeepSurv	RSF
Simulated Linear	0.773677 (0.772,0.775)	**0.774019 (0.772,0.776)**	0.764925 (0.763,0.766)
Simulated Non-linear	0.506951 (0.505,0.509)	**0.648902 (0.647, 0.651)**	0.645540 (0.643,0.648)
WHAS	0.817620 (0.814, 0.821)	0.862620 (0.859,0.866)	**0.893623 (0.891,0.896)**
SUPPORT	0.582870 (0.581,0.585)	**0.618308 (0.616,0.620)**	0.613022 (0.611,0.615)
METABRIC	0.630618 (0.627,0.635)	**0.643374 (0.639,0.647)**	0.624331 (0.620,0.629)
Simulated Treatment	0.481540 (0.480,0.483)	**0.582774 (0.580,0.585)**	0.569870 (0.568,0.572)
Rotterdam & GBSG	0.657750 (0.654, 0.661)	**0.668402 (0.665,0.671)**	0.651190 (0.648, 0.654)

DeepSurv - prediction etc

DeepSurv only provides proportional hazards. To obtain the hazard function and predict survival time, an additional distribution assumption on the baseline hazard function is required.

AFT based models

Lets briefly review the AFT model

$$Y_i = X_i'\beta + W_i,$$

where $Y_i = \log T_i$ and W_i follows some distribution. A straightforward approach involves using a neural network to model the distribution parameters and employing likelihood (accounting for censoring) to train the model.
Specifically, assuming W_i follows a Weibull distribution, two fully connected neural networks, $\alpha(X_i)$ and $\beta(X_i)$, are used to model its parameters. The model is then trained by maximizing the following likelihood.

$$\prod_{i=1}^{n} f^{\delta_i}(t_i) S^{1-\delta_i}(t_i)$$

AFT based models - DART

Deep AFT Rank-regression model for Time-to-event prediction (DART) [Lee+23] is based a semi-parameterized AFT model

$$\log T_i = g(X_i;\theta) + \epsilon_i,$$

where $g(X_i;\theta)$ denotes arbitrary neural networks with input feature X_i and parameter θ. let $e_i = \log t_i - g(X_i;\theta)$, the objective loss function can be formulated as

$$L_{rank} = \frac{1}{N}\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(e_i - e_j)I(e_i \geq e_j)$$

AFT based models - DART

The predicted output $g(X_i;\theta)$ from the DART model represents the estimated expectation of $\log T_i$.
For AFT-based models, estimating survival quantities such as $\lambda(t)$ cannot be done directly. Instead, they use the Nelson-Aalen estimator to determine the baseline hazard function $\lambda_0(t)$, and then define the conditional hazard function accordingly.

$$\lambda(t|X_i) = \lambda(t)_0(t\exp(-g(X_i;\theta)))\exp(-g(X_i;\theta))$$

AFT based models - RankDeepSurv

The RankDeepSurv[Jin+19] model is similar to DART except for an additional loss term. Specifically, RankDeepSurv use the following model

$$T_i = g(X_i;\theta) + \epsilon_i.$$

The loss function contains two terms

$$L_1 = \frac{1}{n}\sum_{I(j)=1}(g(X_j;\theta) - y_j)^2, \quad I(j) = \begin{cases} 1, if & \delta_j = 1 \\ & or \delta_j = 0, g(X_j;\theta) \leq y_j \\ 0, else & \end{cases}$$

and

$$L_2 = \frac{1}{N}\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(e_i - e_j)I(e_i \geq e_j),$$

where $e_i = t_i - g(X_i;\theta)$

AFT based models - RankDeepSurv

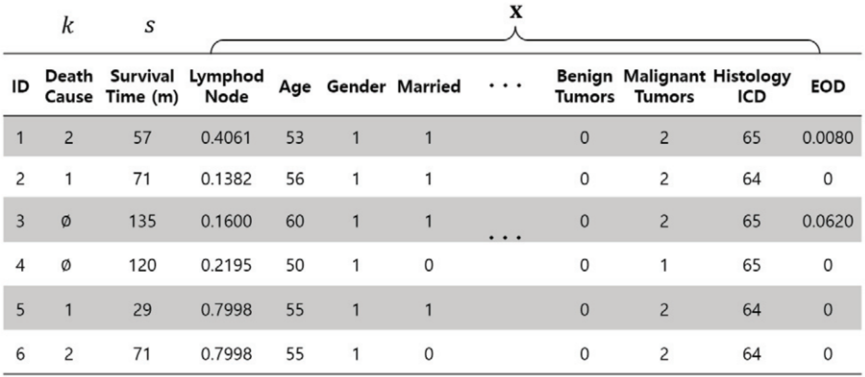
L_2 is similar to the rank loss in DART while L_1 is to enhance survival time prediction accuracy. Finally, the loss is defined as

$$L = \alpha \times L_1 + \beta \times L_2$$

Other methods - DeepHit

- DeepHit [Lee+18] is a distinctive model that diverges from traditional statistical models. Unlike Accelerated Failure Time (AFT) or hazard-based models, it directly models the probability of events occurring at specific times.
- DeepHit treats survival time as discrete by rounding up continuous times and considers the time horizon as $T_i \leq T_{max}$. It accounts for K possible events of interest, with one event occurring for each instance (e.g., a patient can die from one specific cause).
- DeepHit try to model $P(t, k|X)$, the probability that a particular event k occurs on t conditional on covariates X.

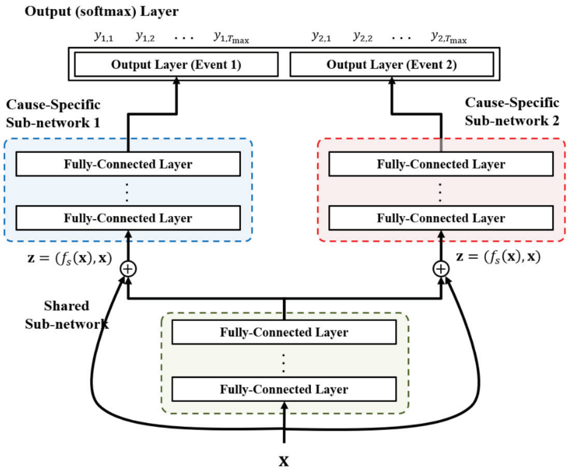
Other methods - DeepHit



	k	s	$\mathbf{x}$								
ID	Death Cause	Survival Time (m)	Lymphod Node	Age	Gender	Married	· · ·	Benign Tumors	Malignant Tumors	Histology ICD	EOD
1	2	57	0.4061	53	1	1		0	2	65	0.0080
2	1	71	0.1382	56	1	1		0	2	64	0
3	ø	135	0.1600	60	1	1	· · ·	0	2	65	0.0620
4	ø	120	0.2195	50	1	0		0	1	65	0
5	1	29	0.7998	55	1	1		0	2	64	0
6	2	71	0.7998	55	1	0		0	2	64	0

Figure: An example dataset (SEER)

Other methods - DeepHit



Output (softmax) Layer
$y_{1,1}$ $y_{1,2}$ ... $y_{1,T_{\max}}$
$y_{2,1}$ $y_{2,2}$ ... $y_{2,T_{\max}}$
Output Layer (Event 1)
Output Layer (Event 2)
Cause-Specific Sub-network 1
Cause-Specific Sub-network 2
Fully-Connected Layer
Fully-Connected Layer
Fully-Connected Layer
Fully-Connected Layer
$\mathbf{z} = (f_s(\mathbf{x}), \mathbf{x})$
$\mathbf{z} = (f_s(\mathbf{x}), \mathbf{x})$
Shared Sub-network
Fully-Connected Layer
Fully-Connected Layer
$\mathbf{x}$

Figure: Network structure of DeepHit with two events

Other methods - DeepHit

- The network employs a shared embedding layer and a separate tower for each event, a common structure in multi-task learning.
- Let $y_{k,t}^{(i)}, 1 \leq k \leq K, 1 \leq t \leq T_{max}$ be the outputs of the network, which models $P(t, k|X_i)$.
- The softmax layer ensures that the probabilities of each event occurring at different times sum up to one.

Other methods - DeepHit

The loss function of DeepHit is also the sum of two terms $L = L_1 + L_2$.

$$L_1 = -\sum_{i=1}^{n} \left[\delta_i \log(y_{k,t}^{(i)}) + (1-\delta_i) \log(1 - \sum_{k}^{K} \sum_{t=1}^{t_i} y_{k,t}^{(i)}) \right],$$

the first term captures the information provided by uncensored patients, while the second term addresses the censoring bias by utilizing the knowledge that patients are alive at the censoring time.

$$L_2 = \sum_{k=1}^{K} \alpha_k \sum_{i \neq j} 1(k^{(i)} = k, t_i < t_j) \eta(\hat{F}(t_i|X_i), \hat{F}(t_i|x_j))$$

Incorporating L_2 into the total loss function penalizes incorrect ordering of pairs (with respect to each event) and so minimizing the total loss encourages correct ordering of pairs (with respect to each event).

References I

[Jin+19] Bingzhong Jing et al. “A deep survival analysis method based on ranking”. In: *Artificial intelligence in medicine* 98 (2019), pp. 1–9.

[Kat+18] Jared L Katzman et al. “DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network”. In: *BMC medical research methodology* 18 (2018), pp. 1–12.

[Lee+18] Changhee Lee et al. “Deephit: A deep learning approach to survival analysis with competing risks”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[Lee+23] Hyunjun Lee et al. “Towards Flexible Time-to-Event Modeling: Optimizing Neural Networks via Rank Regression”. In: *arXiv preprint arXiv:2307.08044* (2023).

Table of contents

Problem formulation

- Let $(X, Y) \sim P_{X,Y}$, QR concerns the τth conditional quantile

$$Q_Y^{\tau}(x) = F_{Y|X=x}^{-1}(\tau), \qquad \text{for } \tau \in (0,1).$$

- Given $\tau \in (0,1)$, the $Q_Y^{\tau}(x)$ can be consistently estimated by

$$\arg\min_{f \in \mathcal{F}} \mathbb{E}_{X,Y}[\rho_\tau(Y - f(X))],$$

  where $\rho_\tau(a) = a[\tau - 1(a < 0)]$ is the check loss and $\mathcal{F}$ is a class of neural networks.

- Objective of distributional learning: $Q_Y^{\tau_1}(x), \ldots, Q_Y^{\tau_K}(x)$ at K levels:

$$\arg\min_{f \in \mathcal{F}} L(f) = \arg\min_{f \in \mathcal{F}} \sum_{k=1}^{K} \frac{1}{K} \mathbb{E}_{X,Y}[\rho_{\tau_k}(Y - f_k(X))]. \qquad (1)$$

DQR

- DQR[PTC22] provides the most straightforward deep learning solution for quantile regression.
- Specifically, DQR use K different neural networks to model each of the $f_k(X)$. Then use equation 1 as learning objective.

Crossing Issue

From the definition of quantile, an implicit restriction in quantile regression is the prevention of crossing quantiles. Specifically,

$$Q_Y^{\tau_1}(x) \leq \ldots \leq Q_Y^{\tau_K}(x)$$

The DQR model fails to satisfy the non-crossing condition, which reduces estimation accuracy (as demonstrated in the practice section) and compromises model interpretability.
Therefore, many models improve upon the DQR model to ensure the non-crossing property.

DQR*

DQR* is an extension of the DQR method for multiple quantile estimation that incorporates non-crossing constraints using ReLU neural networks.

$$\hat{h}_{\tau_k} = \arg\min_{h_{\tau_k} \subset \mathcal{F}} \sum_{i=1}^{n} \rho_\tau(y_i - h_{\tau_0}(x_i)) + \sum_{j=1}^{m} \sum_{i=1}^{n} \rho_{\tau_k} \left(y_i - h_{\tau_0}(x_i) - \sum_{l=1}^{j} \log\left(1 + e^{h_{\tau_l}(x_i)}\right) \right)$$

and set

$$Q_Y^{\tau_0}(x) = \hat{h}_{\tau_0}(x), \quad \text{and} \quad Q_Y^{\tau_1}(x) (= \hat{h}_{\tau_0}(x) + \sum_{l=1}^{j} \log\left(1 + e^{\hat{h}_{\tau_l}(x)}\right).$$

The non-negativity of the $\log(1 + e^x)$ function ensures the non-crossing property.

NCQR

Non-crossing Quantile Regression(NCQR) [ZWF20] Deep-Q-Network is proposed for Distributional RL by using non-crossing quantile regression based on the QR-DQN architecture is Distributional RL literature.

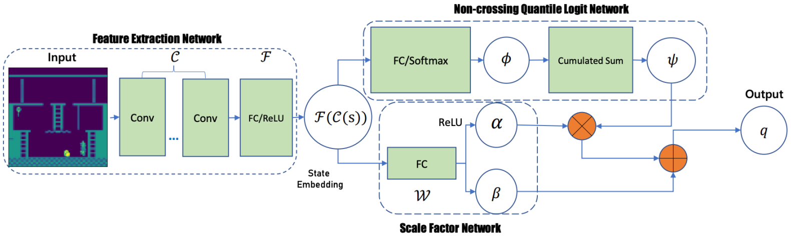
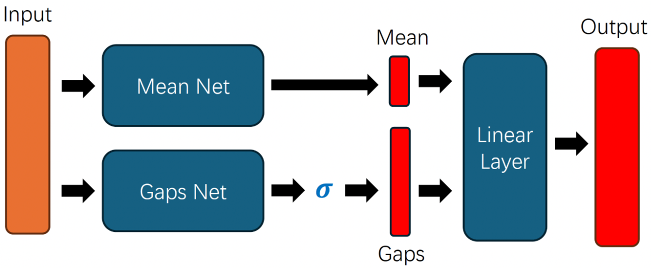


Figure 2: A general picture of the NC-QR-DQN architecture

The logit network generates a series of non-decreasing values through a softmax layer followed by a cumulative sum. The quantiles at different levels τ_k are then modeled by multiplying a positive scale and an intercept.

NQ-Net

Similar to NCQR, NQ-net also have two sub-networks.



where the mean net $v(x)$ tries to model $\frac{1}{K}\sum_{k=1}^{K} Q_Y^{\tau_k}(x)$ while the K gaps $\sigma_k(x)$ model $= Q_Y^{\tau_{k+1}}(x) - Q_Y^{\tau_k}(x)$.

NQ-Net

- The mean net $v(x)$ is a multi-layer feed forward neural network with one output.
- The gap net $\sigma_k(x) = ELU(x) + 1$, where $ELU(x) = x$ if $x \geq 0$ and $exp(x) - 1$ otherwise. This model setup ensures $\sigma_k(x) \geq 0$ thus ensures the non-crossing property.

NQ-Net

- NQ net $f(x) = v(x) \oplus (ELU + 1)(d(x)) \in \mathbb{R}^K$ with $\mathcal{D}$ hidden layers

$$\begin{pmatrix} v(x) \\ d(x) \end{pmatrix} = \mathcal{L}_{\mathcal{D}} \circ \sigma \circ \mathcal{L}_{\mathcal{D}-1} \circ \sigma \circ \cdots \circ \sigma \circ \mathcal{L}_1 \circ \sigma \circ \mathcal{L}_0(x), x \in \mathbb{R}^{d_0}.$$

  1. $\mathcal{L}_i(x) = W_i x + b_i$ is the i-th linear transformation with $x \in \mathbb{R}^{p_i}$ where $W_i \in \mathbb{R}^{p_{i+1} \times p_i}$ is the weight matrix and $b_i \in \mathbb{R}^{p_{i+1}}$ is the bias vector.
  2. $\sigma = \max\{x, 0\}$ is the rectified linear unit (ReLU) activation function

- Class of NQ networks $\mathcal{F} = \{f \text{ over all possible choice of } \{(\mathrm{W_i, b_i})\}_{\mathrm{i=0}}^{\mathcal{D}}, \text{and } \|\mathrm{f}\|_\infty \le \mathcal{B}, \|\frac{\partial}{\partial \tau}\mathrm{f}\|_\infty \le \mathcal{B}'\}.$
  1. Depth $\mathcal{D}$, width $\mathcal{W} = \max\{p_1, ..., p_{\mathcal{D}}\}$
  2. Size $\mathcal{S} = \sum_{i=0}^{\mathcal{D}}\{p_{i+1} \times (p_i + 1)\}$
  3. Number of neurons $\mathcal{U} = \sum_{i=1}^{\mathcal{D}} p_i$

NQ-Net Learning Guarantee

Theorem (Non-asymptotic upper bounds)

Suppose the ground truth Q^Y *are* β*-Hölder smooth. For any integers* $U, M \in \mathbb{N}^+$*, let the class of networks* $\mathcal{F}$ *uniformly bounded by* $\mathcal{B}$*, has width* $\mathcal{W} = 38(K+1)(\lfloor\beta\rfloor+1)^2 d_0^{\lfloor\beta\rfloor+1} U \log_2(8U)$ *and depth* $\mathcal{D} = 21(\lfloor\beta\rfloor+1)^2 d_0^{\lfloor\beta\rfloor+1} M \log_2(8M)$*. Then for any* $\delta > 0$*, with prob. at least* $1-\delta$

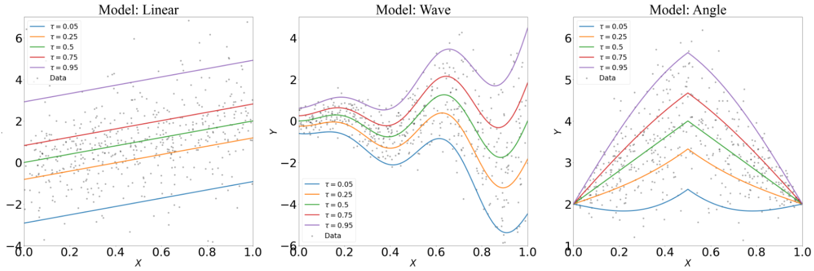
$$\mathcal{R}(\hat{f}_N) := \mathcal{L}(\hat{f}_N) - \mathcal{L}(Q_Y) \le \frac{2\sqrt{2}(K+2)\mathcal{B}}{\sqrt{N}}\left(C\sqrt{K\mathcal{S}\mathcal{D}\log(\mathcal{S})\log(N)} + \sqrt{\log(1/\delta)}\right)$$
$$+ 18(K+2)\mathcal{B}(\lfloor\beta\rfloor+1)^2 d_0^{\lfloor\beta\rfloor+(\beta\vee 1)/2}(UM)^{-2\beta/d_0}$$
$$+ (K+2)\exp(-\mathcal{B}) \qquad sho$$

for $N \ge c \cdot \mathcal{D}\mathcal{S}\log(\mathcal{S})$ *where* $C, c > 0$ *are universal constants, and* d_0 *is the input dimension of the target quantile functions* Q_Y *and also neural networks in* $\mathcal{F}$*.*

- Stochastic error(variance) increasing in network size, decreasing in sample size N.
- Approximation error(bias) decreasing in network size, smoothness β of target Q^Y.

Simulation and Comparison

To compare these different deep quantile regression models. We we generate the data (X, Y) through an additive model $Y = g(X) + \epsilon$.

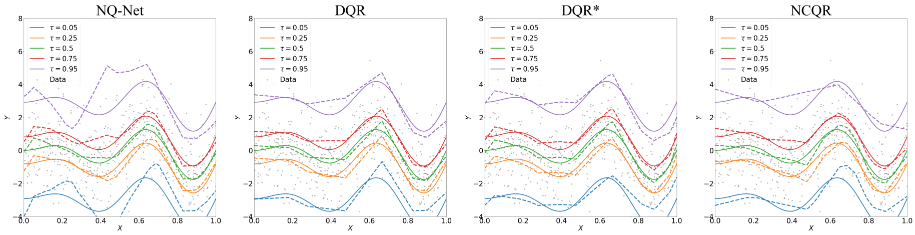


1. Linear: $f_0^\tau(x) = 2x + F_t^{-1}(\tau)$,
2. Wave: $f_0^\tau(x) = 2x\sin(4\pi x) + \exp(4x-2)\Phi^{-1}(\tau)$,
3. Angle: $f_0^\tau(x) = 4(1 - |x - 0.5|) + |\sin(\pi x)|\Phi^{-1}(\tau)$,

where $F_t(\cdot)$ is the CDF of the Student's t with degree of freedom 2, $\Phi(\cdot)$ is the CDF of standard normal.

Simulation and Comparison

In the coding section, we will implement these models and the simulation. Here, we present only the results.



References I

[PTC22] Oscar Hernan Madrid Padilla, Wesley Tansey, and Yanzhen Chen. “Quantile regression with ReLU networks: Estimators and minimax rates”. In: *Journal of Machine Learning Research* 23.247 (2022), pp. 1–42.

[ZWF20] Fan Zhou, Jianing Wang, and Xingdong Feng. “Non-crossing quantile regression for distributional reinforcement learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 15909–15919.

Table of contents

Introduction

Quantile regression has proven valuable in survival analysis, as it directly predicts the variable of interest, naturally capturing the uncertainty of the conditional distribution without making assumptions about the distribution.

In contrast to deep survival models, there has been limited research in deep quantile survival analysis.

CQRNN

Censored quantile regression neural network (CQRNN) [PJZ+22] is a deep quantile survival model based on the Portnoy estimator.
The Portnoy estimator minimizes the following loss

$$L_k = \sum_{i=1}^{n} \left[\delta_i \rho_{\tau_k}(t_i, \hat{t}_{i,\tau_k}) + (1-\delta_i) \left[w_i \rho_{\tau_k}(t_i, \hat{t}_{i,\tau_k}) + (1-w_i)\rho_{\tau_k}(t^*, \hat{t}_{i,\tau_k}) \right] \right]$$

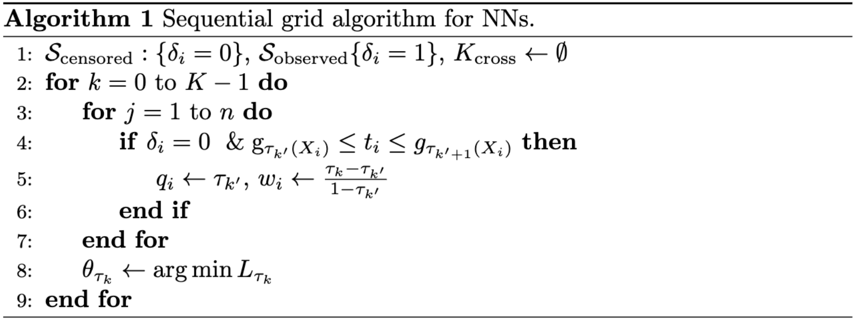
where $\hat{t}_{i,\tau_k}$ is the estimated value of the τ_k-th quantile given X_i, e.g $X_i^T \beta_{\tau_k}$, $t^* \geq max_i t_i$ is some vary large value.

$$w_i = \frac{\tau - q_i}{1 - q_i}$$

where q_i is the quantile at which the datapoint was censored, with respect to the distribution of $\hat{t}_i$. Given these weights, the estimator has been shown to be analogous to KM estimator.

Sequential-Portnoy

Let $\hat{t}_{i,\tau_k} = g_{\tau_k}(X_i;\theta_{\tau_k})$ $k \leq K$ be K neural networks. The parameters of θ_{τ_k}s can be estimated in a sequential manner.

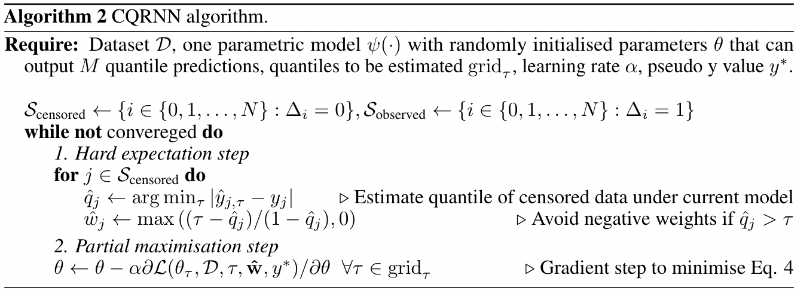


Algorithm 1 Sequential grid algorithm for NNs.

```
1: $\mathcal{S}_{\text{censored}} : \{\delta_i = 0\}$, $\mathcal{S}_{\text{observed}}\{\delta_i = 1\}$, $K_{\text{cross}} \leftarrow \emptyset$
2: for $k = 0$ to $K - 1$ do
3:     for $j = 1$ to $n$ do
4:         if $\delta_i = 0$ & $g_{\tau_{k'}}(X_i) \leq t_i \leq g_{\tau_{k'+1}}(X_i)$ then
5:             $q_i \leftarrow \tau_{k'}$, $w_i \leftarrow \frac{\tau_k - \tau_{k'}}{1-\tau_{k'}}$
6:         end if
7:     end for
8:     $\theta_{\tau_k} \leftarrow \arg\min L_{\tau_k}$
9: end for
```

CQRNN

The sequential Portnoy algorithm is inefficient because the K estimators must be trained sequentially, resulting in a complexity of $O(K)$. While these inefficiencies are less problematic for linear models, the use of neural networks motivates the search for an alternative algorithm.



Algorithm 2 CQRNN algorithm.

```
Require: Dataset $\mathcal{D}$, one parametric model $\psi(\cdot)$ with randomly initialised parameters $\theta$ that can
  output $M$ quantile predictions, quantiles to be estimated grid$_\tau$, learning rate $\alpha$, pseudo y value $y^*$.

  $\mathcal{S}_{\text{censored}} \leftarrow \{i \in \{0, 1, \dots, N\} : \Delta_i = 0\}, \mathcal{S}_{\text{observed}} \leftarrow \{i \in \{0, 1, \dots, N\} : \Delta_i = 1\}$
  while not converged do
      1. Hard expectation step
      for $j \in \mathcal{S}_{\text{censored}}$ do
          $\hat{q}_j \leftarrow \arg\min_\tau |\hat{y}_{j,\tau} - y_j|$        ▷ Estimate quantile of censored data under current model
          $\hat{w}_j \leftarrow \max((\tau - \hat{q}_j)/(1 - \hat{q}_j), 0)$        ▷ Avoid negative weights if $\hat{q}_j > \tau$
      2. Partial maximisation step
      $\theta \leftarrow \theta - \alpha\partial\mathcal{L}(\theta_\tau, \mathcal{D}, \tau, \hat{\mathbf{w}}, y^*)/\partial\theta \;\; \forall\tau \in \text{grid}_\tau$        ▷ Gradient step to minimise Eq. 4
```

References I

[PJZ+22] Tim Pearce, Jong-Hyeon Jeong, Jun Zhu, et al. “Censored quantile regression neural networks for distribution-free survival analysis”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 7450–7461.

Table of contents

- In this section, we are going to see under the hood of DeepSurv and train some Deep Quantile Regression models, including NQ-Net, DQR, DQR^* and NC-QR.