in partnership with

ICR The Institute of Cancer Research

CANCER RESEARCH UK

The ROYAL MARSDEN NHS Foundation Trust

An Efficient Training Algorithm for Kernel Survival Support Vector Machines

Sebastian Pölsterl[1] (sebastian.poelsterl@icr.ac.uk),
Nassir Navab[2,3], Amin Katouzian[4]

1 The Institute of Cancer Research, London, UK
2 Technische Universität München, Munich, Germany
3 Johns Hopkins University, Baltimore, MD, USA
4 IBM Almaden Research Center, San Jose, CA, USA

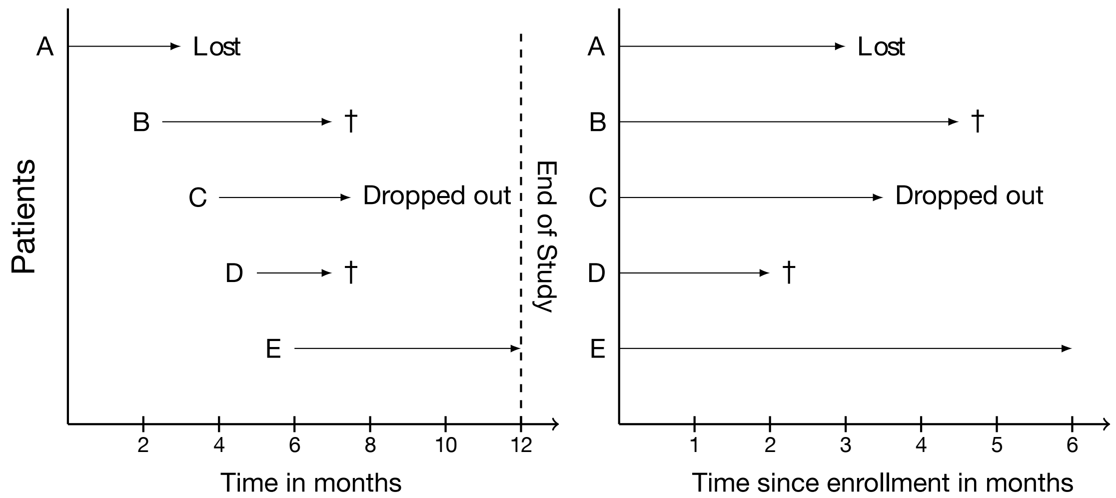
Workshop on Machine Learning in Life Sciences, Riva del Garda, Italy
23 September 2016

Making the discoveries that defeat cancer

Survival Analysis

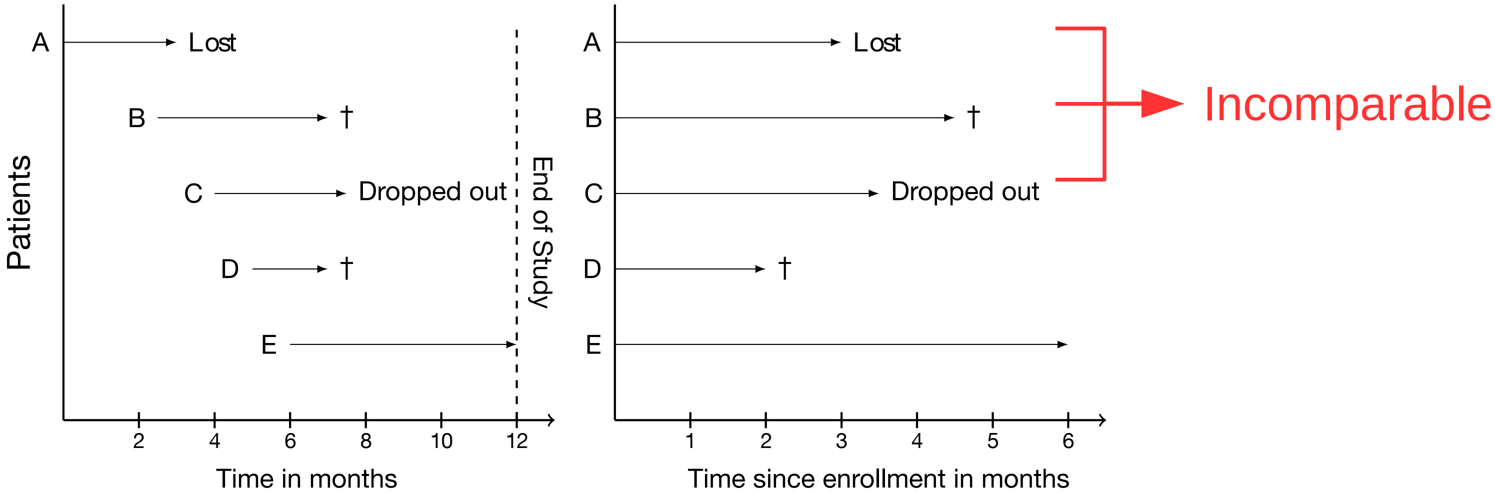
- **Objective**: to establish a connection between a set of features and the time between the start of the study and an event.
- Usually, parts of training and test data can only be partially observed – they are **censored**.
- The survival support vector machine (SSVM) formulates **survival analysis as a ranking-to-rank problem**.
- Survival data consists of *n* triplets:
  - $\boldsymbol{x}_i \in \mathbb{R}^p$ a *p*-dimensional feature vector
  - $y_i = \min(t_i, c_i)$ time of event (t_i) or time of censoring (c_i)
  - $\delta_i = I(t_i < c_i)$ event indicator

Right Censoring



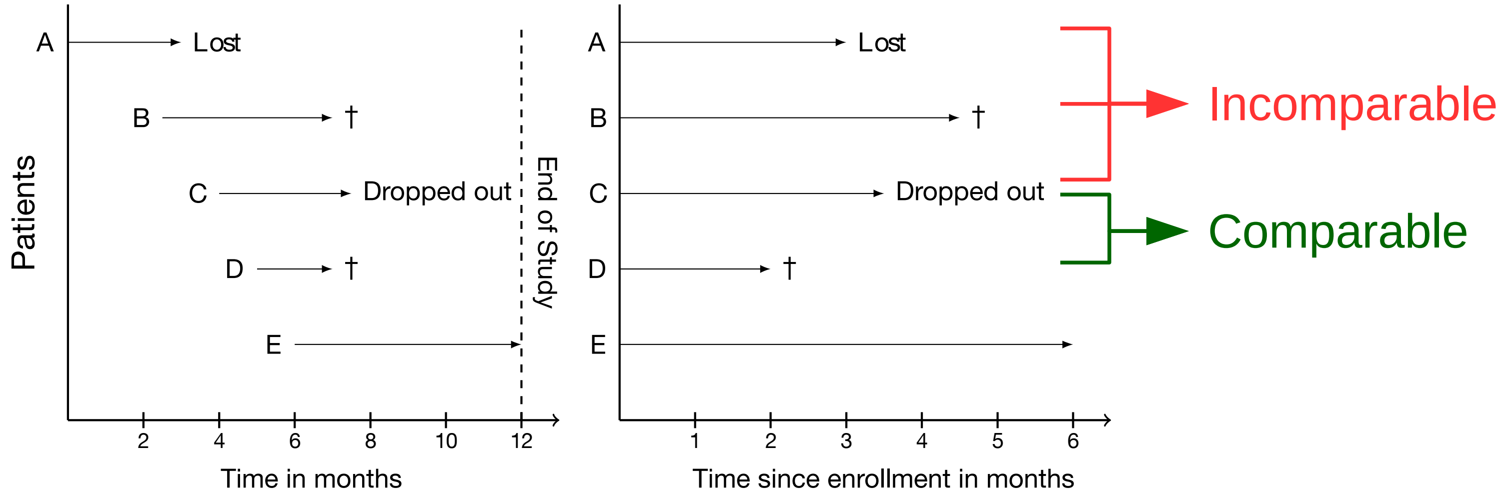
- Only events that occur while the study is running can be recorded (records are **uncensored**).
- For individuals that remained event-free during the study period, it is unknown whether an event has or has not occurred after the study ended (records are **right censored**).

Right Censoring



- Only events that occur while the study is running can be recorded (records are **uncensored**).
- For individuals that remained event-free during the study period, it is unknown whether an event has or has not occurred after the study ended (records are **right censored**).

Right Censoring



- Only events that occur while the study is running can be recorded (records are **uncensored**).
- For individuals that remained event-free during the study period, it is unknown whether an event has or has not occurred after the study ended (records are **right censored**).

Kernel Survival Support Vector Machine

- The survival support vector machine (SSVM) is an extension of the Rank SVM to right censored survival data (Herbrich et al., 2000; Van Belle et al., 2007; Evers et al., 2008):
  - *Rank patients with a lower survival time before patients with longer survival time.*
- **Objective function**: $\mathcal{P} = \{(i,j) \mid y_i > y_j \wedge \delta_j = 1\}_{i,j=1}^{n}$

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \gamma \sum_{(i,j)\in P} \max(0, 1 - \boldsymbol{w}^\top(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)))$$

- **Lagrange dual problem with $\boldsymbol{K}_{i,j} = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)$:**

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \boldsymbol{\alpha}^\top \mathbf{1}_m - \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{A}\boldsymbol{K}\boldsymbol{A}^\top \boldsymbol{\alpha} \\ \text{subject to} \quad & 0 \leq \alpha_{ij} \leq \gamma, \quad \forall (i,j) \in P, \end{aligned}$$

where $\boldsymbol{A}_{k,i} = 1$ and $\boldsymbol{A}_{k,j} = -1$ if $(i,j) \in \mathcal{P}$ and 0 otherwise.

Kernel Survival Support Vector Machine

- The survival support vector machine (SSVM) is an extension of the Rank SVM to right censored survival data (Herbrich et al., 2000; Van Belle et al., 2007; Evers et al., 2008):
  - *Rank patients with a lower survival time before patients with longer survival time.*
- **Objective function:** $\mathcal{P} = \{(i,j) \mid y_i > y_j \wedge \delta_j = 1\}_{i,j=1}^{n}$ Set of comparable pairs

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \gamma \sum_{(i,j)\in P} \max(0, 1 - \boldsymbol{w}^\top(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)))$$

- **Lagrange dual problem with** $\boldsymbol{K}_{i,j} = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)$:

$$\max_{\boldsymbol{\alpha}} \quad \boldsymbol{\alpha}^\top \mathbf{1}_m - \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{A}\boldsymbol{K}\boldsymbol{A}^\top \boldsymbol{\alpha}$$
$$\text{subject to} \quad 0 \leq \alpha_{ij} \leq \gamma, \quad \forall (i,j) \in P,$$

where $\boldsymbol{A}_{k,i} = 1$ and $\boldsymbol{A}_{k,j} = -1$ if $(i,j) \in \mathcal{P}$ and 0 otherwise.

Kernel Survival Support Vector Machine

- The survival support vector machine (SSVM) is an extension of the Rank SVM to right censored survival data (Herbrich et al., 2000; Van Belle et al., 2007; Evers et al., 2008):
  - *Rank patients with a lower survival time before patients with longer survival time.*
- **Objective function:** $\mathcal{P} = \{(i, j) \mid y_i > y_j \wedge \delta_j = 1\}_{i,j=1}^{n}$ Set of comparable pairs

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \gamma \sum_{(i,j)\in P} \max(0, 1 - \boldsymbol{w}^\top(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)))$$

- **Lagrange dual problem with** $\boldsymbol{K}_{i,j} = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)$ **:**

$$\max_{\boldsymbol{\alpha}} \quad \boldsymbol{\alpha}^\top \mathbf{1}_m - \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{A}\boldsymbol{K}\boldsymbol{A}^\top \boldsymbol{\alpha}$$

Requires *O*(n^4) space

$$\text{subject to} \quad 0 \leq \alpha_{ij} \leq \gamma, \quad \forall (i, j) \in P,$$

where $\boldsymbol{A}_{k,i} = 1$ and $\boldsymbol{A}_{k,j} = -1$ if $(i, j) \in \mathcal{P}$ and 0 otherwise.

Training the Kernel SSVM

- **Problem**: For a dataset with *n* samples and *p* features, previous training algorithms require $O(n^4)$ space and $O(pn^6)$ time.
- Recently, an **efficient training algorithm for linear SSVM** with much lower time complexity and linear space complexity has been proposed (Pölsterl et al., 2015).
- We **extend this optimisation scheme to the non-linear case** and show that it allows analysing large-scale data with no loss in prediction performance.

Proposed Optimisation Scheme

The form of the optimisation problem is very similar to the one of linear SSVM, which allows applying many of the ideas employed in its optimisation

- Substitute hinge loss for differentiable **squared hinge**
- Perform **optimisation in the primal** rather than the dual
  - **Directly apply the representer theorem** (Kuo et al., 2014)
  - Use truncated Newton optimisation (Dembo and Steihaug, 1983)
  - Use order statistic trees to avoid explicitly constructing all pairwise comparisons of samples, i.e., storing matrix A (Pölsterl et al., 2015)

Objective Function (1)

Find a function $f: \mathcal{X} \to \mathbb{R}$ from a reproducing Kernel Hilbert space $\mathcal{H}_k$ with $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (usually $\mathcal{X} \subset \mathbb{R}^p$):

$$\min_{f \in \mathcal{H}_k} \frac{1}{2} \|f\|^2_{\mathcal{H}_k} + \frac{\gamma}{2} \sum_{(i,j) \in P} \max(0, 1 - (f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)))^2$$

Objective Function (2)

Apply representer theorem to express $f(\boldsymbol{z})$ as $f(\boldsymbol{z}) = \sum_{i=1}^{n} \beta_i k(\boldsymbol{x}_i, \boldsymbol{z})$, where $\boldsymbol{\beta} \in \mathbb{R}^n$ are the coefficients (Kuo et al., 2014).

$$\min_{\boldsymbol{\beta}} R(\boldsymbol{\beta}) \Leftrightarrow \min_{f \in \mathcal{H}_k} \frac{1}{2}\|f\|_{\mathcal{H}_k}^2 + \frac{\gamma}{2} \sum_{(i,j) \in P} \max(0, 1 - (f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)))^2$$

$$\begin{aligned} R(\boldsymbol{\beta}) =& \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_i \beta_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &+ \frac{\gamma}{2} \sum_{(i,j) \in P} \max\left(0, 1 - \sum_{l=1}^{n} \beta_l (k(\boldsymbol{x}_l, \boldsymbol{x}_i) - k(\boldsymbol{x}_l, \boldsymbol{x}_j))\right)^2 \\ =& \frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{K} \boldsymbol{\beta} + \frac{\gamma}{2} (\mathbf{1}_m - \boldsymbol{A}\boldsymbol{K}\boldsymbol{\beta})^\top \boldsymbol{D}_{\boldsymbol{\beta}} (\mathbf{1}_m - \boldsymbol{A}\boldsymbol{K}\boldsymbol{\beta}) \end{aligned}$$

$$(\boldsymbol{D}_{\boldsymbol{\beta}})_{k,k} = \begin{cases} 1 & \text{if } f(\boldsymbol{x}_j) > f(\boldsymbol{x}_i) - 1 \Leftrightarrow \boldsymbol{K}_j \boldsymbol{\beta} > \boldsymbol{K}_i \boldsymbol{\beta} - 1, \\ 0 & \text{else.} \end{cases}$$

Truncated Newton Optimisation (1)

- **Problem**: Explicitly storing the Hessian matrix can be prohibitive for large-scale survival data.
- Avoid constructing Hessian matrix by using truncated Newton optimization, which only requires computation of Hessian-vector product (Dembo and Steihaug, 1983).
- Hessian:

$$\boldsymbol{H} = \frac{\partial^2 R(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} = \boldsymbol{K} + \gamma \boldsymbol{K} \boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}} \boldsymbol{K} \qquad (\text{with } \boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}} = \boldsymbol{A}^\top \boldsymbol{D}_{\boldsymbol{\beta}} \boldsymbol{A})$$

- Hessian-vector product:

$$\boldsymbol{H}\boldsymbol{v} = \boldsymbol{K}\boldsymbol{v} + \gamma \boldsymbol{K} \boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}} \boldsymbol{K} \boldsymbol{v} = \boldsymbol{K}\boldsymbol{v} + \gamma \boldsymbol{K} \begin{pmatrix} (l_1^+ + l_1^-)\boldsymbol{K}_1 \boldsymbol{v} - (\sigma_1^+ + \sigma_1^-) \\ \vdots \\ (l_n^+ + l_n^-)\boldsymbol{K}_n \boldsymbol{v} - (\sigma_n^+ + \sigma_n^-) \end{pmatrix}$$

Truncated Newton Optimisation (2)

Hessian-vector product:

$$\boldsymbol{H}\boldsymbol{v} = \boldsymbol{K}\boldsymbol{v} + \gamma \boldsymbol{K} \begin{pmatrix} (l_1^+ + l_1^-)\boldsymbol{K}_1\boldsymbol{v} - (\sigma_1^+ + \sigma_1^-) \\ \vdots \\ (l_n^+ + l_n^-)\boldsymbol{K}_n\boldsymbol{v} - (\sigma_n^+ + \sigma_n^-) \end{pmatrix}$$

where in analogy to linear SSVM

$$\mathrm{SV}_i^+ = \{s \mid y_s > y_i \wedge \boldsymbol{K}_s\boldsymbol{\beta} < \boldsymbol{K}_i\boldsymbol{\beta} + 1 \wedge \delta_i = 1\}, \quad l_i^+ = |\mathrm{SV}_i^+|, \quad \sigma_i^+ = \sum_{s \in \mathrm{SV}_i^+} \boldsymbol{K}_s\boldsymbol{v}$$

$$\mathrm{SV}_i^- = \{s \mid y_s < y_i \wedge \boldsymbol{K}_s\boldsymbol{\beta} > \boldsymbol{K}_i\boldsymbol{\beta} - 1 \wedge \delta_s = 1\}, \quad l_i^- = |\mathrm{SV}_i^-|, \quad \sigma_i^- = \sum_{s \in \mathrm{SV}_i^-} \boldsymbol{K}_s\boldsymbol{v}$$

Truncated Newton Optimisation (2)

Hessian-vector product:

$$\boldsymbol{H}\boldsymbol{v} = \boldsymbol{K}\boldsymbol{v} + \gamma \boldsymbol{K} \begin{pmatrix} (l_1^+ + l_1^-)\boldsymbol{K}_1\boldsymbol{v} - (\sigma_1^+ + \sigma_1^-) \\ \vdots \\ (l_n^+ + l_n^-)\boldsymbol{K}_n\boldsymbol{v} - (\sigma_n^+ + \sigma_n^-) \end{pmatrix}$$

where in analogy to linear SSVM

$$\mathrm{SV}_i^+ = \{s \mid y_s > y_i \wedge \boldsymbol{K}_s\boldsymbol{\beta} < \boldsymbol{K}_i\boldsymbol{\beta} + 1 \wedge \delta_i = 1\}, \quad l_i^+ = |\mathrm{SV}_i^+|, \quad \sigma_i^+ = \sum_{s \in \mathrm{SV}_i^+} \boldsymbol{K}_s\boldsymbol{v}$$

$$\mathrm{SV}_i^- = \{s \mid y_s < y_i \wedge \boldsymbol{K}_s\boldsymbol{\beta} > \boldsymbol{K}_i\boldsymbol{\beta} - 1 \wedge \delta_s = 1\}, \quad l_i^- = |\mathrm{SV}_i^-|, \quad \sigma_i^- = \sum_{s \in \mathrm{SV}_i^-} \boldsymbol{K}_s\boldsymbol{v}$$

Can be computed in logarithmic time by first sorting by predicted scores $f(\boldsymbol{x}_i) = \boldsymbol{K}_i\boldsymbol{\beta}$ and incrementally constructing order statistic trees to hold SV_i^+ and SV_i^- (Pölsterl et al., 2015).

Complexity Analysis

- Assuming the kernel matrix $\boldsymbol{K}$ cannot be stored in memory and evaluating the kernel function costs $O(p)$
- Computing the Hessian-vector product during one iteration of truncated Newton optimisation requires
  1) $O(n^3 p)$ to compute $\boldsymbol{K}_i \boldsymbol{v}$ for all *i*
  2) $O(n \log n)$ to sort samples according to values of $\boldsymbol{K}_i \boldsymbol{v}$
  3) $O(n^2 + n + n \log n)$ to calculate the Hessian-vector product
- Overall (if kernel matrix is stored in memory):

$$O(n^2 p) + \left[O(n \log n) + O(n^2 + n + n \log n)\right] \cdot \bar{N}_{\mathrm{CG}} \cdot N_{\mathrm{Newton}}$$

Complexity Analysis

- Assuming the kernel matrix $\boldsymbol{K}$ cannot be stored in memory and evaluating the kernel function costs $O(p)$
- Computing the Hessian-vector product during one iteration of truncated Newton optimisation requires
  1) $O(n^3 p)$ to compute $\boldsymbol{K}_i \boldsymbol{v}$ for all *i*
  2) $O(n \log n)$ to sort samples according to values of $\boldsymbol{K}_i \boldsymbol{v}$
  3) $O(n^2 + n + n \log n)$ to calculate the Hessian-vector product
- Overall (if kernel matrix is stored in memory):

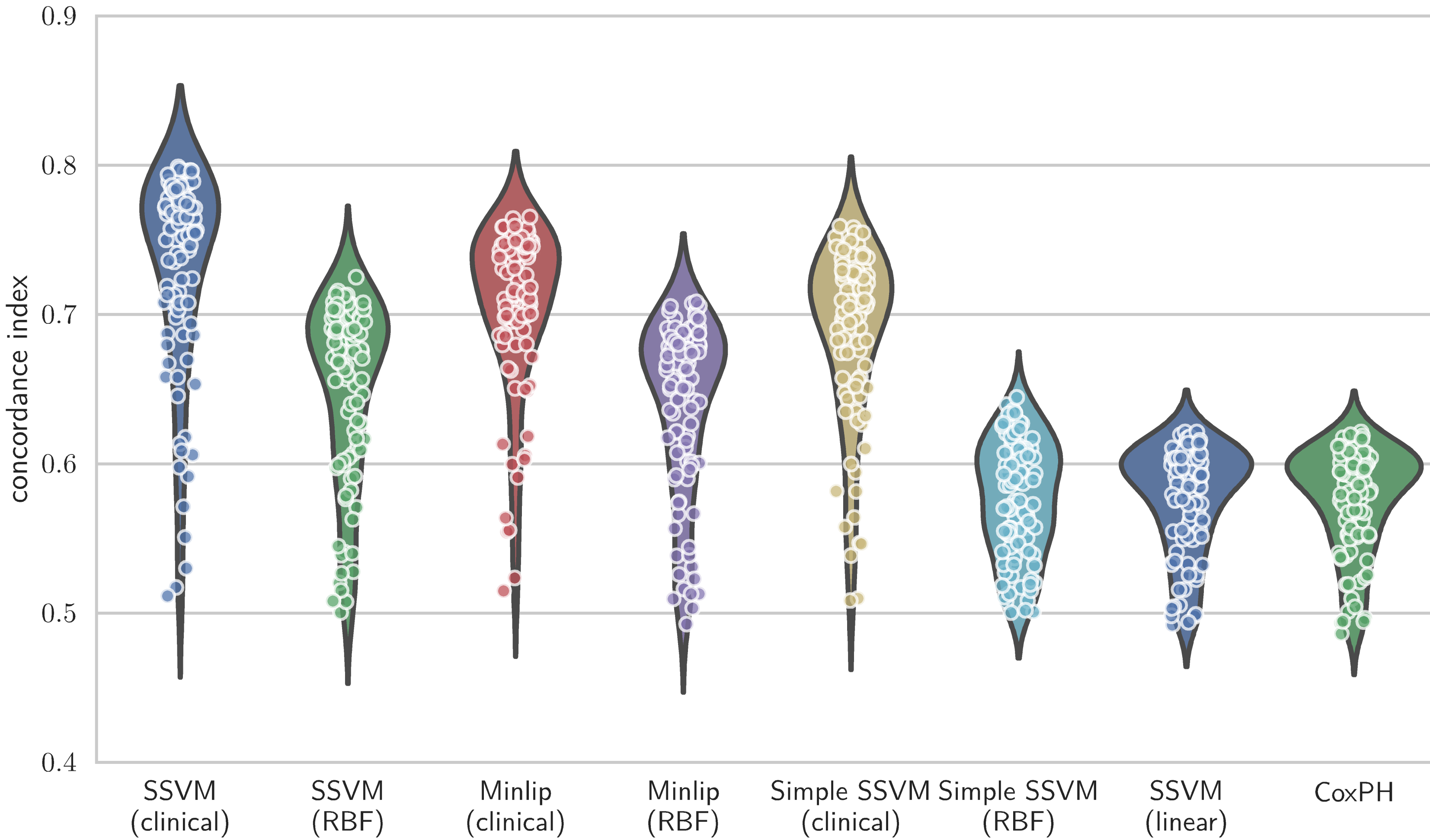
$$\boxed{O(n^2 p)} + \left[O(n \log n) + O(n^2 + n + n \log n)\right] \cdot \bar{N}_{\text{CG}} \cdot N_{\text{Newton}}$$

Constructing the kernel matrix is the bottleneck

Experiments

- **Synthetic data**: 100 pairs of train and test data of 1,500 samples with about 20% of samples right censored in the training data
- **Real-world datasets**: 5 datasets of varying size, number of features, and amount of censoring
- **Models**:
  - Simple SSVM with hinge loss and $\mathcal{P}$ restricted to pairs (i, j), where j is the largest uncensored sample with $y_i > y_j$ (Van Belle et al, 2008),
  - Minlip survival model (Van Belle et al., 2011),
  - linear SSVM (Pölsterl et al., 2015),
  - Cox's proportional hazards model with ℓ_2 penalty (Cox, 1972).
- **Kernels**:
  - RBF kernel
  - Clinical kernel (Daemen et al., 2012)

Experiments – Synthetic Data



Experiments – Real-world Data

		SSVM (ours)	SSVM (simple)	Minlip	SSVM (linear)	Cox
AIDS study (91.7% censored)	Harrel's c	**0.759**	0.682	0.729	0.767	0.770
	Uno's c	**0.711**	0.621	0.560	0.659	0.663
	iAUC	**0.759**	0.685	0.724	0.766	0.771
Coronary artery disease (86.5% censored)	Harrel's c	**0.739**	0.645	0.698	0.706	0.768
	Uno's c	**0.780**	0.751	0.745	0.730	0.732
	iAUC	**0.753**	0.641	0.703	0.716	0.777
Framingham offspring (76.2% censored)	Harrel's c	0.778	0.707	**0.786**	0.780	0.785
	Uno's c	**0.732**	0.674	0.724	0.699	0.742
	iAUC	0.827	0.742	**0.837**	0.829	0.832
Lung cancer (6.6% censored)	Harrel's c	0.676	0.605	**0.719**	0.716	0.716
	Uno's c	0.664	0.605	**0.716**	0.709	0.712
	iAUC	0.740	0.630	**0.790**	0.783	0.780
WHAS (57% censored)	Harrel's c	0.768	0.724	**0.774**	0.770	0.770
	Uno's c	0.772	0.730	**0.778**	0.775	0.773
	iAUC	0.799	0.749	**0.801**	0.796	0.796

Conclusion

- We proposed an efficient method for training non-linear ranking-based survival support vector machines
- Our algorithm is a straightforward extension of our previously proposed training algorithm for linear survival support vector machines
- Our optimisation scheme allows analysing datasets of much larger size than previous training algorithms
- Our optimisation scheme is the preferred choice when learning from survival data with high amounts of right censoring

in partnership with

ICR The Institute of Cancer Research

CANCER RESEARCH UK

The ROYAL MARSDEN
NHS Foundation Trust

Thanks for your attention!

Implementation in Python @ https://github.com/tum-camp/survival-support-vector-machine/

Acknowledgements

We would like to thank Bissan Al-Lazikani and Carmen Rodriguez-Gonzalvez.

This work has been supported by

- The CRUK Centre at the Institute of Cancer Research and Royal Marsden (Grant No. C309/A18077)
- The Heather Beckwith Charitable Settlement
- The John L Beckwith Charitable Trust
- The Leibniz Supercomputing Centre (LRZ, www.lrz.de)

Bibliography

Cox: Regression models and life tables. J. R. Stat. Soc. Series B Stat. Methodol. 34, pp. 187–220. 1972

Evers et al.: Sparse kernel methods for high-dimensional survival data. Bioinformatics 24(14). pp. 1632–38. 2008

Daemen et al.: Improved modeling of clinical data with kernel methods. Artif. Intell. Med. 54, pp. 103–14. 2012

Dembo and Steihaug: Truncated newton algorithms for large-scale optimization. Math. Program. 26(2). pp. 190–212. 1983

Herbrich et al.: Large margin rank boundaries for ordinal regression. Advances in Large Margin Classifiers. 2000

Kuo et al.: Large-scale kernel RankSVM. SIAM International Conference on Data Mining. 2014

Pölsterl et al.: Fast training of support vector machines for survival analysis. ECML PKDD 2015

Van Belle et al.: Support vector machines for survival Analysis. 3rd Int. Conf. Comput. Intell. Med. Healthc. 2007

Van Belle et al.: Survival SVM: a practical scalable algorithm. 16th Euro. Symb. Artif. Neural Netw. 2008

Van Belle et al.: Learning transformation models for ranking and survival analysis. JMLR. 12, pp. 819–62. 2011