ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

Regularisation Path for Ranking SVM

Karina Zapien[1], Thomas Gärtner[2], Gilles Gasso[1], and Stephane Canu[1]

1- LITIS - INSA De Rouen, St. Etienne du Rouvray, France
2- Fraunhofer IAIS, Schloß Birlinghoven, 53754 Sankt Augustin, Germany

Abstract. Ranking algorithms are often introduced with the aim of automatically personalising search results. However, most ranking algorithms developed in the machine learning community rely on a careful choice of some regularisation parameter. Building upon work on the regularisation path for kernel methods, we propose a parameter selection algorithm for ranking SVM. Empirical results are promising.

1 Introduction

Ranking algorithms are typically introduced as a tool for personalising the order in which (web) search results are presented, i.e., the more important a result is to the user, the earlier it should be listed. To this end, one can consider two possible settings: (i) the algorithm tries to interactively rearrange the results of one search such that relevant results come the closer to the top the more (implicit) feedback the user provides and (ii) the algorithm tries to generalise over several queries and presents the results of one search in an order depending on the feedback obtained from previous searches. Here, this problem is tackled using the rank SVM approach [1, 2].

Kernel methods like the SVM or the rank SVM solve optimisation problems of the form $\hat{f}^\lambda = \operatorname{argmin}_f \mathcal{V}[f] + \lambda\Omega[f]$ where $\mathcal{V}$ is a loss function, $\lambda \in \mathbb{R}^+$ is a regularisation parameter, Ω is the regulariser. Although a key bottleneck for applying such algorithms in the real-world is choosing λ, research often ignores this. As empirical results, however, strongly depend on the chosen λ, runtime intensive repeated cross-validations have to be performed. Hence, in this paper we concentrate on speeding up and automating this choice by building on the *regularisation path* for SVMs [3].

In fact, similar to SVM classification, it turns out that $\hat{f}^\lambda$ as a function of λ is piecewise linear and hence forms a *regularisation path*. The breakpoints of this path correspond to certain events. Points of the regularisation path which are not breakpoints can not be distinguished in terms of margin-errors of the training data. To choose a particular solution to the ranking problem, an evaluation of $\hat{f}^\lambda$ on a validation set is performed for each breakpoint of the path.

In Section 2 we describe the ranking SVM and in Section 3 its regularisation path. Experiments are shown in Section 4. Finally, Section 5 concludes.

2 Ranking SVM

In ranking problems like (i) and (ii) from the introduction, user preferences can be modeled by a (typically acyclic) digraph (V, E) with $E \subseteq V^2$. For (i) the set

of web pages forms the vertex set V of the digraph and we are also given some further information about the web pages (like a bag-of-words representation). For (ii) each vertex of the graph is a pair containing a query ($q \in \mathcal{Q}$) and a document ($d \in \mathcal{D}$). Hence, the vertex set is $V \subseteq \mathcal{Q} \times \mathcal{D}$ and edges of the form $((q,d),(q,d'))$ with $d, d' \in \mathcal{D}; q \in \mathcal{Q}$ represent that d was more relevant than d' for an user asking query q.

In both cases, the ranking algorithms aim to find an ordering (permutation) of the vertex $\pi : V \to [\![n]\!]$ where $n = |V|$ and $[\![n]\!] = \{1, \ldots, n\}$ such that similar documents are ranked as closely together as possible, while as few as possible preferences are violated by the permutation. Rank SVM approaches such learning problems by solving the following primal optimisation problem:

$$\begin{aligned} \hat{f}_\lambda \;=\; & \underset{f \in \mathcal{H}}{\operatorname{argmin}} \quad \xi^\top \mathbb{1} + \tfrac{\lambda}{2} \|f\|^2_{\mathcal{H}} \\ & \text{subject to:} \quad f(v) - f(u) \geq 1 - \xi_{vu} \quad \xi_{vu} \geq 0 \quad \forall (u,v) \in E. \end{aligned} \tag{1}$$

Here, $\mathcal{H} \subseteq \{h : V \to \mathbb{R}\}$ is a reproducing kernel Hilbert space (RKHS), $\lambda \in \mathbb{R}^+$ is a regularisation parameter, and the square norm $\|f\|^2_{\mathcal{H}}$ in the Hilbert space serves as the regulariser. The final permutation π is then obtained by sorting V according to f and resolving ties randomly. Now, let $k : V \times V \to \mathbb{R}$ be the reproducing kernel of $\mathcal{H}$ and denote the vertex by $\mathbf{x}_i$ such that $V = \{\mathbf{x}_i \mid i \in [\![n]\!]\}$. The set of violated constraints is $\{(\mathbf{x}_i, \mathbf{x}_j) \in E \mid \pi(\mathbf{x}_i) < \pi(\mathbf{x}_j)\}$.

Using $\hat{f}_\lambda(\mathbf{x}) = \sum_{i=1}^n \boldsymbol{\beta}_i k(\mathbf{x}_i, \mathbf{x})$ with $\boldsymbol{\beta}_i \in \mathbb{R}$, $i \in [\![n]\!]$. With slight abuse of notation we denote $\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), k(\mathbf{x}, \mathbf{x}_2), ..., k(\mathbf{x}, \mathbf{x}_n))^\top$ so that $f(\mathbf{x}_i) = \boldsymbol{\beta}^\top \mathbf{k}(\mathbf{x}_i)$ and $\|f\|^2_{\mathcal{H}} = \boldsymbol{\beta}^\top K \boldsymbol{\beta}$. Then Eq. (1) with m preferences $E = \{(\mathbf{x}_{k_i}, \mathbf{x}_{l_i}) \mid i \in [\![m]\!]\}$ becomes:

$$\begin{aligned} \hat{\boldsymbol{\beta}}(\lambda) = \; & \underset{\boldsymbol{\beta} \in \mathbb{R}^n, \boldsymbol{\xi} \in \mathbb{R}^m}{\operatorname{argmin}} \quad \boldsymbol{\xi}^\top \mathbb{1} + \tfrac{\lambda}{2} \boldsymbol{\beta}^\top K \boldsymbol{\beta} \\ & \text{s. t.} \quad \boldsymbol{\beta}^\top (\mathbf{k}(\mathbf{x}_{k_i}) - \mathbf{k}(\mathbf{x}_{l_i})) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i \in [\![m]\!] \end{aligned} \tag{2}$$

with $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. The Lagrangian $\mathcal{L}$ of this problem then becomes:

$$\mathcal{L} = \boldsymbol{\xi}^\top \mathbb{1} + \frac{\lambda}{2} \boldsymbol{\beta}^\top K \boldsymbol{\beta} - \sum_{i=1}^m \alpha_i \left(\boldsymbol{\beta}^\top \big(\mathbf{k}(\mathbf{x}_{k_i}) - \mathbf{k}(\mathbf{x}_{l_i}) \big) - 1 + \xi_i \right) - \sum_{i=1}^m \gamma_i \xi_i$$

with $\alpha_i \geq 0, \gamma_i \geq 0$. A matrix $P \in \mathbb{R}^{m \times n}$ can be defined with entries

$$P_{ij} = \begin{cases} +1 & \text{if } j = k_i \\ -1 & \text{if } j = l_i \\ 0 & \text{otherwise} \end{cases} \quad \Longrightarrow \quad PK = \begin{pmatrix} \mathbf{k}(\mathbf{x}_{k_1})^\top - \mathbf{k}(\mathbf{x}_{l_1})^\top \\ \mathbf{k}(\mathbf{x}_{k_2})^\top - \mathbf{k}(\mathbf{x}_{l_2})^\top \\ \vdots \\ \mathbf{k}(\mathbf{x}_{k_m})^\top - \mathbf{k}(\mathbf{x}_{l_m})^\top \end{pmatrix} \tag{3}$$

so that the Lagrangian can be expressed as:

$$\mathcal{L} = \boldsymbol{\xi}^\top \mathbb{1} + \frac{\lambda}{2} \boldsymbol{\beta}^\top K \boldsymbol{\beta} - \boldsymbol{\beta}^\top K P^\top \boldsymbol{\alpha} + \mathbb{1}^\top \boldsymbol{\alpha} - \boldsymbol{\xi}^\top \boldsymbol{\alpha} - \boldsymbol{\xi}^\top \boldsymbol{\gamma}$$

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

Using the KKT conditions, we obtain: $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\xi}} = 0 \Rightarrow 0 = \mathbb{1} - \boldsymbol{\alpha} - \boldsymbol{\gamma}$ and $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}} = 0 \Rightarrow 0 = \lambda K\boldsymbol{\beta} - KP^\top\boldsymbol{\alpha}$, giving $0 \leq \alpha_i \leq 1$ and $\boldsymbol{\beta} = \frac{1}{\lambda}P^\top\boldsymbol{\alpha}$ such that

$$f(\mathbf{x}) = \frac{1}{\lambda}\boldsymbol{\alpha}^\top P\mathbf{k}(\mathbf{x}). \tag{4}$$

Finally, the dual of Problem (2) is a QP problem:

$$\begin{array}{rcll} \hat{\boldsymbol{\alpha}}(\lambda) & = & \underset{\boldsymbol{\alpha}\in\mathbb{R}^n}{\operatorname{argmax}} & \boldsymbol{\alpha}^\top\mathbb{1} - \frac{1}{2\lambda}\boldsymbol{\alpha}^\top PKP^\top\boldsymbol{\alpha} \\ & & \text{s.t.} & \mathbf{0} \leq \boldsymbol{\alpha} \leq \mathbb{1}. \end{array} \tag{5}$$

3 Regularisation Path for Ranking SVM

Following Hastie's work [3] we now derive the regularisation path of ranking SVM. According to [4], the solution $\hat{\boldsymbol{\alpha}}(\lambda)$ of the above problem is a piecewise linear function of λ. For given λ let $f^\lambda(\mathbf{x})$ be the decision function corresponding to $\hat{\boldsymbol{\alpha}}(\lambda)$. Then, the following partition derived from the KKT optimality conditions can be made:

- $\mathcal{I}_\alpha^\lambda = \{i \in [\![m]\!] \mid f^\lambda(\mathbf{x}_{k_i}) - f^\lambda(\mathbf{x}_{l_i}) = 1\} = \{i \in [\![m]\!] \mid 0 < \hat{\boldsymbol{\alpha}}_i(\lambda) < 1\}$,
- $\mathcal{I}_0^\lambda = \{i \in [\![m]\!] \mid f^\lambda(\mathbf{x}_{k_i}) - f^\lambda(\mathbf{x}_{l_i}) > 1\} = \{i \in [\![m]\!] \mid \hat{\boldsymbol{\alpha}}_i(\lambda) = 0\}$, and
- $\mathcal{I}_1^\lambda = \{(i \in [\![m]\!] \mid f^\lambda(\mathbf{x}_{k_i}) - f^\lambda(\mathbf{x}_{l_i}) < 1\} = \{i \in [\![m]\!] \mid \hat{\boldsymbol{\alpha}}_i(\lambda) = 1\}$.

We choose $\lambda^1 > \lambda^2 > \ldots$ such that the above sets remain unchanged for all $\lambda \in (\lambda^{t+1}, \lambda^t]$ and denote $\boldsymbol{\alpha}^t = \hat{\boldsymbol{\alpha}}(\lambda^t)$, $f^t = f^{\lambda^t}$, as well as $(\mathcal{I}_\alpha^t, \mathcal{I}_1^t, \mathcal{I}_0^t) = (\mathcal{I}_\alpha^{\lambda^t}, \mathcal{I}_1^{\lambda^t}, \mathcal{I}_0^{\lambda^t})$. For other λ we suppress the dependence of f and $\boldsymbol{\alpha}$ on λ. Then $\hat{\boldsymbol{\alpha}}(\lambda)$ for $\lambda \in (\lambda^{t+1}, \lambda^t)$ depends linearly on λ as:

$$\begin{aligned} f(\mathbf{x}) &= \left[f(\mathbf{x}) - \frac{\lambda^t}{\lambda}f^t(\mathbf{x})\right] + \frac{\lambda^t}{\lambda}f^t(\mathbf{x}) = \frac{1}{\lambda}\Big[(\boldsymbol{\alpha} - \boldsymbol{\alpha}^t)^\top P\mathbf{k}(\mathbf{x}) + \lambda^t f^t(\mathbf{x})\Big] \\ f(\mathbf{x}) &= \frac{1}{\lambda}\Big[(\boldsymbol{\alpha}_{\mathcal{I}_\alpha} - \boldsymbol{\alpha}_{\mathcal{I}_\alpha}^t)^\top P_{\mathcal{I}_\alpha}\mathbf{k}(\mathbf{x}) + \lambda^t f^t(\mathbf{x})\Big] \end{aligned} \tag{6}$$

with $P_{\mathcal{I}_\alpha}$ being the submatrix of P containing the rows corresponding to $\mathcal{I}_\alpha$ and all columns. The last line holds as $\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_i^t = 0$ for all $i \notin \mathcal{I}_\alpha$. As all sets remain fixed for $\lambda \in (\lambda^{t+1}, \lambda^t)$, we also have that $1 = f(\mathbf{x}_{k_i}) - f(\mathbf{x}_{l_i}) = f^t(\mathbf{x}_{k_i}) - f^t(\mathbf{x}_{l_i})$ for all $i \in \mathcal{I}_\alpha$, so Eq. (4) leads to

$$\lambda - \lambda^t = \left(\boldsymbol{\alpha}_{\mathcal{I}_\alpha} - \boldsymbol{\alpha}_{\mathcal{I}_\alpha}^t\right)^\top P_{\mathcal{I}_\alpha}\left(\mathbf{k}(\mathbf{x}_{k_i}) - \mathbf{k}(\mathbf{x}_{l_i})\right) \quad \forall i \in \mathcal{I}_\alpha. \tag{7}$$

The latter set of equations can be simplified by using Eq. (3) to obtain:

$$(\lambda - \lambda^t)\mathbb{1}_{\mathcal{I}_\alpha} = P_{\mathcal{I}_\alpha}KP_{\mathcal{I}_\alpha}^\top\left(\boldsymbol{\alpha}_{\mathcal{I}_\alpha} - \boldsymbol{\alpha}_{\mathcal{I}_\alpha}^t\right) \tag{8}$$

If we define $\boldsymbol{\eta} = (P_{\mathcal{I}_\alpha}KP_{\mathcal{I}_\alpha}^\top)^{-1}\mathbb{1}_{\mathcal{I}_\alpha}$, with $\mathbb{1}_{\mathcal{I}_\alpha}$ a vector of ones of size $|\mathcal{I}_\alpha|$, then it can finally be seen that $\boldsymbol{\alpha}_i, i \in \mathcal{I}_\alpha$ changes piecewise linear in λ as follows:

$$\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_i^t - (\lambda^t - \lambda)\eta_i \qquad i \in \mathcal{I}_\alpha. \tag{9}$$

For all $\lambda \in (\lambda^{t+1}, \lambda^t)$, the optimal solution $\boldsymbol{\alpha}$ can be easily obtained until the sets change, i.e., an event occurs. From any given optimal solution $\boldsymbol{\alpha}^t$ for λ^t, the corresponding sets $\mathcal{I}_\alpha^t, \mathcal{I}_0^t$, and $\mathcal{I}_1^t$ can be deduced and thereon the corresponding λ^{t+1} that generates an event together with the optimal solution.

3.1 Initialisation

If λ is very large, $\boldsymbol{\beta} = \mathbf{0}$ minimises Problem (2) and $\mathcal{I}_1 = [\![M]\!], \mathcal{I}_1 = \mathcal{I}_0 = \emptyset$. This implies that $\xi_i = 1$ and because of the strict complementary and KKT conditions, $\gamma_i = 0 \Rightarrow \boldsymbol{\alpha}_i = 1$. To have at least one element in $\mathcal{I}_\alpha$, we need a pair $(\mathbf{x}_{k_i}, \mathbf{x}_{l_i})$ for which $\boldsymbol{\beta}^\top(\mathbf{k}(\mathbf{x}_{k_i}) - \mathbf{k}(\mathbf{x}_{l_i})) = 1$. As $\frac{1}{\lambda} P^\top \boldsymbol{\alpha} = \boldsymbol{\beta} = \mathbf{0}$ we get $\boldsymbol{\alpha} = \mathbb{1}$ and define $\lambda_i = \boldsymbol{\alpha}^\top P(\mathbf{k}(\mathbf{x}_{k_i}) - \mathbf{k}(\mathbf{x}_{l_i}))$. Now $\lambda^0 = \max\{\lambda_i \mid i \in [\![m]\!]\}$, $\mathcal{I}_\alpha = \arg\max_i\{\lambda_i\}$ and $\mathcal{I}_1 = [\![m]\!] \setminus \arg\max_i\{\lambda_i\}$.

3.2 Event Detection

At step t the optimal solution $\boldsymbol{\alpha}^t$ defines a partition $\mathcal{I}_\alpha, \mathcal{I}_1, \mathcal{I}_0$. If these sets remain fixed for all λ in a given range then the optimal solution $\boldsymbol{\alpha}(\lambda)$ is a linear function of $\boldsymbol{\alpha}^t$. If an event occurs, i.e., the sets change, then the linear equation has to be readjusted. Two types of events have to be determined: *a)* a pair in $\mathcal{I}_\alpha$ goes to $\mathcal{I}_1$ or $\mathcal{I}_0$ and *b)* a pair in $\mathcal{I}_1$ or $\mathcal{I}_0$ goes to $\mathcal{I}_\alpha$.

3.2.1 Pair in $\mathcal{I}_\alpha$ goes to $\mathcal{I}_1$ or $\mathcal{I}_0$

This event can be determined by analysing at which value of λ the corresponding α_i turns zero or one. Eq. (9) is used and the following systems are solved for λ_i:

$$1 = \alpha_i^t - (\lambda^t - \lambda_i)\eta_i \qquad i \in \mathcal{I}_\alpha \tag{10}$$

$$0 = \alpha_i^t - (\lambda^t - \lambda_i)\eta_i \qquad i \in \mathcal{I}_\alpha. \tag{11}$$

Using this last equation, the exact values for λ_i that produce an event on pairs in $\mathcal{I}_\alpha$ moving to $\mathcal{I}_0 \cup \mathcal{I}_1$ can be determined.

3.2.2 Pair in $\mathcal{I}_1$ or $\mathcal{I}_0$ goes to $\mathcal{I}_\alpha$

To detect this event, note that Equation (8) can also be written as follows:

$$\left(\boldsymbol{\alpha}_{\mathcal{I}_\alpha} - \boldsymbol{\alpha}_{\mathcal{I}_\alpha}^t\right) = (\lambda - \lambda^t)\left[\left(P_{\mathcal{I}_\alpha} K P_{\mathcal{I}_\alpha}^\top\right)^{-1} \mathbb{1}_{\mathcal{I}_\alpha}\right] = (\lambda - \lambda^t)\boldsymbol{\eta}. \tag{12}$$

Plugging Eq. (12) in Eq. (6), we can write $f(\mathbf{x})$ in a convenient manner:

$$f(\mathbf{x}) = \frac{1}{\lambda}\left[\lambda^t f^t(\mathbf{x}) + \left(\lambda - \lambda^t\right) h^t(\mathbf{x})\right] \text{ with } h^t(\mathbf{x}) = \boldsymbol{\eta}^\top P_{\mathcal{I}_\alpha} \mathbf{k}(\mathbf{x}). \tag{13}$$

An event on pair $(\mathbf{x}_{k_i}, \mathbf{x}_{l_i}) \in \mathcal{I}_0 \cup \mathcal{I}_1 \longrightarrow \mathcal{I}_\alpha$ means that $f(\mathbf{x}_{k_i}) - f(\mathbf{x}_{l_i}) = 1$ and can be detected by using Equation (3.2.2). The corresponding λ_i that generates this event is calculated as follows:

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

$$\lambda_i = \frac{\lambda^t \left[(f(\mathbf{x}_{k_i}) - f(\mathbf{x}_{l_i})) - (h^t(\mathbf{x}_{k_i}) - h^t(\mathbf{x}_{l_i})) \right]}{1 - (h^t(\mathbf{x}_{k_i}) - h^t(\mathbf{x}_{l_i}))} \tag{14}$$

λ^{t+1} will be the largest resulting $\lambda_i < \lambda^t$ from Eqs. (10), (11) and (14).

Experimentally, we observed that the validation error is typically lower at the break points rather than between them.

The numerical complexity of the algorithm depends on the number of iterations needed to explore the overall solution path and the mean size of $\mathcal{I}_\alpha$. At each iteration, a linear system is solved to get $\boldsymbol{\eta}$ which has complexity $O(|\mathcal{I}_\alpha|^2)$. Empirically we observed that the number of iterations is typically only 2-3 times larger than the number of training pairs

In SVM methods, another key point is the determination of kernel hyperparameter. This problem was not tackled here. However, one can seek to combine our regularisation path with the kernel parameter path developed in [5].

4 Experimental Results

Three datasets were used to test the algorithm. A toy example generated from Gaussian distributions and two more datasets taken from the UCI datasets[1].

The toy dataset [3] was originally designed for binary classification with instances $\mathbf{x}_i$ and corresponding labels $y_i \in \{\pm 1\}$. It can, however be also viewed as a ranking problem with $E = \{(\mathbf{x}_i, \mathbf{x}_j) \mid y_i > y_j\}$. It contains 100 positive and 100 negative points which induce 10000 constraints. The other two datasets have regression problems and can also be viewed as ranking problems by letting $E = \{(\mathbf{x}_i, \mathbf{x}_j) \mid y_i > y_j\}$. Three measures were used: The number of wrong classified pairs, the NDCG measure [6], and the Kendall rank correlation coefficient.

The experimental design is as follows: $\frac{1}{5}$ of the original dataset was randomly taken to form a test set. From the remaining data points a random sample of size 100 was drawn. On this subsample of the training data, the kernel parameter is chosen by cross validation. Using this kernel parameter, 5-fold cross validation in the training data was used to choose the vertex of the regularisation path minimising the validation error. Test error of this model was measured on the test set. Results are summarized in the following tables.

Dataset	# Training pairs	# Features	σ	λ^* value	Size of $\mathcal{A}$
Mixture	10000	200	0.5	7.45	126
Auto	75245	392	16	0.0033	248
Housing	127137	506	5.67	0.0025	320

Table 1: Data and result summary

[1] http://archive.ics.uci.edu/ml/datasets.html

Dataset	Misclassified pairs	Percentage	NDCG	Kendall
Mixture	13	5.42	1	-0.08
Auto	209	10.87	0.4341	0.755
Housing	221	6.87	0.8784	0.85

Table 2: Test error

5 Conclusions

The proposed approach calculates efficiently the regularisation path of the ranking SVM by solving small linear problems. Then, the regularisation parameter can efficiently be chosen as the vertex of the regularisation path that minimises the validation error. The computational complexity is highly related to the total number of breakpoints on the path and the mean number of support vectors. In our experiments, we have seen that the latter number is generally low and that the former number is 2-3 times the size of the problem. A possible extension of this work is the efficient combination of our regularisation path and the kernel parameter path [5].

References

[1] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. *Large margin rank boundaries for ordinal regression*. MIT Press, Cambridge, MA, 2000.

[2] T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 133–142, 2002.

[3] Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5:1391–1415, October 2004.

[4] Saharon Rosset and Ji Zhu. Piecewise linear regularized solution paths. *Annals of Statistics*, 35(3):1012–1030, 2007.

[5] Dit-Yan Yeung Gang Wang and Frederick H. Lochovsky. A kernel path algorithm for support vector machines. In *Proceedings of ICML'2007*, 2007.

[6] Stephen Robertson and Hugo Zaragoza. On rank-based effectiveness measures and optimization. *Inf. Retr.*, 10(3):321–339, 2007.