AUC Maximization with K-hyperplane

Majdi Khalid*, Indrakshi Ray, and Hamidreza Chitsaz
Colorado State University, Fort Collins, USA
majdi.khaled@gmail.com, indrakshi.ray@colostate.edu, chitsaz@chitsazlab.org
www.colostate.edu

Abstract. Area Under the ROC Curve (AUC) is a reliable metric for measuring the quality of the classification performance on imbalanced data. The existing pairwise learn to rank linear algorithms can optimize the AUC metric efficiently, but lack modeling the nonlinearity underlying the data. The large scale and nonlinear distribution of many real world data render the kernelized variants of these linear classifiers impractical. In this paper, we propose a linear K-hyperplane classifier for AUC maximization. The proposed method employs a set of linear pairwise classifiers in order to approximate the nonlinear decision boundary. More specifically, we define a set of landmark points using unsupervised algorithm, and associate each landmark with a linear pairwise classifier. These linear classifiers are learned jointly on pairs of instances where each instance in the pair is weighted based on its distance from the corresponding landmark point. We use a local coding algorithm to estimate the weights of the instances. Experiments on several benchmark datasets demonstrate that our classifier is efficient and able to achieve roughly comparable AUC classification performance as compared to kernelized pairwise classifier. The proposed classifier also outperforms the linear RankSVM classifiers in terms of AUC performance.

1 Introduction

Area Under the ROC Curve \cite{9} is a reliable metric for measuring the quality of the classification performance on imbalanced data \cite{5}. The learning algorithms that optimize the AUC metric, a.k.a. bipartite ranking, have a broad range of applications in machine learning and data mining such as recommender systems, information retrieval, online advertisements and bioinformatics \cite{19, 4, 22}. The optimization of the AUC aims to learning a score function that scores a random positive instance higher than any negative instance. Therefore, the objective function maximizing the AUC metric optimizes a sum of pairwise losses. The pairwise objective function can be solved by learning a binary classifier on pairs of positive and negative instances that constitute the difference space. Intuitively, the complexity of such algorithms increases linearly with respect to the number of pairs \cite{3}.

* corresponding author
Many studies have devoted to develop pairwise ranking classifiers that can optimize the AUC metric \cite{10,12,23,1,18,16}. Recent methods \cite{31,15,7,11,14,29} propose linear and nonlinear online learning to address the complexity of learning the AUC objective function. Despite these methods have shown superior efficiency, their AUC performances are inferior to batch AUC algorithms. The optimization of the AUC metric by linear classifiers \cite{23,18} such as linear support vector machines has demonstrated to be practical as linear classifiers can efficiently scale up to large datasets. Unfortunately, the capability of linear classifiers lacks modeling complex decision boundary. This problem can be addressed by learning a classifier on the data mapped implicitly into a high dimensional space using kernel trick \cite{25}. However, the kernelized support vector machines have a complexity that grows quadratically in the number of instances. The kernel RankSVM \cite{16} is developed to reduce the complexity of learning nonlinear rank SVM. However, this method is still very expensive to train compared to linear RankSVM. In SVM literatures, several methods are developed to scale up kernel classifiers by approximating the function or the matrix of the nonlinear kernel. One approach approximates the nonlinear kernel using explicit feature maps (e.g., Random Fourier Features and Nyström approximation) \cite{28,21,26}, then learning a linear classifier on this feature space. However, the algorithms in this approach approximate the nonlinear kernel with no knowledge about the task at hand as they ignore the class labels of the data in the approximation steps. Also, these methods can achieve a reasonable accuracy only when generating a high dimensional space, which in turns increasing the computational and the storage costs. A recent approach \cite{6} integrates the classification task with learning the nonlinear maps, which results in a low nonlinear feature space. However, this method is restricted to shift-invariant kernels.

Another approaches tackle the nonlinearity by learning a set of classifiers that exploit the underlying structure of the data \cite{17,5,13}, but non of these methods have used to maximize the AUC metric. For imbalanced learning, the work \cite{20} deals with nonlinearity by learning K-hyperplane classifier. However, this method is not explicitly optimizing the AUC metric.

In this paper, we propose a method that learns a set of linear hyperplanes that maximize the AUC metric. In particular, we employ multiple linear pairwise classifiers in order to approximate a smooth nonlinear decision boundary. First, we identify $K$ landmark points on the input space using an unsupervised algorithm. Each landmark point is associated with a linear classifier. We then jointly train a linear classifier for each landmark point on the pairs, where each data point in the pair is weighted based on its distance from the corresponding landmark point. This distance coefficient is computed using a fast coding algorithm proposed in \cite{27}. This learning scheme is influenced by the locally linear SVM (LL-SVM) \cite{17}. However, our proposed algorithm tackles the problem of maximizing the AUC metric whereas the LL-SVM optimizes the error rate.
2 AUC Maximization with K-hyperplane

2.1 Problem Definition

Given a sample $S = \{x^+_i \cup x^-_j \in \mathcal{R}^d | i = \{1, \ldots, n\}, j = \{1, \ldots, m\}\}$ denotes the input space of dimension $d$ generated from unknown distribution $\mathcal{D}$, where $x^+_i$ is the i-th positive instance and $x^-_j$ is the j-th negative instance. The $n$ and $m$ denote the number of positive and negative instances, respectively. The maximization of the AUC metric is equivalent to the minimization of the following loss function:

$$L(f; S) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} I(f(x^+_i) \leq f(x^-_j)), \quad (1)$$

For a linear classifier $f(x) = w^T x$, where $I(\cdot)$ is the indicator function that outputs 1 if the condition is held, and 0 otherwise. It is common to replace the indicator function with a convex surrogate function,

$$L(f; S) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \ell(f(x^+_i) - f(x^-_j))^p, \quad (2)$$

when $p = 1$ the loss function $\ell(z) = \max(0, 1 - z)$ is the hinge loss function, and with $p = 2$ the loss function $\ell(z) = \max(0, 1 - z)^2$ is the squared hinge loss function. Therefore, the optimal linear pairwise classifier $w$ can be obtained by minimizing the following objective function:

$$\arg\min_w \frac{1}{2} ||w||^2 + C \sum_{i=1}^{n} \sum_{j=1}^{m} \max(0, 1 - w^T (x^+_i - x^-_j))^p, \quad (3)$$

where $||w||$ is the euclidean norm and $C$ is the regularization hyperparameter. Notice that the weight vector $w$ is trained on the pairs of instances $x^+_i - x^-_j$ that form the difference space. This linear pairwise classifier is efficient in dealing with large scale applications, but its modeling capability is limited to linear decision boundary.

2.2 Linear Coding

Given a set of $K$ landmark points $K = [l_1, \ldots, l_{|K|}] \in \mathcal{R}^{d \times |K|}$ generated from the input space using an unsupervised algorithm, the coding methods approximate an instance as a linear combination of landmarks,

$$x_i \approx \sum_{j=1}^{|K|} v_{ij} l_j, \quad (4)$$
the coefficients $v_i$ are the code for $x_i$, where $v_{ij}$ estimates the proximity of the $i$-th instance to the $j$-th landmark point $l_j$. We use the Locality-constrained Linear Coding (LLC) to evaluate the codes $V$. The LLC coding algorithm assesses the coefficients $V \in \mathbb{R}^{(m+n) \times |K|}$ by solving a least square fitting with equality constraints [27],

$$\min _{V} \sum _{i=1}^{m+n} ||x_i - K v_i||^2 + \lambda ||D_i \odot v_i||^2 \quad s.t. \quad 1^T v_i = 1, \forall i$$  \hspace{1cm} (5)$$

where $\odot$ symbolizes the element-wise multiplication, $D_i = \exp(\text{dist}(x_i, K)/\sigma)$ and $\text{dist}(x_i, K) = [\text{dist}(x_i, l_1), \ldots, \text{dist}(x_i, l_{|K|})]^T$. The $\text{dist}(x, l_j)$ is the euclidean distance between the data point $x$ and the $j$-th landmark point. Notice that the L2-norm regularization in (5) yields a small but not sparse coefficients. This is required in our method as both instances forming a pair must belong to the same landmarks.

The linear coding has it is advantage in approximating a smooth nonlinear function. As demonstrated in [30], a linear combination of function values $f(l)$ defined over a set of landmarks (or anchor points) can approximate any Lipschitz smooth function $f(x)$ as,

$$f(x) \approx \sum _{j=1}^{|K|} v_{ij} f(l_j)$$  \hspace{1cm} (6)$$

2.3 The Proposed Model

We exploit the concept of linear coding with pairwise loss function to approximate the nonlinearity underlying the difference space. Therefore, our method addresses the nonlinearity using a set of linear classifiers trained on the pairs of weighted instances. We use the linear coding scheme to estimate the weights of each instance with respect to the landmarks.

First we define $K$ landmark points from the input space using k-means unsupervised algorithm, and designate a linear classifier for each landmark point. Then we estimate the coefficients of each instance with respect to $K$ landmarks using the LLC coding algorithm. Notice that the LLC generates dense coefficients. The set of linear classifiers will then be trained on the weighted pairs.

The objective function we seek to minimize with respect to the linear pairwise K-hyperplane classifier is formulated as follows,

$$\arg\min _{w} \frac{1}{2} \sum _{r=1}^{K} w_r^T w_r + C \sum _{i=1}^{n} \sum _{j=1}^{m} \max (0, 1 - \sum _{r=1}^{K} w_r^T (v_{ir} x_i^+ - v_{jr} x_j^-))^p$$  \hspace{1cm} (7)$$

where $v_{ir}$ is the coefficient of the instance $x_i$ with respect to the landmark point $l_r$. The score function for a new instance $x$ is defined as follows,
AUC Maximization with K-hyperplane

\[ f(x) = \sum_{j=1}^{[K]} w_j^T(v_j x), \quad (8) \]

where the coefficient vector \( v \) is estimated using the coding algorithm with respect to the landmarks generated from the training data.

We can observe that the input space can be transformed into a finite high dimensional space of size \( d \times |K| \). Therefore, we can make the optimization problem more applicable to the off-the-shelf linear RankSVM tools by redefining each instance and the weight vector as,

\[ \tilde{x}_i = [v_1 x_i^T, v_2 x_i^T, \ldots, v_K x_i^T]^T \quad (9) \]
\[ \tilde{w} = [w_1^T, w_2^T, \ldots, w_K^T]^T. \quad (10) \]

Therefore, the optimization problem \( 7 \) can be reformulated as follows,

\[ \arg\min_{\tilde{w}} \frac{1}{2} \|\tilde{w}\|^2 + C \sum_{i=1}^{n} \sum_{j=1}^{m} \max(0, 1 - \tilde{w}_v (\tilde{x}_i^+ - \tilde{x}_j^-))^p, \quad (11) \]

The optimization problem \( 11 \) with hinge loss can be solved using Pegasus [24] and SVMSGD2 [2], with complexity independent of the number of pairs. The linear rankSVM methods [3,18] can be used to solve the squared hinge loss, with complexity \( O((n + m)dK + P) \) or even better with LAUCT [18], where \( P \) is the number of pairs, which at most equals \( O(|S|^2) \). The complexity of generating the landmark points using k-means is also linear to the number of landmarks, and estimating the coefficients for each instance requires \( O(|K| + |K|) \) time complexity. Therefore, the proposed method is feasible to scale to large datasets.

## 3 Experiments

In this section, we conduct experiments on six benchmark datasets in order to evaluate the proposed method and compare its AUC performance and training time with other linear and nonlinear pairwise classifiers. We solved the proposed objective function \( 11 \) using Linear RankSVM-Tron (RSVMT) [18] and Primal RankSVM (PRSVM) [3] where both solvers optimize the L2-loss function.

### 3.1 Real World Datasets

The datasets we use in our experiments can be downloaded from LibSVM website[^1] and UCI[^2]. All these datasets are provided as training and test sets except

[^1]: https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/
[^2]: http://archive.ics.uci.edu/ml/index.php
For those data that are not split, we randomly divide each dataset into 80%-20% for training and test. The multi-class datasets usps, mnist, and covtype are converted into binary imbalanced data. We transform each digits dataset into binary imbalanced data by giving a class label to the digits from one to six while the rest of digits are given the another class. For covtype, we transform the classification task to distinguish between label one and label seven. In order to speed up the training step for large training datasets, we randomly select 25k instances for training. Table 1 shows the characteristics of the datasets along with their imbalance ratio. The features of these datasets are standardized to have zero mean and unit variance.

### Table 1: Benchmark datasets

| Data    | #training | #test  | #feat | ratio |
|---------|-----------|--------|-------|-------|
| usps    | 7,291     | 2,007  | 256   | 1.94  |
| magic04 | 15,216    | 3,804  | 10    | 1.84  |
| ijcn1   | 49,990    | 91,701 | 22    | 9.45  |
| mnist   | 60,000    | 10,000 | 784   | 2.18  |
| skin    | 171,539   | 73,518 | 3     | 3.82  |
| covtype | 185,825   | 46,525 | 54    | 10.30 |

### 3.2 Compared Methods and Model Selection

**RBF-RankSVM** [10]: This is the nonlinear kernel ranking SVM. We use Gaussian Kernel $K(x, y) = \exp(-\gamma \|x - y\|^2)$. The best width of the kernel $\gamma$ is chosen by 3-fold cross validation on the training set by searching in $\{10^{-2:2}\}$. The regularization parameter $C$ is also tuned by 3-fold cross validation by searching in the grid $\{10^{-4:2}\}$.

**Linear RankSVM-Tron (RSVMT)** [18]: The linear RankSVM that optimize the squared hinge loss function using a trust region Newton method with selection trees that aims to reduce the training complexity. The best regularization parameter $C$ is chosen from the grid $\{10^{-8:2}\}$ via 3-fold cross validation.

**Primal RankSVM (PR SVM)** [3]: The linear RankSVM that optimize the squared hinge loss function using truncated Newton. The regularization parameter $C$ is tuned in similar way as in RSVMT.

**Linear K-hyperplane AUC Maximization**: This is our proposed method that learns multiple pairwise classifiers. We refer to this proposed method as LAUCT when it is solved by Linear RankSVM-Tron and LAUC when it is solved by the Primal RankSVM. For each dataset we generate six landmarks points using k-means algorithm. The coefficients of the instances with respect to the landmarks are obtained using the linear coding algorithm [27]. The regularization parameter $C$ is tuned by 3-fold cross validation by searching in the same grid used with RSVMT.
As the proposed method involves a random step to generate the landmarks, we report the mean of the AUC and the training time of three runs.

### 3.3 Results on Benchmark Datasets

The AUC performance on the test set is shown in Table 2, while Table 3 shows the training time of all methods.

The proposed LAUCT and LAUC are assessed with six pairwise hyperplanes (i.e., six landmarks are generated). The LAUCT and LAUC show a similar AUC performance on most datasets. We observed that both LAUCT and LAUC outperform the RSVMT and the PRSVM, which learn a single pairwise classifier, in terms of AUC performance, while both LAUCT and LAUC show a trivial increase in the training time. This because the LAUCT and LAUC are able to generalize better by modeling the nonlinearity. This renders our method to be very appealing for large-scale data. We should mention that the LAUCT is implemented in C language while the LAUC is implemented in Matlab.

Moreover, the AUC performance of LAUCT and LAUC show a moderate loss on some datasets compared to RBF-RankSVM. However, both LAUCT and LAUC demonstrate a significant advantage over RBF-RankSVM in terms of training time. Consequently, the LAUCT and LAUC will be able to surpass RBF-RankSVM in terms of testing time as well.

The AUC performances of the RSVMT and PRSVM are comparable on almost all datasets. The difference in their training times is because the PRSVM is written in matlab while the RSVMT is implemented in the C language. Also, we can see that the RSVMT and PRSVM achieve a low AUC compared to the RBF-RankSVM. This because they lack modeling the nonlinearity underlying the data. However, they demonstrate a low computational complexity compared to the RBF-RankSVM.

We also study the effect of the number of hyperplanes on the classification performance of the proposed LAUCT and LAUC algorithms. We repeat the experiments in the previous subsection with different number of hyperplanes based on the grid \( \{2, 4, 6, 8, 10, 12, 14\} \). Figure 1 illustrates the AUC on the test set as a function of different number of hyperplanes, while Figure 2 depicts the training time including the computation of the landmarks. We can see that the classification performance is improved with respect to increasing the number of hyperplanes. Further, we can observe that the classification performance is not widely fluctuating when varying the number of hyperplanes. This indicates that performing a cross validation to select the best number of hyperplanes will be reliable. We can also notice a slight loss in the classification performance on ijcnn1 dataset when increasing the number of hyperplanes. This because large number of classifiers could potentially overfit the training data. In terms of training time, both LAUCT and LAUC shows a negligible increase in the training complexity compared to their counterparts that train a single classifier. On usps and covtype, the training complexity of the LAUC is lower than or on the par with PRSVM. This might be attributed to the fast convergence of LAUC.
Table 2: Comparison of AUC performance of pairwise classifiers on benchmark datasets. The LAUCT and LAUC are evaluated with 6 hyperplanes

| Data | RBF-RankSVM | RSVMT | PRSVM | LAUCT | LAUC |
|------|-------------|-------|-------|-------|------|
| usps | 100.00      | 94.13 | 92.38 | 98.30 | 98.84 |
| magic04 | 91.47      | 84.48 | 83.77 | 90.68 | 89.91 |
| icmcm1 | 99.6       | 93.26 | 91.56 | 97.09 | 96.76 |
| mnist | 99.91       | 93.60 | 95.96 | 95.02 | 98.39 |
| skin | 99.99       | 94.62 | 94.84 | 99.97 | 99.93 |
| covtype | 99.07      | 95.49 | 95.52 | 97.01 | 97.31 |

Table 3: Comparison of training time (in seconds) of pairwise classifiers on benchmark datasets. The proposed LAUCT and LAUC are evaluated with 6 hyperplanes

| Data | RBF-RankSVM | RSVMT | PRSVM | LAUCT | LAUC |
|------|-------------|-------|-------|-------|------|
| usps | 20.16       | 0.81  | 25.03 | 4.99  | 22.04 |
| magic04 | 599.84     | 0.05  | 110.83 | 0.87  | 134.25 |
| icmcm1 | 4486.3     | 0.14  | 90.40 | 2.93  | 103.87 |
| mnist | 985.83      | 5.34  | 300.30 | 211.73 | 330.16 |
| skin | 288.56      | 0.04  | 158.90 | 0.61  | 189.93 |
| covtype | 7811.85 | 0.42  | 110.18 | 6.47  | 111.98 |

4 Conclusions and Future Work

In this paper, we have proposed a linear pairwise K-hyperplane classifier that maximize the AUC metric. The proposed method defines a set of landmarks using unsupervised algorithm and associates each landmark with a linear pairwise classifier. These linear classifiers are intended to approximate nonlinearity by learning on instances weighted based on their distances from the corresponding landmarks. The weights of instances is computed using a fast coding algorithm. We show that our method can be readily integrated with off-the shelf linear RankSVM tools. The experimental results show that our method achieves an AUC performance close to the nonlinear kernel RankSVM, while being more efficient. Also, we experimentally demonstrate that our method significantly improve upon linear pairwise classifier in terms of AUC performance with insignificant increase in training time. As a future work, we will investigate the integration of linear coding with online AUC optimization methods in order to improve their classification performance.

References

1. Airola, A., Pahikkala, T., Salakoski, T.: Training linear ranking svms in linearithmic time using red–black trees. Pattern Recognition Letters 32(9), 1328–1336 (2011)
Fig. 1: AUC performance of LAUCT and LAUC with different number of hyperplanes (K)

Fig. 2: Training time (in seconds) of LAUCT and LAUC with different number of hyperplanes (K)
2. Bordes, A., Bottou, L., Gallinari, P.: Sgd-qn: Careful quasi-newton stochastic gradient descent. Journal of Machine Learning Research 10(Jul), 1737–1754 (2009)
3. Chapelle, O., Keerthi, S.S.: Efficient algorithms for ranking with svms. Information Retrieval 13(3), 201–215 (2010)
4. Chaudhuri, S., Theocharous, G., Ghavamzadeh, M.: Recommending advertisements using ranking functions (Jan 18 2016), uS Patent App. 14/997,987
5. Cortes, C., Mohri, M.: Auc optimization vs. error rate minimization. Advances in neural information processing systems 16(16), 313–320 (2004)
6. Felix, X.Y., Kumar, S., Rowley, H., Chang, S.F.: Compact nonlinear maps and circulant extensions. stat 1050, 12 (2015)
7. Gao, W., Jin, R., Zhu, S., Zhou, Z.H.: One-pass auc optimization. In: ICML (3). pp. 906–914 (2013)
8. Gu, Q., Han, J.: Clustered support vector machines. In: Artificial Intelligence and Statistics. pp. 307–315 (2013)
9. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (roc) curve. Radiology 143(1), 29–36 (1982)
10. Herschtal, A., Raskutti, B.: Optimising area under the roc curve using gradient descent. In: Proceedings of the twenty-first international conference on Machine learning. p. 49. ACM (2004)
11. Hu, J., Yang, H., King, I., Lyu, M.R., So, A.M.C.: Kernelized online imbalanced learning with fixed budgets. In: AAAI. pp. 2666–2672 (2015)
12. Joachims, T.: A support vector method for multivariate performance measures. In: Proceedings of the 22nd international conference on Machine learning. pp. 377–384. ACM (2005)
13. Jose, C., Goyal, P., Aggrwal, P., Varma, M.: Local deep kernel learning for efficient non-linear svm prediction. In: International Conference on Machine Learning. pp. 486–494 (2013)
14. Khalid, M., Ray, I., Chitsaz, H.: Confidence-weighted bipartite ranking. In: Advanced Data Mining and Applications: 12th International Conference, ADMA 2016, Gold Coast, QLD, Australia, December 12-15, 2016, Proceedings 12. pp. 35–49. Springer (2016)
15. Kotlowski, W., Dembczynski, K.J., Huellermeier, E.: Bipartite ranking through minimization of univariate loss. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11). pp. 1113–1120 (2011)
16. Kuo, T.M., Lee, C.P., Lin, C.J.: Large-scale kernel ranksvm. In: Proceedings of the 2014 SIAM international conference on data mining. pp. 812–820. SIAM (2014)
17. Ladicky, L., Torr, P.: Locally linear support vector machines. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11). pp. 985–992 (2011)
18. Lee, C.P., Lin, C.J.: Large-scale linear ranksvm. Neural computation 26(4), 781–817 (2014)
19. Liu, T.Y.: Learning to rank for information retrieval. Foundations and Trends in Information Retrieval 3(3), 225–331 (2009)
20. Osadchy, M., Hazan, T., Keren, D.: K-hyperplane hinge-minimax classifier. In: Proceedings of the 32nd international conference on machine learning (ICML-15). pp. 1558–1566 (2015)
21. Rahimi, A., Recht, B.: Random features for large-scale kernel machines. In: Advances in neural information processing systems. pp. 1177–1184 (2008)
22. Rendle, S., Balby Marinho, L., Nanopoulos, A., Schmidt-Thieme, L.: Learning optimal ranking with tensor factorization for tag recommendation. In: Proceedings
AUC Maximization with K-hyperplane

of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 727–736. ACM (2009)

23. Sculley, D.: Large scale learning to rank. In: NIPS Workshop on Advances in Ranking. pp. 58–63 (2009)

24. Shalev-Shwartz, S., Singer, Y., Srebro, N., Cotter, A.: Pegasos: Primal estimated sub-gradient solver for svm. Mathematical programming 127(1), 3–30 (2011)

25. Smola, A.J., Schölkopf, B.: Learning with kernels. GMD-Forschungszentrum Informationstechnik (1998)

26. Vedaldi, A., Zisserman, A.: Efficient additive kernels via explicit feature maps. IEEE transactions on pattern analysis and machine intelligence 34(3), 480–492 (2012)

27. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. pp. 3360–3367. IEEE (2010)

28. Williams, C.K., Seeger, M.: Using the nyström method to speed up kernel machines. In: Advances in neural information processing systems. pp. 682–688 (2001)

29. Ying, Y., Wen, L., Lyu, S.: Stochastic online auc maximization. In: Advances in Neural Information Processing Systems. pp. 451–459 (2016)

30. Yu, K., Zhang, T., Gong, Y.: Nonlinear learning using local coordinate coding. In: Advances in neural information processing systems. pp. 2223–2231 (2009)

31. Zhao, P., Jin, R., Yang, T., Hoi, S.C.: Online auc maximization. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11). pp. 233–240 (2011)