RESEARCH ARTICLE

Improved linear classifier model with Nyström

Changming Zhu*, Xiang Ji, Chao Chen, Rigui Zhou, Lai Wei, Xiafen Zhang

College of Information Engineering, Shanghai Maritime University, Shanghai, China

* cmzhu@shmtu.edu.cn

Abstract

Most data sets consist of interlaced-distributed samples from multiple classes and since these samples always cannot be classified correctly by a linear hyperplane, so we name them nonlinearly separable data sets and corresponding classifiers are named nonlinear classifiers. Traditional nonlinear classifiers adopt kernel functions to generate kernel matrices and then get optimal classifier parameters with the solution of these matrices. But computing and storing kernel matrices brings high computational and space complexities. Since INMKMHKS adopts Nyström approximation technique and NysCK changes nonlinearly separable data to linearly ones so as to reduce the complexities, we combines ideas of them to develop an improved NysCK (INysCK). Moreover, we extend INysCK into multi-view applications and propose multi-view INysCK (MINysCK). Related experiments validate the effectiveness of them in terms of accuracy, convergence, Rademacher complexity, etc.

Citation: Zhu C, Ji X, Chen C, Zhou R, Wei L, Zhang X (2018) Improved linear classifier model with Nyström. PLoS ONE 13(11): e0206798. https://doi.org/10.1371/journal.pone.0206798

Editor: Antonio Agudo, Institut de Robotica i Informatica Industrial, SPAIN

Received: November 14, 2017

Accepted: October 14, 2018

Published: November 5, 2018

Data Availability Statement: Data underlying the study belong to third parties and are available as follows: data are available from UCI machine learning repository (http://archive.ics.uci.edu/ml). Data NUS-WIDE is from http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm. Data YMVG is from http://archive.ics.uci.edu/ml/datasets/youtube+multiview+video+games+dataset. Data DBLP is from http://dblp.uni-trier.de/xml/. Data Cora is from http://www.cs.umd.edu/~sen/lbc-proj/data/cora.tgz. The authors did not have any special access that others would not have.

Introduction

Background

In real-world applications, most data sets consist of interlaced-distributed samples from multiple classes. If samples cannot (can) be classified correctly with a linear hyperplane, we name them nonlinearly (linearly) separable samples. As we know, linear classifiers including HK, MHKS, and SVM [1] are feasible to process linearly separable samples. While for nonlinearly ones which are ubiquitous, nonlinear classifiers including NCC [2], FC-NTD [3], KMHKS [4], KSVM [5] are more suitable. One kind of nonlinear classifiers is kernel-based ones including MultiV-KMHKS [6], MVMHKS [7], RMVMHKS [8], DLMMLM [9], UDLMMLM [10], etc [11–13] and they adopt kernel functions to generate kernel matrices firstly and get optimal classifier parameters after the solution of these matrices. Here, for convenience, we summary full names and abbreviations for some terms in Table 1.

Problem and previous solutions

Most kernel-based classifiers cost an $O(n^3)$ computational complexity to decompose matrices and an $O(Mn^2)$ space complexity to store them where n is number of samples and M is number of used kernel functions. But the complexities are too high for most real-world classification problems. Fortunately, some classifiers including NMKMHKS [14], INMKMHKS [11], and NysCK which is developed on the base of cluster kernel (CK) [15] are developed to reduce

Funding: This work is supported by (1) Natural Science Foundation of Shanghai under grant numbers 16ZR1414500, 16ZR1414400 (2) National Natural Science Foundation of China under grant numbers 61602296, 41701523 (3) Shanghai Pujiang talent plan under grant number 16PJ1403700 and the authors would like to thank their supports. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

Table 1. Full name and abbreviation for some used terms.

Full name	Abbreviation
Ho-Kashyap algorithm	HK
Ho-Kashyap algorithm with squared approximation of the misclassification errors	MHKS
support vector machine	SVM
nonlinearly combined classifiers	NCC
fuzzy clustering with nonlinearly transformed data	FC-NTD
kernelized modification of MHKS	KMHKS
kernel SVM	KSVM
multi-views KMHKS	MultiV-KMHKS
multi-view learning developed from single-view patterns with Ho-Kashyap linear classification strategy	MVMHKS
regularized MVMHKS	RMVMHKS
double-fold localized multiple matrix learning machine	DLMMLM
Universum based DLMMLM	UDLMMLM
Nyström approximation matrix with multiple KMHKSs	NMKMHKS
improved NMKMHKS	INMKMHKS
cluster kernel	CK
Nyström CK	NysCK
improved NysCK	INysCK
multi-view INysCK	MINysCK
multi-view L2-SVM	MSVM
multiple graph regularized generative model	MGGM
multi-view least squares support vector machines	MV-LSSVM
multi-view and multi-feature learning	MVMFL
semi-supervised multi-view maximum entropy discrimination approach	SMVMED
multi-view low-rank sparse subspace clustering	MLRSSC
kernel MLRSSC	KMLRSSC
multi-view kernel spectral clustering	MVKSC
matrix-pattern-oriented MHKS with boundary projection discrimination	BPDMatMHKS
regularized weighted least square support vector classifier	rWLSSVC
novel dissimilarity learning	NDL
locality constrained dictionary learning	LCDL
scale-invariant feature transform	SIFT
singular value decomposition	SVD
radial basis function	RBF

https://doi.org/10.1371/journal.pone.0206798.t001

complexities. (1) NMKMHKS selects s samples from n ones and uses Nyström approximation technique to get approximation form for each kernel matrix. With NMKMHKS, computational complexity can be reduced to $O(Mns^2)$ and space complexity can be reduced to $O(n^2)$. While since the numbers and parameters of used kernel functions should be initialized beforehand and s is set in random, the performance of NMKMHKS maybe poor when comes to noise cases and is sensitive to s. (2) INMKMHKS adopts clustering technology to guide the generations of kernel functions and approximation matrices. This operation can solve the defects of NMKMHKS and keep a lower complexity. (3) NysCK decomposes each kernel matrix $\boldsymbol{K}$ by $\boldsymbol{K} = \boldsymbol{F}\boldsymbol{F}^T$ where each row in $\boldsymbol{F}$ represents a linearly separable sample and then nonlinearly separable samples can be changed to linearly ones. [15] has validated that those linearly ones correspond to the original ones and they can be classified by linear classifiers with a high accuracy.

Motivation and novelty

Since INMKMHKS avoids the setting of s and kernel parameters and NysCK changes the non-linearly separable samples to linearly ones, we combine them in together to develop improved NysCK (INysCK) to reduce complexities further. Moreover, multi-view data set which consists of samples with multiple views and each view consists of multiple features is a widely used one in real world and many corresponding multi-view classifiers are developed [16–18]. Since INysCK has not an ability to process multi-view data sets, thus we extend INysCK into multi-view applications and propose multi-view INysCK (MINysCK).

Since INMKMHKS (NysCK) was developed in 2015 (2017), thus ideas and innovations of them are still new at some extends. What's more, to the best of our knowledge, until now, there is no method combines their ideas in together. In other words, the idea of our methods is novel and it is the first trial for this. In our methods, for the original data set, we first adopt the ideas of INMKMHKS to generate several kernel functions and get the corresponding Nyström approximation matrices. Then on the base of these matrices, we adopt the ideas of NysCK to get $\boldsymbol{F}$. In $\boldsymbol{F}$, each row represents a linearly separable sample which corresponds to an original sample. Then we can classify these linearly separable samples with linear classifiers. This operation is similar with the one of classifying the original samples with some nonlinear classifiers. Moreover, this operation won't influence the classification results.

Contribution

Contributions of our work are (1) provide a new idea to process nonlinear classification problems and needn't to initialize many parameters beforehand; (2) keep low computational and space complexities; (3) first time to process multi-view problems with such an idea.

Related work

Nyström approximation technique

For a kernel-based classifier, whether the solution is feasible or not depends on the eigendecomposition of kernel matrix and in general, the eigendecomposition needs a $O(n^3)$ computational cost where n is number of samples. In order to cut down the computational cost, [19] develops Nyström approximation technique to speed up the eigendecomposition. Simply speaking, one selects s samples from the whole data set to approximate the kernel matrix, then computational complexity can be reduced to $O(ns^2)$. Recently, Nyström approximation technique has been applied into multiple different fields. For example, [20] uses this technique to get approximate infinite-dimensional region covariance descriptor which significantly outperforms the low-dimensional descriptors on image classification task; [21] introduces Nyström into kernel subspace learning and reduces the time and space complexities; [22] combines Nyström method with spectral clustering algorithm to decrease the computation complexity of the spectral clustering with a high clustering accuracy kept.

NysCK

Suppose there is a nonlinearly separable data set $\boldsymbol{X} = [\boldsymbol{X_l}, \boldsymbol{X_v}]$, $\boldsymbol{X_l}$ consists of l labeled samples, $\boldsymbol{X_v}$ consists of v unlabeled samples, $l < v$, $n = l + v$. Objective of NysCK is changing nonlinearly separable samples to linearly ones with Nyström approximation technique and predicting class labels of those unlabeled ones. For these n samples, NysCK constructs a kernel matrix $\boldsymbol{K}$ firstly and then selects s samples from $\boldsymbol{X_l}$ to decompose $\boldsymbol{K}$ with $\boldsymbol{K} = \begin{bmatrix} \boldsymbol{W} & \boldsymbol{K}_{12} \\ \boldsymbol{K}_{21} & \boldsymbol{K}_{22} \end{bmatrix}$ and

$\boldsymbol{C} = \begin{pmatrix} \boldsymbol{W} \\ \boldsymbol{K}_{21} \end{pmatrix}$ where $\boldsymbol{W} \in \mathbb{R}^{s \times s}$ corresponds to the s samples. After that, NysCK carries out SVD on $\boldsymbol{W}$ and gets the Nyström approximation matrix of $\boldsymbol{K}$, i.e., $\widetilde{\boldsymbol{K}} = \boldsymbol{C}\boldsymbol{W}_k^{\dagger}\boldsymbol{C}^T \approx \boldsymbol{K}$ where $\boldsymbol{W}_k^{\dagger}$ denotes the pseudo-inverse of $\boldsymbol{W}_{\boldsymbol{k}}$ which is the best rank-k approximation of $\boldsymbol{W}$. Finally, NysCK decomposes $\widetilde{\boldsymbol{K}}$ with $\widetilde{\boldsymbol{K}} = \boldsymbol{F}\boldsymbol{F}^T$ where $\boldsymbol{F} = \begin{bmatrix} \boldsymbol{F}_l \\ \boldsymbol{F}_v \end{bmatrix}$ represents n linearly separable samples. Each row of $\boldsymbol{F}$ corresponds to an original sample, i.e., $\boldsymbol{F}_{\boldsymbol{l}}$ corresponds to $\boldsymbol{X}_{\boldsymbol{l}}$ while $\boldsymbol{F}_{\boldsymbol{v}}$ corresponds to $\boldsymbol{X}_{\boldsymbol{v}}$. Finally, we can train a linear classifier on $\boldsymbol{F}_{\boldsymbol{l}}$ and classify $\boldsymbol{F}_{\boldsymbol{v}}$. According to [15], computational and space complexities are $O(n(nd + k^2))$ and $O(n(d + k))$ respectively where d is the dimension of each sample.

INMKMHKS

Procedure of INMKMHKS consists of four steps. (1) For a data set $\boldsymbol{X}$ with n samples, INMKMHKS adopts kernel clustering to cover $\boldsymbol{X}$ with M clusters and samples in each cluster have same class labels. Then INMKMHKS regards midpoint and width of each cluster as parameters of a RBF kernel and as a result, M kernel functions are generated without setting initial kernel parameters. (2) With usage of M kernel functions, INMKMHKS generates M kernel matrices $\boldsymbol{K}_{\boldsymbol{p}}$s and gets corresponding Nyström approximation forms $\widetilde{\boldsymbol{K}_p}$s without setting initial s ($p = 1, 2,\ldots, M$). (3) INMKMHKS calculates coefficient of each $\widetilde{\boldsymbol{K}_p}$ and constructs ensemble kernel matrix $\boldsymbol{G}$ with $\widetilde{\boldsymbol{K}_p}$s and corresponding coefficients. (4) INMKMHKS applies $\boldsymbol{G}$ into the KMHKS-based process and gets the final discriminant function.

INysCK and MINysCK

INysCK

Generating kernel functions without setting initial kernel parameters. Suppose there is a L-class data set $\boldsymbol{X}$ including l training samples $\boldsymbol{X}_{\boldsymbol{l}}$ and v test samples $\boldsymbol{X}_{\boldsymbol{v}}$ ($n = l + v$). Here $\boldsymbol{X}_{\boldsymbol{l}} = \{\boldsymbol{X}_1, \boldsymbol{X}_2,\ldots, \boldsymbol{X}_L\} = \{\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_l\}$ and the class labels are $\boldsymbol{Y} = \{y_1, y_2,\ldots, y_L\}$. For each class $\boldsymbol{X}_{\boldsymbol{c}}$ ($c = 1, 2,\ldots, L$), it consists of n_c samples, i.e., $\boldsymbol{X}_c = \{\boldsymbol{x}_{c1}, \boldsymbol{x}_{c2}, \ldots, \boldsymbol{x}_{cn_c}\}$. Here, n_c is the number of samples in $\boldsymbol{X}_{\boldsymbol{c}}$ and $l = n_1 + n_2 + \ldots + n_L$. $\boldsymbol{x}_{\boldsymbol{cj}}$ is jth sample of $\boldsymbol{X}_{\boldsymbol{c}}$ where $j \in \{1, 2, \ldots, n_c\}$. Then on the base of l training ones, we generate kernel functions with the following way.

For generating the first kernel function, we compute midpoint $\boldsymbol{\mu}$ of all training samples, i.e., $\boldsymbol{\mu} = \frac{\sum_{i=1}^{l} \boldsymbol{x}_i}{l}$ and distance between $\boldsymbol{x}_{\boldsymbol{i}}$ and $\boldsymbol{\mu}$, i.e., $d_{\boldsymbol{x}_i}$. Distances are sorted in an ascending order, i.e., $d_{\boldsymbol{x}_{(1)}} \leq d_{\boldsymbol{x}_{(2)}} \leq \ldots \leq d_{\boldsymbol{x}_{(l)}}$ and the corresponding samples are denoted as $\boldsymbol{x}_{(1)}$, $\boldsymbol{x}_{(2)},\ldots,\boldsymbol{x}_{(l)}$. If the class labels of $\boldsymbol{x}_{(1)}$, $\boldsymbol{x}_{(2)},\ldots,\boldsymbol{x}_{(u)}$ are same while the label of $\boldsymbol{x}_{(u+1)}$ is not same as the one of $\boldsymbol{x}_{(u)}$, then we let kernel parameters $(\sigma, \boldsymbol{\mu}') = (\boldsymbol{d}_{\boldsymbol{x}_{(u)}}, \boldsymbol{\mu})$ (in our work, used kernel function is RBF and its expression is $k(\boldsymbol{x}_i, \boldsymbol{\mu}') = exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}'\|^2}{2\sigma^2}\right)$). Then for this kernel function, we let corresponding samples $\boldsymbol{x}_{(1)}$, $\boldsymbol{x}_{(2)},\ldots,\boldsymbol{x}_{(u)}$ be basic samples and u be basic number. What's more, in order to generate the second kernel function, we remove $\boldsymbol{x}_{(1)}$, $\boldsymbol{x}_{(2)},\ldots,\boldsymbol{x}_{(u)}$ from $\boldsymbol{X}_{\boldsymbol{l}}$ and repeat previous steps. We repeat the steps again and again until each training sample belongs to basic samples of one kernel function. After this generation way, we can get M new kernel functions, i.e., $k_1(\boldsymbol{x}_i, \boldsymbol{x}_j),\ldots, k_p(\boldsymbol{x}_i, \boldsymbol{x}_j),\ldots, k_M(\boldsymbol{x}_i, \boldsymbol{x}_j)$ where $p = 1, 2,\ldots, M$, we also get the corresponding M σs and $\boldsymbol{\mu}'$s.

Constructing kernel matrices with Nyström approximation technique. (1) We construct kernel matrices according to these M kernel functions. Suppose for pth kernel function, its parameters are $(\sigma_p, \boldsymbol{\mu}'_p)$ and basic samples are $\boldsymbol{x}_{(1)},\ldots, \boldsymbol{x}_{(u)}$. With all n samples, the corresponding $n \times n$ kernel matrix is $\boldsymbol{K_p} = (k(\boldsymbol{x_i}, \boldsymbol{x_j}))_{n\times n}$ and its ith row and jth column element is $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\sigma_p^2}\right)$ where $i, j = 1,\ldots, n$. For convenience, in $\boldsymbol{K_p}$, $\{\boldsymbol{x_1},\ldots,\boldsymbol{x_l}\}$ corresponds to l training samples and $\{\boldsymbol{x_{l+1}},\ldots,\boldsymbol{x_n}\}$ corresponds to v test ones.

(2) We centralize and normalize $\boldsymbol{K_p}$ with Eqs (1) and (2) just for convenient calculation. Here $\mathbf{1}_{\boldsymbol{n\times n}}$ is a $n \times n$-dimensional identity matrix, *trace* indicates the trace of a matrix, and we use $\boldsymbol{K_p}$ to denote the centralized and normalized matrix $\boldsymbol{K_{trp}}$ for convenience.

$$\boldsymbol{K}_{cp} = \boldsymbol{K_p} - \frac{1}{n}\mathbf{1}_{\boldsymbol{n\times n}}\boldsymbol{K_p} - \frac{1}{n}\boldsymbol{K_p}\mathbf{1}_{\boldsymbol{n\times n}} + \frac{1}{n^2}\mathbf{1}_{\boldsymbol{n\times n}}\boldsymbol{K_p}\mathbf{1}_{\boldsymbol{n\times n}} \tag{1}$$

$$\boldsymbol{K}_{trp} = \frac{\boldsymbol{K}_{cp}}{trace(\boldsymbol{K}_{cp})} \tag{2}$$

(3) For $\boldsymbol{K_p}$, if both $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are in the set $\{\boldsymbol{x}_{(1)},\ldots,\boldsymbol{x}_{(u)}\}$, we combine these $k(\boldsymbol{x_i}, \boldsymbol{x_j})$s together and generate an $u \times u$-dimensional matrix, i.e, $\boldsymbol{W_p}$. If only one of $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ is in this set, we combine these $k(\boldsymbol{x_i}, \boldsymbol{x_j})$s together and generate a $(n - u) \times u$-dimensional matrix, i.e, $\boldsymbol{K_{p21}}$. Then $\boldsymbol{K_{p12}} = \boldsymbol{K}^T_{\boldsymbol{p21}}$. If neither $\boldsymbol{x_i}$ nor $\boldsymbol{x_j}$ is in this set, we combine these $k(\boldsymbol{x_i}, \boldsymbol{x_j})$s together and generate a $(n - u) \times (n - u)$-dimensional matrix, i.e, $\boldsymbol{K_{p22}}$. The relative positions of those $k(\boldsymbol{x_i}, \boldsymbol{x_j})$s in $\boldsymbol{W_p}$, $\boldsymbol{K_{p12}}$, $\boldsymbol{K_{p21}}$, and $\boldsymbol{K_{p22}}$ are not changed. With above definitions, we decompose $\boldsymbol{K_p}$ with Eq (3) and let $s = u$ without initializing s.

$$\boldsymbol{K}_p = \begin{bmatrix} \boldsymbol{W}_p & \boldsymbol{K}_{p12} \\ \boldsymbol{K}_{p21} & \boldsymbol{K}_{p22} \end{bmatrix} \quad and \quad \boldsymbol{C}_p = \begin{pmatrix} \boldsymbol{W}_p \\ \boldsymbol{K}_{p21} \end{pmatrix} \tag{3}$$

(4) We carry out SVD on $\boldsymbol{W_p}$, i.e. $\boldsymbol{W}_p = \boldsymbol{U}_p\Lambda_p\boldsymbol{U}_p^T$. Here, $\boldsymbol{\Lambda_p} = diag(\sigma_{p1}, \cdots, \sigma_{pu})$. σ_{pi} is the ith largest singular value. $\boldsymbol{U_p}$ is composed of the eigenvectors of the $\boldsymbol{W_p}$ based on σ_{pi} and *diag* indicates the diagonalization operation.

(5) We get the rank-k Nyström approximation matrix for $\boldsymbol{K_p}$ by

$$\widetilde{\boldsymbol{K}_p} = \boldsymbol{C}_p\boldsymbol{W}_{pk}{}^{+}\boldsymbol{C}_p^T \tag{4}$$

where $\boldsymbol{W}_{pk}^{+} = \sum_{i=1}^{k}\sigma_{pi}^{-1}\boldsymbol{U}_p^{(i)}\boldsymbol{U}_p^{(i)T}$ is the rank-k pseudo-inverse of $\boldsymbol{W_p}$, k ($k \leq u$) is the rank of $\boldsymbol{W}_{pk}^{+}$ and $\boldsymbol{U}_p^{(i)}$ is ith column of $\boldsymbol{U_p}$.

(6) After repeating previous steps, we can get all $\widetilde{\boldsymbol{K}_p}$s and for each $\boldsymbol{K_p}$, we have a corresponding basic number u. Then we let largest u be s.

(7) According to [14] and Nyström approximation error between $\boldsymbol{K_p}$ and $\widetilde{\boldsymbol{K}_p}$, we calculate coefficient α_p of each $\widetilde{\boldsymbol{K}_p}$ by Eq (5).

$$\xi_p = \|\widetilde{\boldsymbol{K}_p} - \boldsymbol{K}_p\|_F \quad and \quad \alpha_p = \frac{e^{-\eta\xi_p}}{Z} \tag{5}$$

where $\|.\|_F$ represents the Frobenius norm, $\eta > 0$ is a predefined parameter, $Z = \sum_{p=1}^{M} e^{-\eta\xi_p}$ is a normalization factor which is used to set $\sum_{p=1}^{M} \alpha_p = 1$.

(8) Finally, we get the ensemble kernel matrix $\boldsymbol{G}$ with the following equation.

$$\boldsymbol{G} = \sum_{p=1}^{M} \alpha_p \widetilde{\boldsymbol{K}_p} \tag{6}$$

Getting corresponding linearly separable samples. (1) Once we get $\boldsymbol{G}$, we let $\boldsymbol{D} = diag(D_{11},\ldots, D_{nn})$ where element $D_{ii} = \sum_{j=1}^{n} G_{ij}$ and G_{ij} represents the ith row and jth column element of $\boldsymbol{G}$ ($i, j = 1,\ldots, n$). Then we decompose $\boldsymbol{G}$ with Eq (7) where $\boldsymbol{W}_G \in \mathbb{R}^{s\times s}$, $\boldsymbol{G}_{12} \in \mathbb{R}^{s\times(n-s)}$, $\boldsymbol{G}_{12} = \boldsymbol{G}_{21}^T$, $\boldsymbol{G}_{22} \in \mathbb{R}^{(n-s)\times(n-s)}$, $\boldsymbol{C}_G \in \mathbb{R}^{n\times s}$. Since s is gotten with sub-step (6) in previous subsection and $\boldsymbol{G}$ is combination of multiple $\widetilde{K_p}$, so elements in $\boldsymbol{W_G}$, $\boldsymbol{G_{12}}$, $\boldsymbol{G_{21}}$, $\boldsymbol{G_{22}}$ are fixed here. Since $s < l$, $n - s = l + v - s > v$, so $v \times v$ part in the lower right corner of $\boldsymbol{G_{22}}$ corresponds to v test samples.

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{W}_G & \boldsymbol{G}_{12} \\ \boldsymbol{G}_{21} & \boldsymbol{G}_{22} \end{bmatrix} \quad and \quad \boldsymbol{C}_G = \begin{pmatrix} \boldsymbol{W}_G \\ \boldsymbol{G}_{21} \end{pmatrix} \tag{7}$$

(2) We carry out SVD on $\boldsymbol{W_G}$, i.e. $\boldsymbol{W}_{Gk} = \boldsymbol{U}_{W_G,k}\boldsymbol{\Sigma}_{W_G,k}\boldsymbol{U}_{W_G,k}^T$ where $\boldsymbol{W_{Gk}}$ is the best rank-k approximation of $\boldsymbol{W_G}$, $\Sigma_{\boldsymbol{W_G,k}}$ is a diagonal matrix and diagonal consists of first k approximate eigenvalues, and $\boldsymbol{U_{W_G,k}}$ consists of the corresponding k approximate eigenvectors. Then we get Nyström approximation matrix $\widetilde{\boldsymbol{G}}$ of $\boldsymbol{G}$ with Eq (8) where $\boldsymbol{W}_{Gk}^{\dagger}$ denotes the pseudo-inverse of $\boldsymbol{W_{Gk}}$.

$$\boldsymbol{C}_G \boldsymbol{W}_{Gk}^{\dagger} \boldsymbol{C}_G^T \approx \widetilde{\boldsymbol{G}} \tag{8}$$

(3) After that, we compute $\Sigma_{\boldsymbol{k}}$ and $\boldsymbol{U_k}$ using Eq (9) where $\boldsymbol{\Sigma}_{W_G,k}^{\dagger}$ is the pseudo-inverse of $\Sigma_{\boldsymbol{W_G,k}}$. In terms of each element σ_i of $\Sigma_{\boldsymbol{k}}$, we apply Eq (10) and get the $\widetilde{\boldsymbol{\Sigma}_k} = diag(\varphi(\sigma_1), \ldots, \varphi(\sigma_k))$. Generally speaking, since v is larger than 9, so we use $h = l + 9$ is feasible.

$$\boldsymbol{\Sigma}_k = \left(\frac{n}{s}\right)\boldsymbol{\Sigma}_{W_G,k} \quad and \quad \boldsymbol{U}_k = \sqrt{\frac{s}{n}}\boldsymbol{C}_G\boldsymbol{U}_{W_G,k}\sum\nolimits_{W_G,k}^{\dagger} \tag{9}$$

$$\varphi(\sigma_i) = \begin{cases} \sqrt{\sigma_i} & i < h(= l+9) \\ \sigma_i^2 & i \geq h(= l+9) \end{cases} \tag{10}$$

(4) Once we get $\widetilde{\boldsymbol{\Sigma}_k}$, we let $\widetilde{\boldsymbol{L}} \approx \boldsymbol{D}^{-1/2}\boldsymbol{U}_k\widetilde{\boldsymbol{\Sigma}_k}\boldsymbol{U}_k^T\boldsymbol{D}^{-1/2}$ and then $\widetilde{\boldsymbol{L}^{1/2}} \approx \boldsymbol{D}^{-1/2}\boldsymbol{U}_k(\widetilde{\boldsymbol{\Sigma}_k})^{1/2}$. Then, we have $\widetilde{D_{ii}} = 1/\widetilde{L_{ii}}$ and get $\widetilde{\boldsymbol{D}} = diag(\widetilde{D_{11}}, \ldots, \widetilde{D_{nn}})$ where $\widetilde{L_{ii}}$ is ith row and ith column element of $\widetilde{\boldsymbol{L}}$. Finally, we can get $\widetilde{\boldsymbol{D}}^{1/2}$ and linearly separable data set $\boldsymbol{F} = \begin{bmatrix} \boldsymbol{F}_l \\ \boldsymbol{F}_v \end{bmatrix}$ by Eq (11). According to [15] said, each row of $\boldsymbol{F}$ corresponds to an original sample, i.e., $\boldsymbol{F_l}$

Table 2. Algorithm: INysCK.

Input: L-class data set $\boldsymbol{X} = [\boldsymbol{X_l}, \boldsymbol{X_v}]$. $\boldsymbol{X_l}$ consists of l training samples and $\boldsymbol{X_v}$ consists of v test samples. Here, $n = l + v$
1. Generate M kernel functions.
2. For p = 1,2,. . .,M do
3. Construct kernel matrix $\boldsymbol{K_p}$ with pth kernel function.
4. Centralize, normalize, and decompose $\boldsymbol{K_p}$ with Eqs (1) and (3).
5. Carry out SVD on $\boldsymbol{W_p}$ and get the rank-k Nyström approximation matrix $\widetilde{\boldsymbol{K_p}}$ for K_p with Eq (4).
6. End for
7. Compute coefficient α_p of each $\widetilde{\boldsymbol{K_p}}$ with Eq (5) and get ensemble kernel matrix $\boldsymbol{G}$ with Eq (6).
8. On the base of $\boldsymbol{G}$, get $\boldsymbol{D}$ and decompose $\boldsymbol{G}$ with Eq (7).
9. Carry out SVD on $\boldsymbol{W_G}$ and get Nyström approximation matrix $\widetilde{\boldsymbol{G}}$ of $\boldsymbol{G}$ with Eq (8).
10. Compute Σ_k, $\boldsymbol{U_k}$ and get $\widetilde{\boldsymbol{\Sigma_k}}$ with Eqs (9) and (10).
11. Get $\widetilde{\boldsymbol{L}}^{1/2}$ and $\widetilde{\boldsymbol{D}}^{\boldsymbol{1/2}}$ and obtain $\boldsymbol{F}$ with Eq (11).
Output: $\boldsymbol{F} = \begin{bmatrix} \boldsymbol{F_l} \\ \boldsymbol{F_v} \end{bmatrix}$ and $\boldsymbol{F_l}$ corresponds to $\boldsymbol{X_l}$ while $\boldsymbol{F_v}$ corresponds to $\boldsymbol{X_v}$

https://doi.org/10.1371/journal.pone.0206798.t002

corresponds to $\boldsymbol{X_l}$ and $\boldsymbol{F_v}$ corresponds to $\boldsymbol{X_v}$. Once $\boldsymbol{F}$ is gotten, we can adopt linear classifiers to train and classify them. For convenience, Table 2 shows framework of INysCK.

$$\boldsymbol{F} \approx \widetilde{\boldsymbol{D}}^{1/2}\widetilde{\boldsymbol{L}}^{1/2} \tag{11}$$

MINysCK

Suppose there is a multi-view data set $\boldsymbol{X} = \{\boldsymbol{X}^g\}_{g=1}^{V} = \{\boldsymbol{x}_i\}_{i=1}^{n}$ where V is the number of views and n is the number of samples. The gth view is $\boldsymbol{X}^g = \{\boldsymbol{x}_i^g\}_{i=1}^{n}$ and the ith sample is $\boldsymbol{x}_i = \{\boldsymbol{x}_i^g\}_{g=1}^{V}$. $\boldsymbol{x}_i^g$ represents gth view of ith sample. For each view $\boldsymbol{X}^g$, its dimension is d^g which indicates that this view consists of d^g features. Now in the procedure of MINysCK, we conduct INysCK on each view $\boldsymbol{X}^g$ and get the corresponding $\boldsymbol{F}$, i.e., $\boldsymbol{F}^g$. Then for V views, we get V groups $\boldsymbol{F}$. For the $\boldsymbol{X}$, its linear form is $\boldsymbol{F} = \{\boldsymbol{F}^1,\boldsymbol{F}^2,. . .,\boldsymbol{F}^V\}$. Finally, we can adopt some multi-view classifiers to process $\boldsymbol{F}$. Table 3 shows framework of MINysCK.

Computational complexity and space complexity

According to [15], the computational complexity and space complexity of NysCk are $O(n(nd + k^2))$ and $O(n(d + k))$ respectively where d is the dimension of each sample. Then compared with NysCk, the main added steps of INysCK are the generation of kernel functions and matrices. Thus, the added computational complexity is $O(Ml^2)$ and the added space complexity is O

Table 3. Algorithm: MINysCK.

Input: multi-view data set $\boldsymbol{X} = \{\boldsymbol{X}^g\}_{g=1}^{V} = \{\boldsymbol{x}_i\}_{i=1}^{n}$
1. For g = 1,2,. . .,V do
2. Change $\boldsymbol{X}^g$ to corresponding $\boldsymbol{F}^g$ with INysCK.
3. End for
4. Obtain $\boldsymbol{F} = \{\boldsymbol{F}^1,\boldsymbol{F}^2,. . .,\boldsymbol{F}^V\}$.
Output: $\boldsymbol{F}$

https://doi.org/10.1371/journal.pone.0206798.t003

(Ml^2). Since in real-world applications, $M \ll n$ and $l \ll n$, so computational complexity and space complexity of INysCk are almost same as ones of NysCK. For MINysCK, the computational complexity is $\sum_{g=1}^{V} O(n(nd^g + k^2)) = O(n(nd + k^2))$, the space complexity is $\sum_{g=1}^{V} O(n(d^g + k)) = O(n(d + k))$. Thus, we find that computational and space complexities of INysCK and MINysCK are same as the ones of NysCK.

Experiments

Experimental setting

We adopt four multi-view data sets (NUS-WIDE, YMVG, DBLP, Cora) and four UCI machine learning repository (UCI) [23] data sets (YCS, AA, BC, Arrhythmia) for experiments in niche targeting. Among these data sets, half of them are large-scale and the left are small-scale. Information of used UCI data sets is given in Table 4 and in terms of four multi-view ones, we describe them as below where *D* denotes dimensionality. (1) NUS-WIDE is a web image data set which consists of 269648 samples (images), 5018 classes, and 6 views [24]. The six views are color histogram (Col-h, 64-D), color correlogram (Col-c, 144-D), edge direction histogram (Ed-h, 73-D), wavelet texture (Wav, 128-D), block-wise color moments (Bw-cm, 225-D), and bag of words based on SIFT descriptions (B-SIFT, 500-D); (2) YMVG [25] is the abbreviation of YouTube multi-view video games and it consists of 120000 samples (videos) from 31 classes (games). Each sample consists of 13 views. They are audio mfcc (A-m, 2000-D), audio sai boxes (A-s-b, 7168-D), audio sai intervalgrams (A-s-i, 4096-D), audio spectrogram stream (A-s-s, 1024-D), audio volume stream (A-v-s, 64-D), text description unigrams (T-d-u, 558936-D), text game lda 1000 (T-g-l, 1000-D), text tag unigrams (T-t-u, 422627-D), vision cuboids histogram (V-c-h, 512-D), vision hist motion estimate (V-h-m-e, 64-D), vision hog features (V-h-f, 647-D), vision hs hist stream (V-h-h-s, 1024-D), and vision misc (V-m, 838-D); (3) DBLP [26, 27] is abbreviation of digital bibliography and library project. Original DBLP is very large and we select 5000 samples from 4 classes for experiments. Each sample has two views, one is paper name (P-n, 6167-D) and the other is term (Te, 3787-D); (4) Cora [27, 28] is adapted from original Cora data set [29] and it consists of 12004 scientific articles (samples) from 10 thematic classes and for each sample, it has two views, i.e., content (Co, 292-D) and relational (Re, 12004-D). What's more, these data sets are third party ones and others would be able to access these data in the same manner as what we have done. Moreover, we confirm that we don't have any special access privileges that others would not have.

Then we adopt CK, NysCK, INysCK, and MINysCK to change nonlinearly separable samples to linearly ones. If we use original data sets for experiments, we adopt 'Null' for

Table 4. Description of the used UCI data sets.

Data set	No. dimensions	No. classes	No. samples
YouTube Comedy Slam (YCS)	2	2	1138562
Authorship Attribution (AA)	1000	50	93600
Breast Cancer (BC)	10	2	699
Arrhythmia	279	16	452

https://doi.org/10.1371/journal.pone.0206798.t004

Table 5. Used classifiers.

	nonlinear	linear
single-view	KMHKS [4], KSVM [5]	SVM [1], MHKS [1]
	NDL [34], LCDL [35]	BPDMatMHKS [36], rWLSSVC [37]
multi-view	MultiV-KMHKS [6], DLMMLM [9]	MSVM [38], MLRSSC [39]
	MGGM [27], MV-LSSVM [40]	
	MVMFL [41], SMVMED [42]	
	KMLRSSC [39], MVKSC [43]	

https://doi.org/10.1371/journal.pone.0206798.t005

representation. Here, we treat CK as a baseline method and if we only compare with NysCK, NysCK can be regarded as a baseline one.

We use classifiers shown in Table 5 for further processing and linear classifiers are only feasible for linearly separable data sets while nonlinear classifiers are feasible for both nonlinearly and linearly ones. Similarly, multi-view classifiers can process not only multi-view but also single-view data sets while single-view classifiers are only feasible for single-view ones. We adopt SVM and MSVM as two baseline classifiers in respective experiments.

What's more, for each data set, 70% of samples are chosen in random as training samples and the remaining are for test. In order to get the truly experimental results, we adopt 10-fold cross validation strategy [10]. Moreover, one-against-one classification strategy is used for multi-class problems here [30–33]. In order to get the average experimental results, we repeat the experiments for 10 times. The computations are performed on Intel Core 4 processors with 2.66GHz, 4G RAM DDR3, Win 7, and MATLAB 2014 environment.

Independent experiments

This part shows the performances of our proposed methods on different kinds of data sets.

Accuracy comparison on large-scale single-view data sets. First, we show effectiveness of our INysCK on two large-scale single-view data sets YCS and AA. For fair comparison, we select 8 single-view classifiers shown in Table 5 for experiments and adopt CK, NysCK, INysCK to change samples into linearly separable ones. Moreover, as we know, accuracy, true positive rate (acc^{+}), true negative rate (acc^{-}), positive predictive value (PPV), F-Measure, G-Mean, etc. [44] are widely used to evaluate the classification performances. Here, we only show the results about accuracy due to for other evaluation criteria, we draw similar conclusions. Then the top two sub-figures of Fig 1 show the results. According to these two sub-figures, we can see that for large-scale single-view data sets, our INysCK brings a best accuracy no matter which classifier is used. Specially, we find compared with CK and NysCK, for the case AA with SVM, the improvement of INysCK is little while for other cases, the improvement is more. In order to elaborate this phenomenon, we analysis the distributions of YCS and AA. Since these two data sets are large-scale, so we won't show the distributions of their samples with figure and just describe the distributions in short. We find that for these two data sets, their samples distribute with an interlaced way and a high nonlinearity. After carrying out CK-related methods, most samples become linearly separable and compared with CK and NysCK, with INysCK used, samples have a higher linearity. Moreover, since the sizes of YCS and AA are large, so the advantages of linearities derived from INysCK are larger. Thus when we use those single-view classifiers no matter nonlinear ones and linear ones to process the changed samples, accuracies are higher. In terms of the case AA with SVM, we find with CK, NysCK, INysCK used, the classification functions provided by support vectors are similar. That's why that our INysCK brings a little improvement for this case.

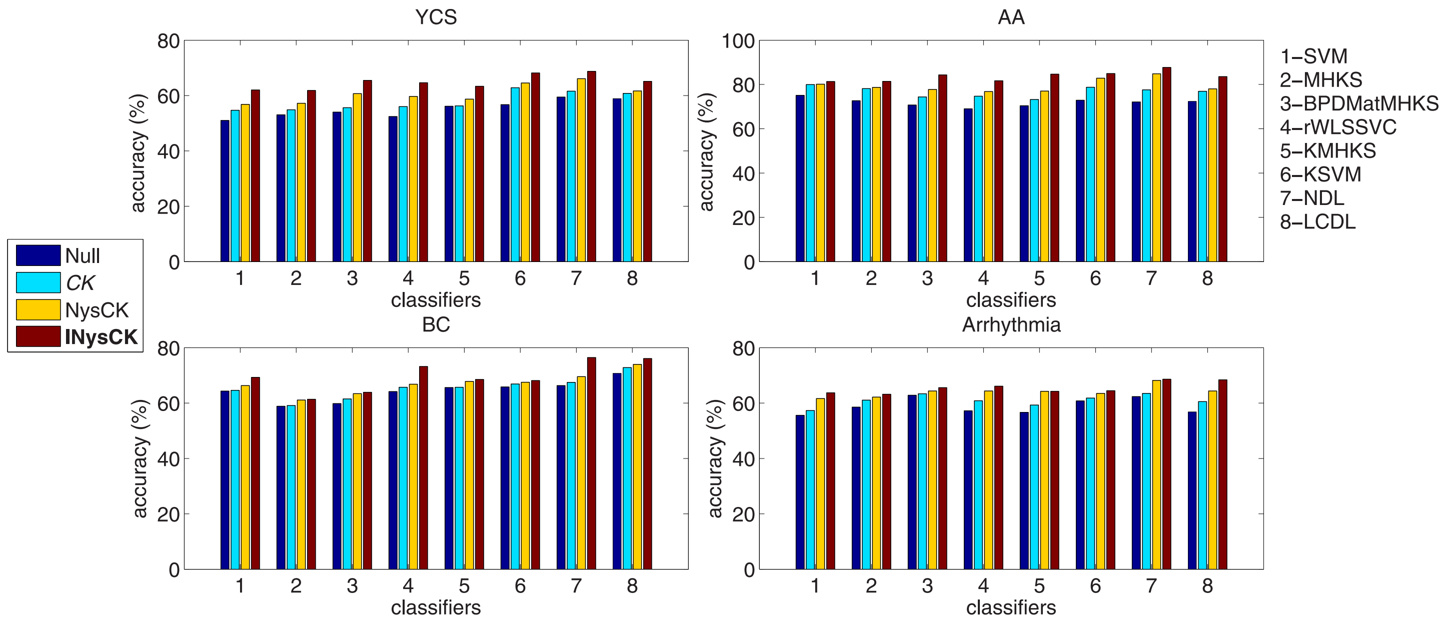

Fig 1. Accuracy with related classifiers and CK-based methods on used single-view data sets. CK-related method in italic represents baseline one and one in bold denotes the proposed one. For classifiers, SVM is used as the baseline one and we just clarify this point in words rather than in font. In other figures and tables, we have similar representations.

https://doi.org/10.1371/journal.pone.0206798.g001

Accuracy comparison on small-scale single-view data sets. Second, we adopt data sets BC and Arrhythmia for experiments so as to validate effectiveness of INysCK on small-scale single-view problems. Similarly with the above experiments, single-view classifiers shown in Table 5, and CK, NysCK are adopted. Then bottom two sub-figures of Fig 1 show the results. According to these two sub-figures, it is found that for small-scale single-view problems, our INysCK still brings a best accuracy in average. But compared with the results given in above experiments, on more cases, INysCK only brings a little improvement and we find for the case Arrhythmia with KMHKS, NysCK outperforms INysCK. For this phenomenon, we also analysis the distributions of BC and Arrhythmia. We find that with INysCK used, samples of these two data sets are linearly separable. But since the sizes of them are small, so advantages of linearities derived from INysCK are not obvious even more not exist. Thus for some cases, the improvement of INysCK is little and for the case Arrhythmia with KMHKS, INysCK performs worse than NysCK due to the advantage of linearity derived from INysCK is not exist in terms of KMHKS.

Accuracy comparison on large-scale multi-view data sets. Then, we use NUS-WIDE and YMVG to show the effectiveness of proposed methods on large-scale multi-view problems. The used classifiers are multi-view ones shown in Table 5. Then here, we use CK, NysCK, INysCK, and MINysCK to change samples. Although NUS-WIDE and YMVG are multi-view data sets, in order to carry out CK, NysCK, and INysCK, we regard all views as a whole view. The top two sub-figures of Fig 2 show the results. According to these two sub-figures, we find that as a multi-view method, our MINysCK brings a best accuracy in average and the improvement is much more than the previous results, especially, for the cases NUS-WIDE with MSVM (MultiV-KMHKS, DLMMLM, MGGM, MV-LSSVM, SMVMED, KMLRSSC) and YMVG with MSVM (MultiV-KMHKS, MGGM). The main reason is that for multi-view data sets, MINysCK is more feasible than the single ones including CK, NysCK, and INysCK. What's more, taking all views as a whole view to carry out CK, NysCK, and INysCK cannot

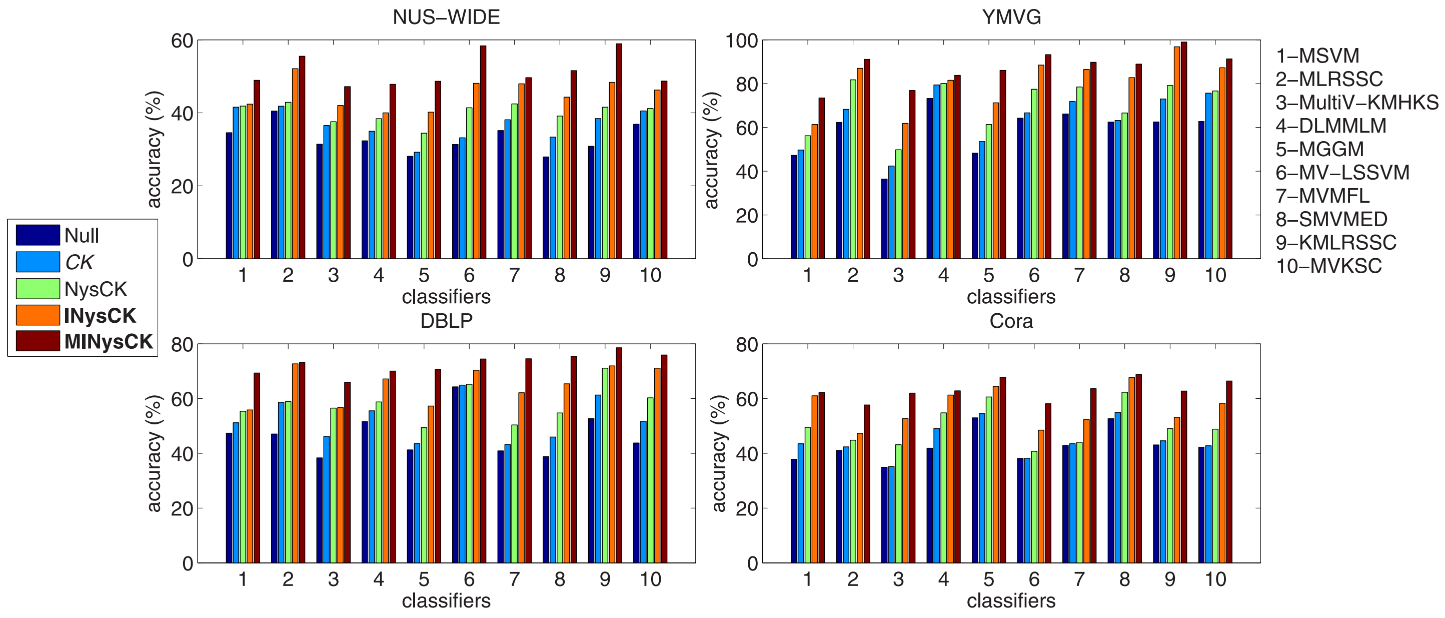

Fig 2. Accuracy with related classifiers and CK-based methods on used multi-view data sets. CK-related method in italic represents baseline one and ones in bold denote the proposed ones. For classifiers, MSVM is used as the baseline one and we just clarify this point in words rather than in font. In other figures and tables, we have similar representations.

https://doi.org/10.1371/journal.pone.0206798.g002

reflect the differences among views and this operation can only keep the linearity of samples on the whole view and cannot promise linearity on respective view. That's why MINysCK performs best on NUS-WIDE and YMVG.

Accuracy comparison on small-scale multi-view data sets. Now we adopt DBLP and Cora for the accuracy comparison on small-scale multi-view problems and other settings are same as ones given in above experiments. The bottom two sub-figures of Fig 2 show the results. From these two sub-figures, we can see that our MINysCK performs best on small-scale multi-view data sets as well. But compared with the above experimental results in careful, we find since the sizes of DBLP and Cora are more smaller than the ones of NUS-WIDE and YMVG, so advantages of linearities derived from MINysCK are not obvious, as a result, the improvement of MINysCK has a reduction.

Comparison about time cost. Besides the accuracy comparisons, we show the time cost comparison here. As we said before, the computational complexity and space complexity of INysCK and MINysCK are same as the ones of NysCK, i.e., $O(n(nd + k^2))$ and $O(n(d + k))$ respectively where d is the dimension of each sample and all of these three NysCK-related methods are used to change the nonlinearly separable samples to the linearly separable ones. Here, we show the practice time of them on different data sets in Table 6 and from this table, it is found that (1) since the procedures of INysCK and MINysCK are more complicate than NysCK, so they both cost longer time in average while the increased time is acceptable; (2) for multi-view data sets, MINysCK costs less time than INysCK, we think the main reason is that we won't regard the multiple views as a single whole view with some fusion techniques and process each view in each small problem. This maybe brings a smaller total time cost.

Comprehensive experiments

This part shows average performances of our proposed methods on all used data sets.

Table 6. Comparison about time (in seconds) cost for the three NysCK-related methods.

Data set	*NysCK*	INysCK	MINysCK	Data set	*NysCK*	INysCK	MINysCK
YCS	43.405	44.642	/	AA	146.672	154.527	/
BC	0.087	0.095	/	Arrhythmia	1.577	1.634	/
NUS-WIDE	1380.392	1434.962	1406.573	YMVG	241078.011	249127.624	247240.890
DBLP	4.166	4.282	4.266	Cora	29.663	32.586	32.069

https://doi.org/10.1371/journal.pone.0206798.t006

Distributions of samples with different CK-related methods. Here, we use a two-dimensional binary-class data set ***X*** to compare the performances of CK, NysCK, and INysCK when they change nonlinearly separable data to linearly ones. The distributions of samples before or after carrying out CK-related methods are shown in Fig 3. According to this figure, it is found that (1) all CK-related methods can change nonlinearly separable samples to linearly separable ones at some extends; (2) with INysCK used, for the same class, most samples locate in an area centrally and only few samples locate far from this area. With calculation, we find that with CK used, 20% samples locate in the area which belongs to different classes. For NysCK, the ratio is 6.5% while for INysCK, the ratio is only 3.5%. This means with our methods used, samples have a higher linearity. What's more, since it is always hard for people to show a multi-view data set or multi-dimensional data set whose dimensionality is large than two with a two-dimensional picture, thus we won't show the distribution of samples with MINysCK used and only use a two-dimensional data set for experiments here. But this won't influence our conclusion.

Convergence analysis. Convergence is an important criterion to assess the effectiveness of a classifier and if a classifier can converge within limited iterations with a better classification performance, we say this classifier is effective. What's more, the distribution of samples

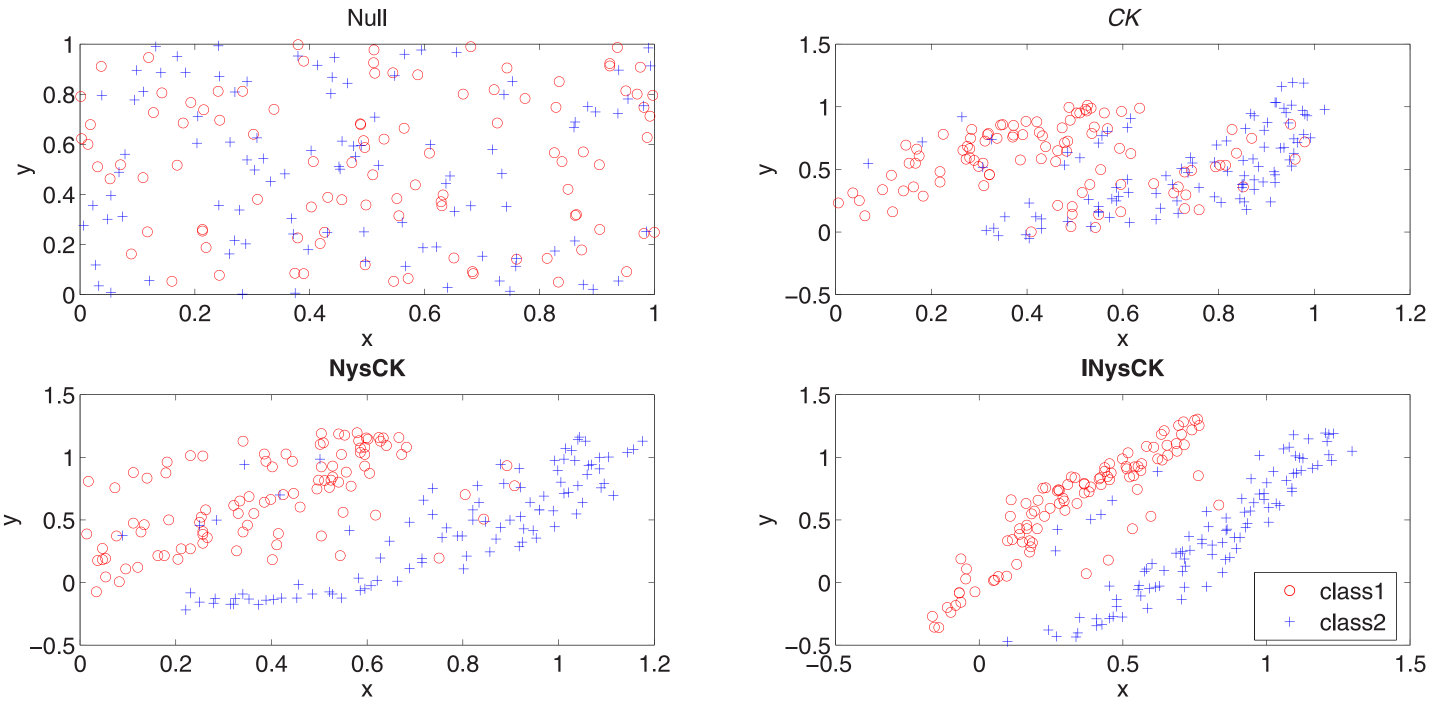

Fig 3. Distributions of samples with different CK-related methods on a binary-class data set.

https://doi.org/10.1371/journal.pone.0206798.g003

Table 7. The numbers of iterations comparisons.

single-view	Null	*CK*	NysCK	INysCK	multi-view	Null	*CK*	NysCK	INysCK	MINysCK
SVM	17.20	17.13	16.06	14.96	MSVM	17.57	16.35	14.72	14.44	13.43
MHKS	21.30	19.69	18.79	18.06	MLRSSC	29.18	26.60	24.44	23.26	22.05
BPDMatMHKS	22.94	22.94	22.29	21.89	MultiV-KMHKS	25.63	24.77	22.71	21.24	20.16
rWLSSVC	20.13	19.30	18.96	18.64	DLMMLM	28.91	27.10	25.68	24.84	23.18
KMHKS	20.82	18.98	17.55	17.13	MGGM	27.03	24.46	23.62	22.07	21.26
KSVM	17.03	16.60	15.53	15.29	MV-LSSVM	43.56	40.94	38.92	37.87	36.39
NDL	19.82	18.92	18.34	17.46	MVMFL	66.96	66.33	64.75	62.91	62.31
LCDL	22.42	21.81	19.92	19.26	SMVMED	37.19	33.95	31.37	29.06	27.69
					KMLRSSC	32.69	32.65	32.00	30.33	28.16
					MVKSC	26.78	25.55	23.48	21.46	20.70

https://doi.org/10.1371/journal.pone.0206798.t007

also affect the convergence and samples with a high linearity always accelerate the optimization of a classifier. Here, we adopt an empirical justification given in [45] to measure the convergence of classifiers with our methods used and Table 7 shows the results. Each cell in this table denotes the average number of iterations of a classifier on all used data sets with a CK-related method used. According to this table and combining the results given before, we know that with our proposed INysCK and MINysCK used, the changed samples have a higher linearity and these samples accelerate the optimization of classifiers which indicates a smaller numbers of iterations. What's more, since MINysCK is more feasible for multi-view data sets, so for the multi-view classifiers, they can converge faster with MINysCK used.

Rademacher complexity analysis. As [14] and [11] said, Rademacher complexity is a reflection about generalization risk bound and performance behavior of a classifier. A smaller Rademacher complexity indicates a better performance of a classifier and a lower generalization risk bound. Here, we adopt the same method given in [11] to compute Rademacher complexity for classifiers with different CK-related methods used. Fig 4 shows the results and according to this figure, we know (1) in terms of single-view classifiers, since samples with INysCK used have a higher linearity, so classifiers have smaller Rademacher complexities; (2) in terms of multi-view classifiers, since MINysCK is more feasible and it also makes the samples have a higher linearity, thus related Rademacher complexities are smaller.

Significance analysis. We adopt Friedman-Nemenyi statistical test [46] to validate the difference between our proposed methods and the previous work is significant. In terms of Friedman-Nemenyi statistical test, Friedman test is used to analyze if the differences between all compared algorithms on multiple data sets are significant or not while Nemenyi test is used to analyze if the differences between two compared algorithms on multiple data sets are significant or not.

In order to carry out Friedman test, we treat each CK-related method as an 'algorithm' and regard each classifier as a 'data set'. Then according to the average accuracy of an 'algorithm' on a 'data set', Friedman test ranks the 'algorithm's for each 'data set' as shown in Table 8. (1) For single-view cases, since we use 4 'algorithm's and 8 'data set's, we carry out Friedman test and get $\chi_F^2 = 21.47$ and $F_F = 59.37$ (the computation equations of χ_F^2 and F_F can be found in [46]). Further, with 4 'algorithm's and 8 'data set's, F_F is distributed according to the F distribution with $4 - 1 = 3$ and $(4 - 1) \times (8 - 1) = 21$ degrees of freedom. The critical value of $F_{0.05}(3, 21)$ when $\alpha = 0.05$ is 3.0725 and $F_{0.10}(3, 21)$ when $\alpha = 0.10$ is 2.3649. As $F_F > 3.0725$ and $F_F > 2.3649$, we say for the single-view cases, the differences between all compared CK-related methods on multiple classifiers are significant. (2) Similarly, for multi-view cases, with 5 'algorithm's and 10 'data set's used, related $\chi_F^2 = 35.06$, $F_F = 63.91$, $F_{0.05}(4, 36) = 2.6335$, and $F_{0.10}(4,

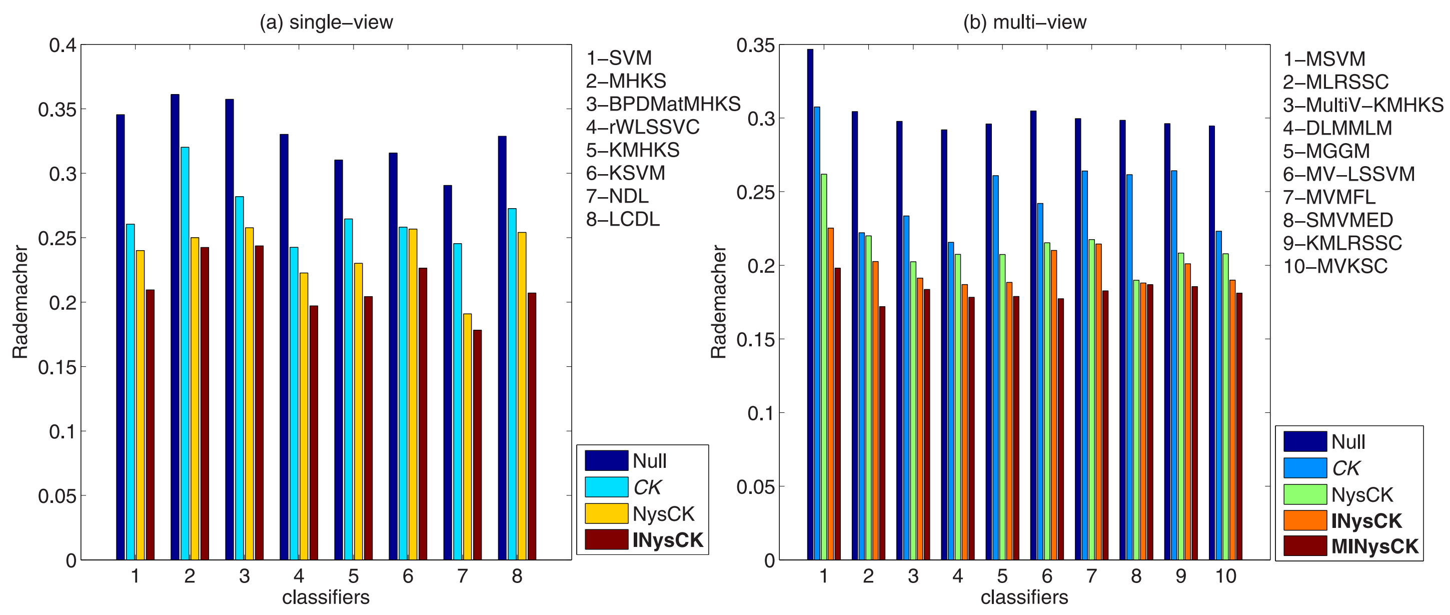

Fig 4. The average Rademacher complexity comparison.

https://doi.org/10.1371/journal.pone.0206798.g004

36) = 2.1079. Since $F_F > 2.6335$ and $F_F > 2.1079$, we can draw a conclusion that for the multi-view cases, the differences between all compared CK-related methods on multiple classifiers are also significant.

Then we use Nemenyi test for pairwise comparisons. (1) For single-view cases, when $\alpha = 0.05$, the critical value $q_{0.05}$ is 2.569 (see Table 9) and the corresponding CD is $2.569\sqrt{\frac{4\cdot(4+1)}{6\cdot 8}} = 1.66$. When $\alpha = 0.10$, the critical value $q_{0.10}$ is 2.291 (see Table 9) and the corresponding CD is $2.291\sqrt{\frac{4\cdot(4+1)}{6\cdot 8}} = 1.48$. Then according to the principle of Nemenyi test, since $1.84 < 1.16 + 1.66 = 2.82 < 3.84$ and $1.84 < 1.16 + 1.48 = 2.64 < 3.84$, so we say the differences between INysCK and CK (NysCK) are (not) significant. (2) For multi-view cases, according to Table 9, since $q_{0.05} = 2.728$ and $q_{0.10} = 2.459$, the corresponding CDs are $2.728\sqrt{\frac{5\cdot(5+1)}{6\cdot 10}} = 1.93$ and $2.459\sqrt{\frac{5\cdot(5+1)}{6\cdot 10}} = 1.74$ respectively. Then since $1.56 < 3.34 < 1.44 + 1.93 = 3.37$ and

Table 8. Average rank comparisons for different CK-related methods and classifiers.

single-view	Null	*CK*	NysCK	INysCK	multi-view	Null	*CK*	NysCK	INysCK	MINysCK
SVM	3.25	3.75	1.75	1.25	MSVM	4.80	4.00	3.20	1.60	1.40
MHKS	3.00	4.00	2.00	1.00	MLRSSC	5.00	3.60	3.40	1.40	1.60
BPDMatMHKS	3.25	3.75	1.75	1.25	MultiV-KMHKS	5.00	3.60	3.40	1.60	1.40
rWLSSVC	3.25	3.75	1.75	1.25	DLMMLM	5.00	3.60	3.40	1.60	1.40
KMHKS	3.00	4.00	2.00	1.00	MGGM	4.80	4.00	3.20	1.60	1.40
KSVM	3.25	3.75	1.75	1.25	MV-LSSVM	5.00	3.60	3.40	1.60	1.40
NDL	3.25	3.75	1.75	1.25	MVMFL	4.60	4.20	3.20	1.60	1.40
LCDL	3.00	4.00	2.000	1.000	SMVMED	5.00	3.60	3.40	1.60	1.40
Average	3.16	3.84	1.84	1.16	KMLRSSC	4.80	3.80	3.40	1.60	1.40
					MVKSC	4.80	3.80	3.40	1.40	1.60
					Average	4.88	3.78	3.34	1.56	1.44

https://doi.org/10.1371/journal.pone.0206798.t008

Table 9. Critical values for the two-tailed Nemenyi test.

No. algorithms	2	3	4	5	6	7	8	9	10
$q_{0.05}$	1.960	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164
$q_{0.10}$	1.645	2.052	2.291	2.459	2.589	2.693	2.780	2.855	2.920

https://doi.org/10.1371/journal.pone.0206798.t009

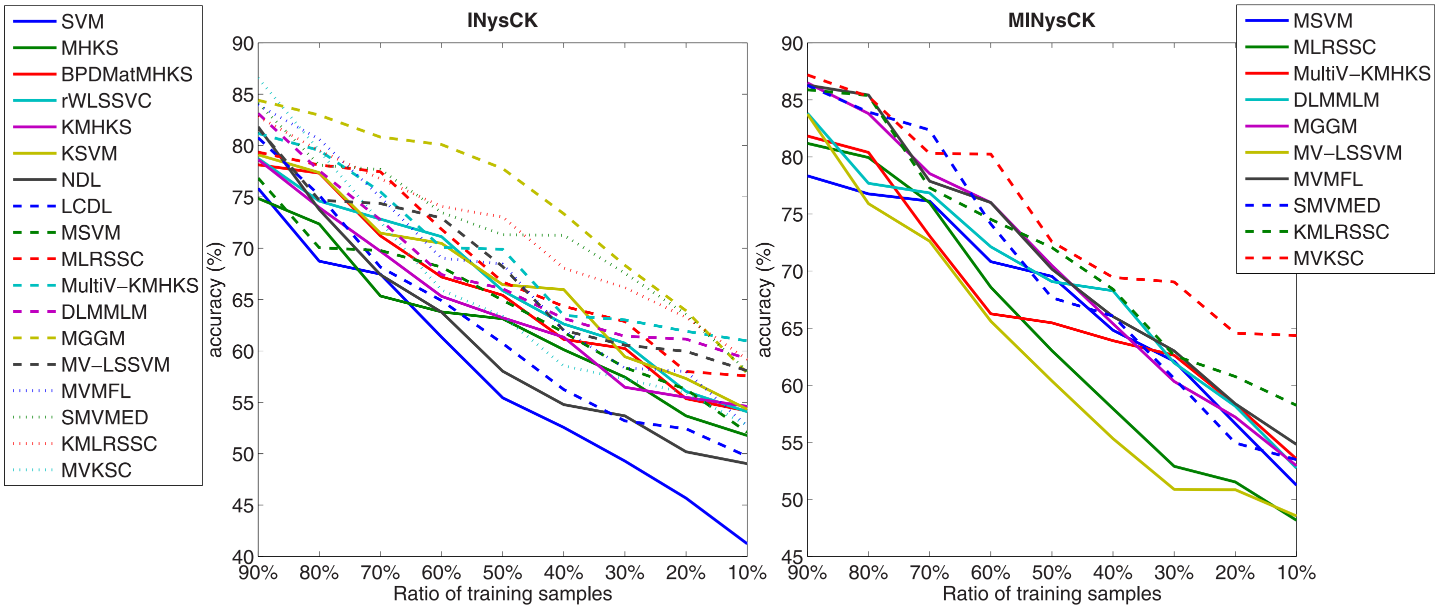

Fig 5. Average influence of ratio of training samples on accuracy with INysCK and MINysCK and corresponding classifiers used.

https://doi.org/10.1371/journal.pone.0206798.g005

$1.56 < 1.44 + 1.74 = 3.18 < 3.34$, we say the differences between MINysCK and CK are significant and the ones between MINysCK and NysCK are significant to a certain extent.

As a summary, we can draw a conclusion that according to Friedman-Nemenyi statistical test, our proposed INysCK (or MINysCK) is an improvement on previous work CK (or NysCK) statistically.

Influence of ratio of training samples. In the previous experiments, for each data set, we randomly choose 70% of samples as training part and the remaining as test part. Here, we change the ratio of training samples and show its average influence on accuracy with Fig 5. According to this figure, it is found that with the increasing of the ratio of training samples, the average accuracy also boosts.

Conclusions and future work

Conclusions

Traditional nonlinear classifiers are developed to process nonlinearly separable data sets and they always use kernel functions to generate several kernel matrices. After the optimization of these matrices, the optimal classifier parameters can be gotten. While one always costs high computational and space complexities to compute and store these matrices, so in order to reduce the complexities, people develop INMKMHKS which adopts Nyström approximation technique and NysCK which changes nonlinearly separable samples to linearly ones. In this work, we combine ideas of them in together to develop INysCK and MINysCK to reduce the complexities further and process single-view data sets and multi-view data sets respectively. In order to validate the effectiveness of them, we use CK and NysCK for comparisons. Then we adopt some large-scale, small-scale, single-view, multi-view data sets and single-view, multi-view, nonlinear, linear classifiers for experiments in niche targeting. Corresponding experiments about accuracy, time cost, convergence, Rademacher complexity, and so on have validated the effectiveness of INysCK and MINysCK.

According to experimental results, we can draw the following conclusions. (1) INysCK and MINysCK can change nonlinearly separable samples to be linearly separable with higher linearities and the accuracies of corresponding classifiers boost. (2) Compared with NysCK, INysCK and MINysCK both cost longer time in average while the increased time is acceptable. (3) With INysCK and MINysCK used, classifiers can converge faster and their Rademacher complexities are smaller. (4) INysCK (or MINysCK) is an improvement on previous work CK (or NysCK) statistically.

Future work

Although our proposed methods perform better for nonlinear classification problems, according to [47] said, Nyström approximation technique is data-dependent even though we adopt the Nyström approximation technique used in INMKMHKS to avoid parameter setting problems. In [47], on the base of Hellinger's kernel and χ^2 kernel, scholars use two mapping functions which are both data-independent to enhance the classification performance. Thus, in our future work, we try to introduce the idea of [47] to our work. In other words, we will try to use data-independent mapping functions to change the nonlinearly separable samples to the linearly separable ones. What's more, besides what we have discussed in this work, there are some other pattern recognition fields attract scholars to research, for example, unsupervised feature selection [48, 49] and multi-label learning [50]. So in our future work, we will try to introduce our methods into these fields.

Acknowledgments

This work is supported by (1) Natural Science Foundation of Shanghai under grant numbers 16ZR1414500, 16ZR1414400 (2) National Natural Science Foundation of China under grant numbers 61602296, 41701523 (3) Shanghai Pujiang talent plan under grant number 16PJ1403700 and the authors would like to thank their supports.

Author Contributions

Conceptualization: Changming Zhu.

Data curation: Xiafen Zhang.

Formal analysis: Changming Zhu.

Funding acquisition: Changming Zhu.

Investigation: Xiang Ji, Lai Wei.

Methodology: Changming Zhu, Rigui Zhou.

Software: Chao Chen.

Writing – original draft: Changming Zhu.

Writing – review & editing: Xiang Ji.

References

1. Vapnik V. Statistical Learning Theory. New York: John Wiley & Sons; 1998.
2. Hassan MF, Abdelqader I. Improving pattern classification by nonlinearly combined classifiers. International Conference on Cognitive Informatics & Cognitive Computingt. 2016;489–495.
3. Zhu XB, Pedrycz W, Li Z. Fuzzy clustering with nonlinearly transformed data. Applied Soft Computing. 2017; 61:364–376. https://doi.org/10.1016/j.asoc.2017.07.026
4. Leski J. Kernel Ho-Kashyap classifier with generalization control. International Journal of Applied Mathematics and Computer Science. 2004; 14(1):53–61.
5. Zhang K, Lan L, Wang Z, Moerchen F. Scaling up kernel SVM on limited resources: A low-rank linearization approach. Conference on Artificial Intelligence and Statistics (AISTATS). 2012;22:1425–1434.
6. Wang Z, Chen SC. Multi-view kernel machine on single-view data. Neurocomputing. 2009; 72:2444–2449. https://doi.org/10.1016/j.neucom.2008.11.017
7. Wang Z, Chen SC, Gao DQ. A novel multi-view learning developed from single-view patterns. Pattern Recognition. 2011; 44(10–11):2395–2413. https://doi.org/10.1016/j.patcog.2011.04.002
8. Wang Z, Xu J, Chen SC, Gao DQ. Regularized multi-view machine based on response surface technique. Neurocomputing. 2012; 97:201–213. https://doi.org/10.1016/j.neucom.2012.05.027
9. Zhu CM, Wang Z, Gao DQ, Feng X. Double-fold localized multiple matrixized learning machine. Information Sciences. 2015; 295:196–220. https://doi.org/10.1016/j.ins.2014.10.024
10. Zhu CM. Double-fold localized multiple matrix learning machine with Universum. Pattern Analasis and Application. 2017; 20(4):1091–1118. https://doi.org/10.1007/s10044-016-0548-9
11. Zhu CM, Gao DQ. Improved multi-kernel classification machine with Nyström approximation technique. Pattern Recognition. 2015; 48:1490–1509. https://doi.org/10.1016/j.patcog.2014.10.029
12. Zhou ZL, Wang YL, Wu QMJ, Yang CN, Sun XM. Effective and Efficient Global Context Verification for Image Copy Detection. IEEE Transactions on Information Forensics and Security. 2017; 12(1):48–63. https://doi.org/10.1109/TIFS.2016.2601065
13. Xia ZH, Wang XH, Zhang LG, Qin Z, Sun XM, Ren K. A Privacy-preserving and Copy-deterrence Content-based Image Retrieval Scheme in Cloud Computing. IEEE Transactions on Information Forensics and Security. 2016; 11(11):2594–2608. https://doi.org/10.1109/TIFS.2016.2590944
14. Wang Z, Zhu CM, Niu ZX, Gao DQ, Feng X. Multi-kernel classification machine with reduced complexity. Knowledge-Based Systems. 2014; 65:83–95. https://doi.org/10.1016/j.knosys.2014.04.012
15. Hou BJ, Zhang LJ, Zhou ZH. Storage Fit Learning with Unlabeled Data. Twenty-Sixth International Joint Conference on Artificial Intelligence. 2017;1844–1850.
16. Ye HJ, Zhan DC, Miao Y, Jiang Y, Zhou ZH. Rank consistency based multi-view learning: a privacy-preserving approach. ACM International on Conference on Information and Knowledge Management. 2015;991–1000.
17. Sharma A, Kumar A, Daume H, Jacobs DW. Generalized multiview analysis: a discriminative latent space. IEEE Conference on Computer Vision and Pattern Recognition. 2012;157:2160–2167.
18. Wang W, Zhou ZH. Multi-view active learning in the nonrealizable case. Neural Information Processing System. 2010; 23:2388–2396.
19. Williams CKI, Seeger M. Using the Nyström method to speed up kernel machines. Conference on Neural Information Processing Systems. 2000;661–667.
20. Faraki M, Harandi MT, Porikli FM. Approximate infinite-dimensional region covariance descriptors for image classification. IEEE International Conference on Acoustics, Speech and Signal Processing. 2015;1364–1368.
21. Iosifidis A, Gabbouj M. Nyström-based approximate kernel subspace learning. Pattern Recognition. 2016; 57:190–197. https://doi.org/10.1016/j.patcog.2016.03.018
22. Li LC, Wang SL, Xu SJ, Yang YQ. Constrained spectral clustering using Nyström method. Procedia Computer Science. 2018; 129:9–15. https://doi.org/10.1016/j.procs.2018.03.036
23. Frank A, Asuncion A. UCI machine learning repository (http://archive.ics.uci.edu/ml). Irvine: University of California; 2010.
24. http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm
25. http://archive.ics.uci.edu/ml/datasets/youtube+multiview+video+games+dataset
26. http://dblp.uni-trier.de/xml/
27. Wang SK, Wang EK, Li XT, Ye YM, Lau RYK, Du XL. Multi-view learning via multiple graph regularized generative model. Knowledge-Based Systems. 2017; 121:153–162. https://doi.org/10.1016/j.knosys.2017.01.022

28. http://www.cs.umd.edu/sen/lbc-proj/data/cora.tgz
29. Jacob Y, Denoyer L, Gallinari P. Classification and annotation in social corpora using multiple relations. Proceedings of the 20th ACM International Conference on Information and Knowledge Management. 2011;1215–1220.
30. Milgram J, Cheriet M, Sabourin R. "One against one" or "one against all": which one is better for handwriting recognition with SVMs?. Tenth International Workshop on Frontiers in Handwriting Recognition. 2013.
31. Debnath R, Takahide N, Takahashi H. A decision based one-against-one method for multi-class support vector machine. Pattern Analysis and Applications. 2004; 7(2):164–175. https://doi.org/10.1007/s10044-004-0213-6
32. Hsu CW, Lin CJ. A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks. 2002; 13(2):415–425. https://doi.org/10.1109/72.991427 PMID: 18244442
33. Cortes C, Vapnik V. Support vector machine. Machine Learning. 1995; 20(3):273–297. https://doi.org/10.1007/BF00994018
34. Rice I. Improved data visualisation through nonlinear dissimilarity modelling. Pattern Recognition. 2018; 73:76–88. https://doi.org/10.1016/j.patcog.2017.07.016
35. Liu L, Ma SW, Rui L, Lu J. Locality constrained dictionary learning for non-linear dimensionality reduction and classification. IET Computer Vision. 2017; 11(1):60–67. https://doi.org/10.1049/iet-cvi.2015.0482
36. Wang Z, Zhu ZH. Matrix-pattern-oriented classifier with boundary projection discrimination. Knowledge-Based Systems. 2018; 149:1–17.
37. Yang B, Shao QM, Pan L, Li WB. A study on regularized weighted least square support vector classifier. Pattern Recognition Letters. 2018; 108:48–55. https://doi.org/10.1016/j.patrec.2018.03.002
38. Huang CQ, Chung FL, Wang ST. Multi-view L2-SVM and its multi-view core vector machine. Neural Networks. 2016; 75:110–125. https://doi.org/10.1016/j.neunet.2015.12.004 PMID: 26773824
39. Brbić M, Kopriva I. Multi-view low-rank sparse subspace clustering. Pattern Recognition. 2018; 73: 247–258. https://doi.org/10.1016/j.patcog.2017.08.024
40. Houthuys L, Langone R, Suykens JAK. Multi-view least squares support vector machines classification. Neurocomputing. 2018; 282:78–88. https://doi.org/10.1016/j.neucom.2017.12.029
41. Li JX, Zhang B, Lu GM, Zhang D. Generative multi-view and multi-feature learning for classification. Information Fusion. 2019; 45:215–226. https://doi.org/10.1016/j.inffus.2018.02.005
42. Chao GQ, Sun SL. Semi-supervised multi-view maximum entropy discrimination with expectation Laplacian regularization. Information Fusion. 2019; 45:296–306. https://doi.org/10.1016/j.inffus.2018.03.002
43. Houthuys L, Langone R, Suykens JAK. Multi-view kernel spectral clustering. Information Fusion. 2018; 44:46–56. https://doi.org/10.1016/j.inffus.2017.12.002
44. Zhu CM, Wang Z. Entropy-based matrix learning machine for imbalanced data sets. Pattern Recognition Letters. 2017; 88:72–80. https://doi.org/10.1016/j.patrec.2017.01.014
45. Ye J. Generalized low rank approximations of matrices. Machine Learning. 2005; 61(1–3):167–191. https://doi.org/10.1007/s10994-005-3561-6
46. Demsar J. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research. 2006; 7:1–30.
47. Wang QL, Li PH, Zuo WM, Zhang L. RAID-G: robust estimation of approximate infinite dimensional Gaussian with application to material recognition. IEEE Conference on Computer Vision and Pattern Recognition. 2016;4433–4441.
48. Wang WZ, Zhang HZ, Zhu PF, Zhang D, Zuo WM. Non-convex regularized self-representation for unsupervised feature selection. Proceedings of International Conference on Intelligent Science and Big Data Engineering (part II). 2015;55–65.
49. Zhu PF, Zhu WC, Hu QH, Zhang CQ, Zuo WM. Subspace clustering guided unsupervised feature selection. Pattern Recognition. 2017; 66:364–374. https://doi.org/10.1016/j.patcog.2017.01.016
50. Zhu PF, Xu Q, Hu QH, Zhang CQ, Zhao H. Multi-label feature selection with missing labels. Pattern Recognition. 2018; 74:488–502. https://doi.org/10.1016/j.patcog.2017.09.036