Pattern Recognition 41 (2008) 3366 – 3372

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

An efficient kernel matrix evaluation measure

Canh Hao Nguyen*, Tu Bao Ho

School of Knowledge Science, Japan Advanced Institute of Science and Technology, 1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan

ARTICLE INFO

Article history:
Received 16 October 2007
Accepted 3 April 2008

Keywords:
Classification
Kernel methods
Kernel matrix quality measure
Kernel target alignment
Class separability measure

ABSTRACT

We study the problem of evaluating the goodness of a kernel matrix for a classification task. As kernel matrix evaluation is usually used in other expensive procedures like feature and model selections, the goodness measure must be calculated efficiently. Most previous approaches are not efficient except for kernel target alignment (KTA) that can be calculated in $O(n^2)$ time complexity. Although KTA is widely used, we show that it has some serious drawbacks. We propose an efficient surrogate measure to evaluate the goodness of a kernel matrix based on the data distributions of classes in the feature space. The measure not only overcomes the limitations of KTA but also possesses other properties like invariance, efficiency and an error bound guarantee. Comparative experiments show that the measure is a good indication of the goodness of a kernel matrix.

1. Introduction

Kernel methods, such as support vector machines (SVMs), Gaussian processes, etc., have delivered extremely high performance in a wide variety of supervised and nonsupervised learning tasks [1]. The key to success is that kernel methods can be modularized into two modules: the first is to map data into a (usually higher dimensional) feature space; and the second is to use a linear algorithm in the feature space, which is efficient and has theoretical guarantees [2]. The process of kernelization of linear algorithms makes them run in a time complexity that is independent of the dimension of the input space. It allows to introduce nonlinearity into the algorithms implicitly by the kernel maps. It also allows inputs to be of arbitrary types provided that kernels can be constructed [1–4].

While many linear algorithms can be kernelized, the major effort in kernel methods is put into designing good mappings defined by kernel functions ϕ:

$$\phi : X \to H. \tag{1}$$

X is the original data space and H is the feature space, which is usually chosen to be a reproducing kernel Hilbert space (RKHS) [5]. The reasons are that RKHS is small enough compared to Hilbert space so that it does not contain many nonsmooth functions and it is large enough to contain the optimal function. While H may be a many-dimensional space, the algorithm takes advantage of the kernel trick to operate solely on the kernel matrix induced from data:

$$K = \{\langle \phi(x_i), \phi(x_j) \rangle\}_{i=1\ldots n, j=1\ldots n}. \tag{2}$$

The goodness of a kernel function can only be seen from the goodness of the kernel matrix K; therefore, measuring the goodness of K is of primary interest in various contexts.

There are many ways to measure the goodness of a kernel matrix (kernel function) for a classification task, differently reflecting the expected quality of the kernel function. Commonly used measures are regularized risk [1], negative log-posterior [6] and hyperkernels [7]. These measures do not give a specific value, but only assert certain criteria in form of regularities in certain spaces, for example, RKHS or hyper-RKHS. All of these kernel measures require an optimization procedure for evaluation. Therefore, they are prohibitively expensive to be incorporated into other expensive procedures like model and feature selections. Other techniques like cross validation, leave-one-out estimators or radius-margin bound can also be considered as expected quality functionals. Like the previously mentioned works, they require the whole learning process.

To be used in feature and model selection processes, goodness measures of kernel matrices must be efficiently calculated. Simple formula is an additional merit as one can design kernels taking a goodness measure as an objective function. The most commonly used efficient kernel goodness measure is kernel target alignment (KTA) [8]. Due to its simplicity and efficiency, KTA has been used in many works for two central problems in kernel methods: designing kernels and learning kernels from data.

In this work, we first analyze KTA to show that having a high KTA is only a *sufficient* condition to be a good kernel matrix, but not a *necessary* condition. It is possible for a kernel matrix to have a very good performance even though its KTA is still low. We then propose an efficient surrogate measure based on the data distributions in the feature space to relax the strict conditions imposed by KTA. The measure is invariant to linear operators in the feature space and

* Corresponding author. Tel.: +81 761 51 1111x1857; fax: +81 761 51 1795.
E-mail addresses: canhhao@jaist.ac.jp (C.H. Nguyen), bao@jaist.ac.jp (T.B. Ho).

0031-3203/$30.00 © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2008.04.005

retains several properties of KTA, such as efficiency and an error bound guarantee. The measure is closely related to some other works on worst-case error bounds and distance metric learning. We show experimentally that the new measure is more closely correlated to the goodness of a kernel matrix, and conclude finally with some future works.

2. Kernel target alignment

KTA is used to measure how well a kernel matrix aligns to a target (or another kernel) [8]. Alignment to the target is defined as the (normalized) Frobenius inner product between the kernel matrix and the covariance matrix of the target vector. It is interpreted as cosine distance between these two bi-dimensional vectors. We first introduce some notations.

Denote the training example set as $\{x_i\}_{i=1\dots n} \subset X$ with the corresponding target vector $y = \{y_1|y_2|\cdots|y_n\}^{\mathrm{T}} \in \{-1, 1\}^n$. Suppose that $y_1 = \cdots = y_{n_+} = 1$ and $y_{n_++1} = \cdots = y_{n_++n_-} = -1$; n_+ examples belong to class 1, n_- examples belong to class -1, $n_+ + n_- = n$. Under a feature map ϕ, the kernel matrix is defined as

$$K = \{k_{ij} = \langle \phi(x_i), \phi(x_j)\rangle\}_{i,j=1\dots n}. \tag{3}$$

Recall that Frobenius inner product is defined as

$$\langle K, K^*\rangle_F = \sum_{i=1}^{n}\sum_{j=1}^{n} k_{ij}k^*_{ij}. \tag{4}$$

The target matrix is defined to be $y \cdot y^{\mathrm{T}}$. KTA of K is defined as the normalized Frobenius inner product:

$$A(K, y) = \frac{\langle K, y \cdot y^{\mathrm{T}}\rangle_F}{\sqrt{\langle K, K\rangle_F \langle y \cdot y^{\mathrm{T}}, y \cdot y^{\mathrm{T}}\rangle_F}}. \tag{5}$$

There are many advantageous properties of KTA that make it a popular choice for measuring the goodness of fit of a kernel matrix to a supervised task. It is efficient as its computational time is $O(n^2)$ with a simple formula, allowing it to be an objective function in an optimization procedure. It is highly concentrated around its expected value, namely

$$P(S|\hat{A}(X) - A(y) \geqslant \hat{\varepsilon}) \leqslant \delta, \tag{6}$$

where $\hat{\varepsilon} = C(X)\sqrt{8\ln(2/\delta)/n}$, $C(X)$ is nontrivial. This gives a high probability that an empirical estimate of KTA is close to the true alignment value. There is an Parzen window classifier that emits an generalization error bound of

$$\hat{A}(X) - \sqrt{\frac{8}{n}\ln\left(\frac{2}{\delta}\right)}. \tag{7}$$

This means that when an alignment is high, one can expect that the error rate using the kernel must be limited to a certain amount.

Evidently, KTA is used extensively in many kernel design and learning methods. Most popularly, KTA is directly used as an objective function to optimize in learning parameters for kernels: weighting of kernels in a multiple kernel learning framework [9]; feature selection [10]; subspace kernels for feature extraction [11]. It is also used to adapt kernels in Refs. [8,12], to design kernels with boosting [13], to learn semantic similarity [14]. It is also extended to regression problems in Ref. [15].

We claim that having a very high value $A(K, y)$ is only a sufficient condition, but not a necessary condition, for K to be a good kernel matrix for a given task specified by the target y. As $0 \leqslant A(K, y) \leqslant 1$,[1] when $A(K, y)=1$, the two bi-dimensional vectors K and $y \cdot y^{\mathrm{T}}$ are linear. Up to a constant, the optimal alignment happens when $k_{ij} = y_i \cdot y_j$. This is equivalent to two conditions:

(1) All examples of the same class are mapped into the same vector in the feature space ($k_{ij} = 1$ for x_i, x_j in the same class).
(2) The mapped vectors of different classes in the feature space are additively inverse ($k_{ij} = -1$ for x_i, x_j in different classes).

It is a sufficient condition that having a high K, one can expect good performance, as classes are collapsed into a point in the feature space. Violating any of the conditions would mean that KTA is penalized. However, it is not a necessary condition as we analyze below.

The first condition implies that the within-class variance penalizes KTA. However, there is a gap between the condition and the concept of margin in SVMs. Varying data (in the feature space) along the separating hyperplane will not affect the margin. If data varies along the perpendicular direction of the separating hyperplane, the margin can be changed. However, KTA does not take into account variances in different directions.

The second condition is too strict, not applicable in many cases because it requires the mapped vectors of different classes to be additive inverses of each other. Ideally, if the kernel function maps all examples of a class to one vector in the feature space, the kernel matrix should be evaluated as optimal. This is the reason why having high alignment is only a sufficient condition, but not a necessary condition. One of the effects of this shortcoming was also reported in Ref. [13]. The following theorem shows the source of this limitation.

Theorem 1. *KTA is not invariant under data translation in the feature space.*

Sketch of Proof. Translate data in the feature space, $\phi(x_i) \mapsto \phi(x_i) + \delta$, $\forall i = 1 \dots n$. This results in KTA to be a function of δ. Therefore, KTA is not translation invariant in the feature space. We take some examples to show this limitation quantitatively in some situations. □

Example 1 (*The best case*). The kernel function ϕ maps all examples of class 1 into vector $\phi_+ \in H$ and all examples of class -1 into vector $\phi_- \in H$. Assume that $\phi_+ \cdot \phi_+ = \phi_- \cdot \phi_- = 1$ and $\phi_+ \cdot \phi_- = \alpha$, $-1 \leqslant \alpha < 1$. For any α, the kernel matrix K should be evaluated as optimal, as it is induced from an optimal feature map. However, its alignment value is

$$A(K, y) = \frac{n_+^2 + n_-^2 - 2n_+n_-\alpha}{\sqrt{n_+^2 + n_-^2 + 2n_+n_-\alpha^2 \cdot n}}. \tag{8}$$

Alignment values of these kernel matrices change as α varies from 1 to -1, and any value in that range can be the alignment value of a kernel matrix of an optimal feature function. As $\lim_{\alpha\to 1} A(K, y) = (n_+ - n_-)^2/n^2$, KTA ranges from $(n_+ - n_-)^2/n^2$ to 1 in this case.

Example 2 (*The worst case*). The kernel function ϕ maps a half of the examples of each class into vector $\phi_1 \in H$ and the other half into vector $\phi_2 \in H$. Assume that $\phi_1 \cdot \phi_1 = \phi_2 \cdot \phi_2 = 1$ and $\phi_1 \cdot \phi_2 = \alpha$, $-1 \leqslant \alpha \leqslant 1$. For any α, the kernel matrix K should be evaluated very

[1] In Ref. [8], it is $-1 \leqslant A(K, y)$. However, both K and $y \cdot y^{\mathrm{T}}$ are positive semidefinite. Hence, both K and $y \cdot y^{\mathrm{T}}$ lie in the positive semidefinite cone, making the cosine of the angle and dot product between them nonnegative [16]. The inequality $0 \leqslant A(K, y)$ follows.

low, as it is induced from the worst kernel function, which fuses the two classes together and has no generalization ability. However, its alignment value is

$$A(K,y)=\frac{(n_+-n_-)^2\cdot(1+\alpha)/2}{n^2*\sqrt{(1+\alpha^2)/2}}. \tag{9}$$

As shown above, $\lim_{\alpha\to 1}A(K,y)=(n_+-n_-)^2/n^2$, which is the same limit as the best case. We can see that KTA of this case ranges from 0 to $(n_+-n_-)^2/n^2$. There is a similar caution in Ref. [17]. As the best and the worst cases cover the whole range of alignment values, any other case would coincide with one of them. Therefore, KTA may mistake any case to be either the best or the worst.

Example 3 (*The popular case*). Consider the case when a kernel function maps all examples into an orthant in the feature space, inducing a kernel matrix with nonnegative entries, i.e. $k_{ij}\geqslant 0$. In this case

$$A(K,y)\leqslant\frac{\sqrt{n_+^2+n_-^2}}{n}. \tag{10}$$

Proof can be derived by using the fact that $k_{ij}\geqslant 0$ and the Cauchy–Schwarz inequality [18]. Take a special case when $n_+=n_-$, $A(K,y)\leqslant 1/\sqrt{2}$. Recall that KTA of a kernel matrix ranges from 0 to 1, and the higher the better. This example means that these types of kernel functions will have bounded alignment values, no matter how good the kernel functions are. This is a drawback of KTA. Unfortunately, this type of kernel function is extensively used in practice. Gaussian kernels [2] and many other kernels defined over discrete structures [19] or distributions [20], etc. fall into this category.

The above examples show that KTA can mistake any kernel matrix to be the best or the worst. KTA is always bound to some numbers for many popular kernel matrices with positive entries (e.g., Gaussian and many kernels for discrete structures or distributions). KTA will disadvantage the use of such kernels.

3. Feature space-based kernel matrix evaluation measure

In this section, we introduce a new goodness measure of a kernel matrix for a given (binary classification) task, named feature space-based kernel matrix evaluation measure (FSM). This measure should be computed on the kernel matrix efficiently, and overcome the limitations of KTA. Our idea is to use the data distributions in the feature space. Specifically, two factors are taken into account: (1) *the within-class variance* in the direction between-class centers; and (2) *the distance between the class centers*. These factors are depicted in Fig. 1. The first factor improves the first condition of KTA, by allowing data to vary in certain directions. The second factor solves the problem imposed by the second condition of KTA. Let *std* be the total within-class standard deviation (for both classes) in the direction between-class centers. Denote the center of a class as the mean of the class data in the feature space: $\phi_+=\sum_{i=1}^{n_+}\phi(x_i)/n_+$ and

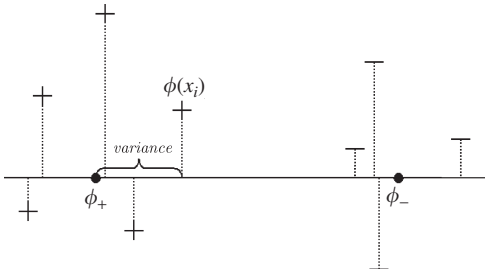

Fig. 1. Data spreading in the direction between-class centers.

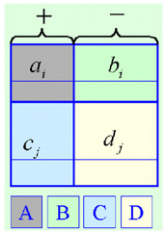

Fig. 2. Visual description of auxiliary variables on the kernel matrix.

$\phi_-=\sum_{i=n_++1}^{n}\phi(x_i)/n_-$. Concretely, the evaluation measure FSM is defined to be *the ratio of the total within-class standard deviation in the direction between the class centers to the distance between the class centers*:

$$FSM(K,y)\stackrel{\text{def}}{=}\frac{std}{\|\phi_--\phi_+\|}. \tag{11}$$

3.1. FSM calculation

We show that the evaluation measure FSM can be calculated using the kernel matrix efficiently with simple formulas. We first calculate the within-class standard deviation (of both classes) in the direction between-class centers. Denote $e=\phi_--\phi_+/\|\phi_--\phi_+\|$ as the unit vector in this direction. The total within-class standard deviation of two classes in the direction is

$$std=\sqrt{\frac{\sum_{i=1}^{n_+}\langle\phi(x_i)-\phi_+,e\rangle^2}{n_+-1}}+\sqrt{\frac{\sum_{i=n_++1}^{n}\langle\phi(x_i)-\phi_-,e\rangle^2}{n_--1}}. \tag{12}$$

We calculate the first term in Eq. (12), the total within-class standard deviation of the class +1 in the direction between ϕ_+ and ϕ_-, denoted as std_+.

$$\begin{aligned}(n_+-1)std_+^2&=\sum_{i=1}^{n_+}\langle\phi(x_i)-\phi_+,e\rangle^2\\&=\frac{\sum_{i=1}^{n_+}\langle\phi(x_i)-\phi_+,\phi_--\phi_+\rangle^2}{(\phi_--\phi_+)^2}\\&=\frac{\sum_{i=1}^{n_+}(\phi(x_i)\phi_-+\phi_+^2-\phi(x_i)\phi_+-\phi_+\phi_-)^2}{(\phi_--\phi_+)^2}.\end{aligned} \tag{13}$$

We substitute $\phi_+=\sum_{j=1}^{n_+}\phi(x_j)/n_+$ and $\phi_-=\sum_{j=n_++1}^{n}\phi(x_j)/n_-$ into Eq. (13), and then we define some auxiliary variables as follows (Fig. 2). For $i=1\ldots n_+$:

- $a_i=\phi(x_i)\phi_+=\sum_{j=1}^{n_+}\phi(x_i)\phi(x_j)/n_+=\sum_{j=1}^{n_+}k_{ij}/n_+$,
- $b_i=\phi(x_i)\phi_-=\sum_{j=n_++1}^{n}\phi(x_i)\phi(x_j)/n_-=\sum_{j=n_++1}^{n}k_{ij}/n_-$.

For $i=n_++1\ldots n$:

- $c_i=\phi(x_i)\phi_+=\sum_{j=1}^{n_+}\phi(x_i)\phi(x_j)/n_+=\sum_{j=1}^{n_+}k_{ij}/n_+$,
- $d_i=\phi(x_i)\phi_-=\sum_{j=n_++1}^{n}\phi(x_i)\phi(x_j)/n_-=\sum_{j=n_++1}^{n}k_{ij}/n_-$.

Denote

$$A = \frac{\sum_{i=1}^{n_+} a_i}{n_+},$$
$$B = \frac{\sum_{i=1}^{n_+} b_i}{n_+},$$
$$C = \frac{\sum_{i=n_++1}^{n} c_i}{n_-},$$
$$D = \frac{\sum_{n_++1}^{n} d_i}{n_-}.$$

Hence,

$$A = \phi_+\phi_+,$$
$$B = C = \phi_+\phi_-,$$
$$D = \phi_-\phi_-.$$

Therefore,

$$(\phi_- - \phi_+)^2 = A + D - B - C.$$

Plugging them into Eq. (13), then

$$(n_+ - 1)std_+^2 = \frac{\sum_{i=1}^{n_+} (b_i - a_i + A - B)^2}{A + D - B - C}. \quad (14)$$

We can use a similar calculation for the second term in Eq. (12):

$$(n_- - 1)std_-^2 = \frac{\sum_{i=n_++1}^{n} (c_i - d_i + D - C)^2}{A + D - B - C}. \quad (15)$$

The proposed kernel matrix evaluation measure FSM, as defined in formula (11), is calculated as

$$FSM(K, y) = \frac{std_+ + std_-}{\sqrt{A + D - B - C}}. \quad (16)$$

$FSM(K, y) \geqslant 0$, and the smaller $FSM(K, y)$ is, the better K is.

One can easily see that a_i, b_i, c_i, d_i, A, B, C and D can be calculated all together in $O(n^2)$ time complexity (one pass through the kernel matrix). Therefore, the evaluation measure is also efficiently calculated in $O(n^2)$ time complexity.

3.2. Invariance

There are some properties of $FSM(K, y)$ regarding linear operations that make it closer to SVMs than KTA is.

Theorem 2. *$FSM(K, y)$ is invariant under translation, rotation and scale in the feature space.*

Sketch of Proof. $FSM(K, y)$ is scale invariant owing to its normalization factor $A + D - B - C$. $FSM(K, y)$ is rotation invariant, as it is built on k_{ij}-s, which are rotation invariant. It is translation invariant as $A - B$, $D - C$ are subtracted from the numerator. □

The performance of SVMs and other kernel methods is unchanged under rotation, translation and scale in the feature space. Therefore, it is reasonable to ask for measures to have these properties. However, KTA is not translation invariant.

3.3. Error bound

We show that for any kernel matrix, it is possible to obtain a training error rate, which is bounded by some amount proportionate to $FSM(K, y)$. This means that a low $FSM(K, y)$ value can guarantee a low training error rate. In this case, we can expect a low generalization error rate.

Theorem 3. *There exists a separating hyperplane such that its training error is bounded by*:

$$FSMerr \stackrel{\text{def}}{=} \frac{FSM(K, y)^2}{1 + FSM(K, y)^2}. \quad (17)$$

Proof. We use an one-tailed version of Chebyshev's inequality [21]. Consider the data distribution in the direction between-class centers. For each class, the data distribution has a mean of either ϕ_+ or ϕ_-, and the corresponding standard deviation std_+ or std_-. The following inequalities can be derived:

$$P(\langle \phi - \phi_-, -e\rangle \geqslant k \cdot std_- | \phi \in class(-)) \leqslant \frac{1}{1 + k^2},$$
$$P(\langle \phi - \phi_+, e\rangle \geqslant k \cdot std_+ | \phi \in class(+)) \leqslant \frac{1}{1 + k^2}. \quad (18)$$

The separating hyperplane, which takes e as its norm vector and intersects the line segment between ϕ_+ and ϕ_- at the point h such that $|h - \phi_+|/|h - \phi_-| = std_+/std_-$, has the formula:

$$f(x) = e \cdot x - e \cdot \frac{std_-\phi_+ + std_+\phi_-}{std_+ + std_-} = 0. \quad (19)$$

The error rate of this hyperplane for each class is bounded using the inequalities in Eq. (18) with $k = |\phi_- - \phi_+|/(std_+ + std_-) = 1/FSM(K, y)$. Therefore, the total training error rate on both classes is also bounded by

$$\frac{1}{1 + k^2} = \frac{FSM(K, y)^2}{1 + FSM(K, y)^2}. \quad \square \quad (20)$$

3.4. Discussion

The measure FSM can be interpreted as the ratio of within-class variance to between-class variance. It indicates how well the two classes are separated. It is advantageous over KTA because it takes into account the within-class variance (namely standard derivation) at a finer scale, and relaxes the strict conditions of KTA by considering relative positions of classes in the feature space. For the examples in Section 2, FSM values of the best cases are always ∞, those of the worst cases are always 0, and for the popular case, there are no bounds. This measure also has some similarities with other works.

Uneven data problem: Uneven data (a.k.a. imbalanced data problem [22]) may cause classifiers to perform poorly on the under-represented class. Therefore, KTA is modified for this situation by weighting the alignment according to numbers of training examples of classes, namely $1/n_+$ and $-1/n_-$ [15]. However, by taking (directional) standard derivation, FSM naturally deals with uneven data problem in a similar fashion.

Minimax Probability Machine: The minimax probability machine (MPM) [23] controls misclassification probability in the worst-case setting. The worst-case misclassification probability is defined to be

$$\min_a k(a) = \frac{\sqrt{a^T Cov_+ a} + \sqrt{a^T Cov_- a}}{\langle a, \phi_+ - \phi_-\rangle}, \quad (21)$$

where Cov_+ and Cov_- are covariance matrices of the $+1$ and -1 class, respectively. It is minimized with respect to a using Semidefinite Programming [16]. The worst case is determined by making no assumption rather than the mean and the covariance matrix of each class. It is a property of covariance matrices that $\sqrt{a^T Cov_+ a}$ is the directional standard deviation of data in the $+1$ class on the dimension spanned by a, $\sqrt{a^T Cov_+ a}$ is the directional standard deviation of

data in the -1 class. The objective function of MPM is similar to FSM in the sense that our measure also shows the worst-case misclassification error. The difference is that the objective of MPM considers the data spread in all directions (any a) using the covariance matrix, while FSM takes only the standard derivation in the direction between the class centers ($a = \alpha \cdot (\phi_+ - \phi_-), \alpha \in R$). That is the reason why FSM is a lightweight version of the objective of MPM and can be calculated efficiently. A special case of MPM is when the a, which minimizes (21), satisfies ($a = \alpha \cdot (\phi_+ - \phi_-), \alpha \in R$), FSM is the optimal result of MPM.

Class separability measure: The work that can be considered to be in between MPM and FSM is class separability measure (CSM) [24]. It is similar to the objective function of MPM in the sense that it considers data variance in all directions. However, being an efficient measure, it is different that only trace of the covariance matrix is taken into account:

$$CSM = c \cdot \frac{trace(Cov_+) + trace(Cov_-)}{\|\phi_+ - \phi_-\|^2} \tag{22}$$

for some constant c (after some manipulation of the denominator). It is different from FSM that data variance in all directions is taken into account (by using traces). However, variance in all directions are countable for class separability. For that, CSM is not known to have any error bound as FSM. One typical case is that when data are multimodal [25], data variance from different clusters is not informative. Projecting them into the direction between-class centers might be an option to cancel out this effect. This is another advantage of FSM over CSM. We will show that CSM is not a reliable quality measure in the second experiment later.

Distance learning for nearest neighbor: In the context of nearest neighbor classification, one of the criteria of learning a good distance function is to transform data such that examples from the same class are collapsed into one point [26]. Our measure also shows quantitatively how much a class collapses. However, the key difference is that in their method, the objective is the whole class collapses to one point, while our measure shows how the whole class collapses to a hyperplane. This difference explains why their method is applied to nearest neighbor classification, while our measure is for kernel methods.

Fisher discriminant analysis: FSM is akin to kernel Fisher discriminant analysis (KFDA) [27] in the sense that they both measure the ratio of data variance in one direction to data variance of class centers. While Fisher criterion is defined to be the maximal ratio in all directions, FSM is the ratio of at projection with other properties as analyzed above.

4. Experiments

As the purpose of these measures is to predict efficiently how good a kernel matrix is for a given task using kernel methods, we compare the measures to the de facto standard of cross validation error rates of SVMs. We mimicked the model selection process by choosing different kernels. We monitored the error rates (as baselines), KTA and FSM, to see how they reflect the baselines. We used 10-times fivefold stratified cross validations to estimate the error rates.

To facilitate visual inspection of the results, we instead showed the following quantities: (a) $1 - KTA$, (b) $CSMnorm \stackrel{\text{def}}{=} CSM/(CSM + 1)$,[2] (c) *FSMerr*, defined as error bound based on FSM measure in formula (17), and (d) *error*, cross validation error rates. The reason is that all of these quantities range from 0 to 1 and relate to the expected error rates of the kernel function in some sense. The reason we chose *CSMnorm* is that we want to normalize it to make it ranging from 0 to 1. We showed CSM because it is also an efficient kernel matrix quality measure. Normalization of CSM is in a similar fashion as *FSMerr* to FSM. The smaller their values, the better the kernel. The first three quantities are expected to be correlated to the last one.

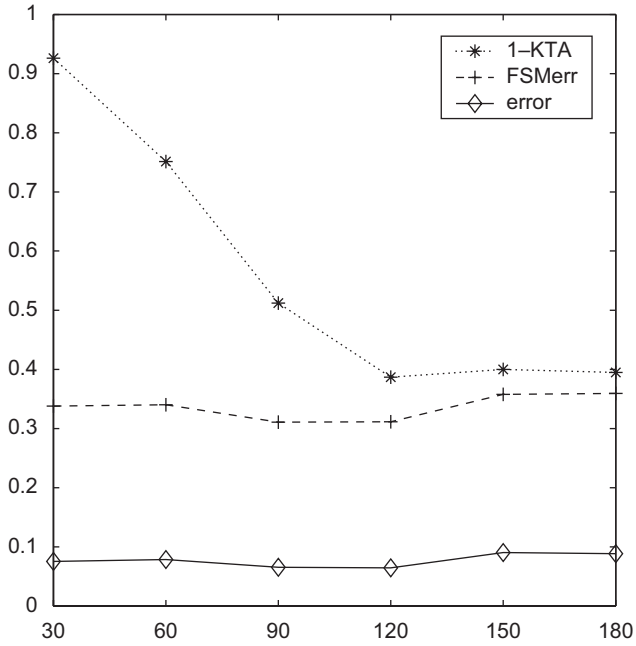

Fig. 3. Results on synthetic data with different β values.

4.1. Synthetic data

We generated synthetic data in R^2, and used linear kernels to simulate different data distributions by different kernels in a feature space. For each data set parameterized by an angle β, we used two isotropic Gaussian distributions (fixed standard derivation in all directions) for two classes centered at $\phi_+ = (1, 0) \in R^2$ for class $+1$ and at $\phi_- = (\cos(\beta), \sin(\beta)) \in R^2$ for class -1. Each class contains 500 training examples. The standard derivations of the Gaussian distributions are determined to be $var_+ = var_- = \frac{1}{2}\|\phi_- - \phi_+\|$. This ensures that for any β, any data set can be images of another after a linear operation. Therefore, these data sets with linear kernels are equivalent to one data set with different kernel functions. For any $\beta > 0$, the problems should have the same level of error rates (or other measures) when using linear kernels, i.e. linear kernels should be evaluated at the same level of goodness. We run experiments with different β values of 30°, 60°, 90°, 120°, 150° and 180° in turn. The results of $1 - KTA$, *FSMerr* and *error* are shown in Fig. 3.

From Fig. 3, we can observe that *error* is stable across different β's, as we expected. *FSMerr* is also rather stable, varying similarly to *error*. $1 - KTA$ changes dramatically from 0.936 down to 0.395. It can be concluded that KTA is too sensitive to absolute positions of data in the feature space. This confirms our claim about the limitations of KTA, and FSM can solve this problem.

4.2. Benchmark data

We mimicked the model selection process by using surrogate measures to choose kernel types. We showed the advantage of our proposed measure over KTA and CSM in real situations by selecting several popular data sets from the UCI collection for

[2] It is noteworthy that *CSMnorm* is a monotonically increasing function of CSM, hence ranking using either CSM or *CSMnorm* give the same result. We do not take the square of CSM as it contains square terms already.

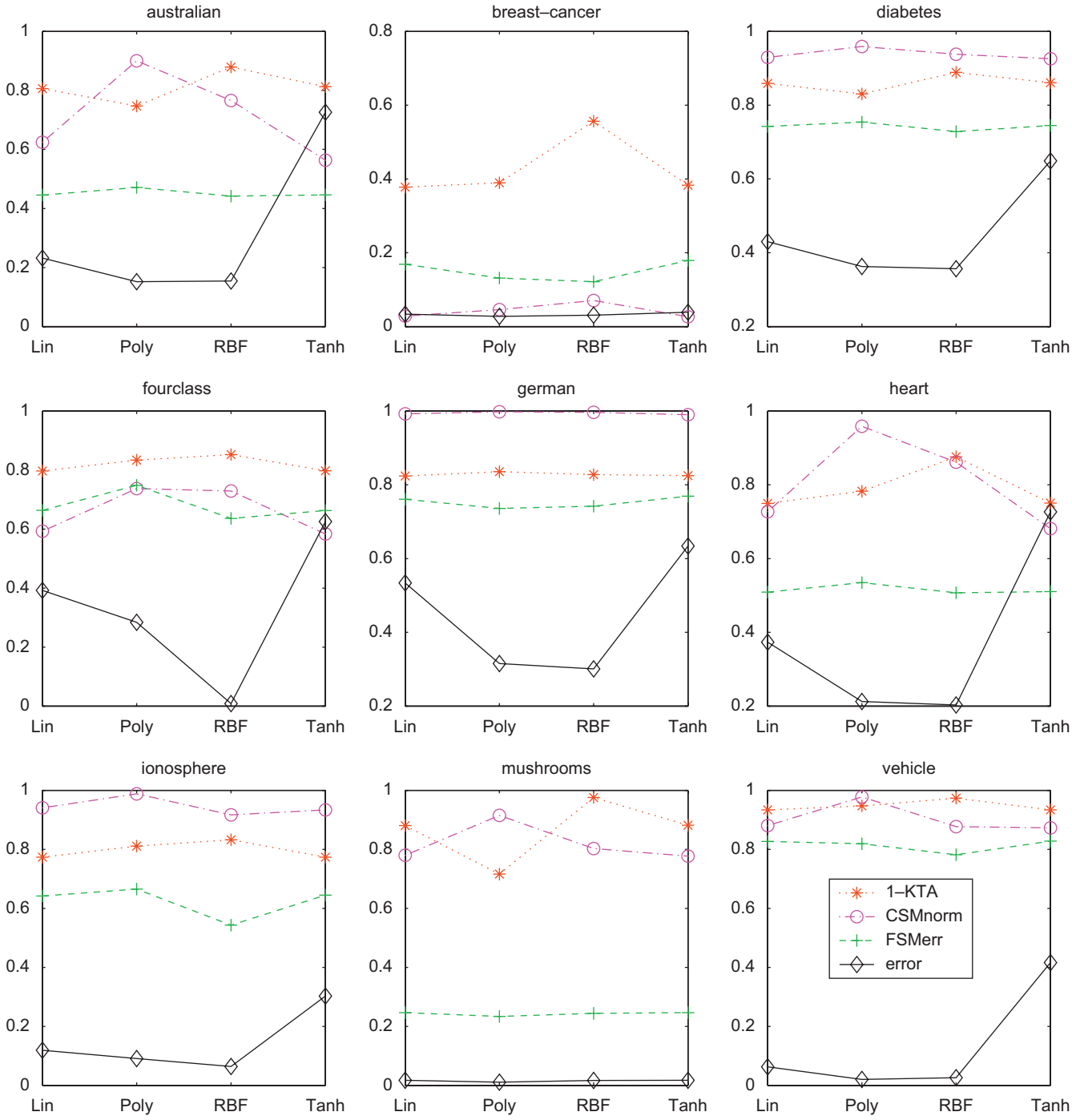

Fig. 4. Model comparison with different surrogate measures on UCI data sets. Taking cross validation as baseline, FSM gives a more reliable indicator of good kernels compared to KTA and CSM.

experiments Data names and experimental results are displayed in Fig. 4. Data were preprocessed as follows. It was first normalized to $[-1, 1]$. Classes were grouped to make binary classification problems. We chose four types of kernel for model selection: linear (denoted as Lin) kernels, polynomial (Poly) kernels (degree 3 with scale 1), Gaussian (RBF) kernels (default γ), and tan-hyperbolic (Tanh) kernels (default).[3] As described above, we ran cross validations and displayed the results of different kernels.

The following conclusions can be observed from the graphs:

- Both RBF and polynomial kernels give the (approximately) lowest cross validation error rates in nine out of nine data sets. However, $1 - KTA$ fails to rank any of either RBF or polynomial kernels in six data sets. Likewise, *CSMnorm* fails in eight data sets. *FSMerr* only fails in one data sets.
- Similarly, either Tanh kernels or linear kernels give the highest cross validation error rates in all cases. However $1 - KTA$ and *CSMnorm* never show them to be the worst options. For this, *FSMerr* agrees with cross validation in five data sets.

[3] Default parameter values are set by LIBSVM environment at http://www.csie.ntu.edu.tw/~cjlin/libsvm/

Table 1
Model comparison

Data	$1 - KTA$	*CSMnorm*	*FSMerr*
Australian	1	4	4
Breast-cancer	3	3	2
Diabetes	4	3	1
Fourclass	4	3	1
German	3	3	2
Heart	4	3	1
Ionosphere	4	1	1
Mushrooms	1	4	1
Vehicle	3	4	2
Average	3.00	3.11	1.67

Ranking the best model (in terms of cross validation error rates) using surrogate measures.

- In case of very low error rates as in *breast-cancer* and *mushrooms* data sets, FSMerr is highly correlated to error rates, but $1 - KTA$ and *CSMnorm* are not.

We ranked the kernels according to $1 - KTA$, *CSMnorm*, *FSMerr* and *error* separately from 1 to 4. We collected the rank of the best kernel (according to *error*) for each data set. On average, $1 - KTA$ ranks the best kernel at 3.00 ($\pm$1.22), *CSMnorm* ranks at 3.11 ($\pm$0.93), *FSMerr* ranks it at 1.67 ($\pm$1.00). A t-test (at 95% level of confidence) shows that *FSMerr* ranks the best kernels smaller than *KTA* and *CSMnorm*. This also confirms the advantage of *FSMerr* (Table 1).

In summary, we used synthetic and real data sets and compared measures against the de facto standard of cross validation error rates. We showed that using our measure correlates the best model is more closely to the error rates than using KTA and CSM. When ranking the best kernel according to the error rates, FSM shows a significantly lower rank, on average. This suggests that our measure is more reliable. This confirms our analysis of the limitations of KTA and CSM, and that our measure can overcome these limitations. It was also observed that there is still a gap between kernel matrix evaluation measures and error rates, as error rates are collected from many training rounds while measures are calculated with one pass through the kernel matrices. This is the trade off necessary for efficiency.

5. Conclusion

The paper shows that KTA, an efficient kernel matrix measure, which is popular in many contexts, has fundamental limitations, as it is only a sufficient, not a necessary condition to evaluate the goodness of a kernel matrix. A new measure is proposed to overcome those limitations by using data distributions in the feature space. This new measure follows the conventional wisdom of measuring within-class and between-class spreading (specifically, standard derivations) in an appropriate way. The measure provides the same efficiency as KTA, as evaluated in $O(n^2)$ time complexity, and other properties. It also has links with other methods. This measure reflects better the error rates of SVMs than the ones based on KTA and CSM do.

The implication of this work is vast. One can take into account finer data distribution models in the feature space, to improve the current work. It is an interesting direction to extend this to multimodal data distributions and to regression problems. Also, there are a large number of applications of this measure on other works. We can directly apply this measure (similarly to KTA) to many feature and model selection problems such as boosting kernel matrices, multiple kernel learning, feature selection and so on. In general, having an efficient kernel matrix evaluation, we can leverage the work of kernel matrix and kernel function learning, which is of central interest in kernel methods.

References

[1] B. Schölkopf, A.J. Smola, Learning with Kernels, MIT Press, Cambridge, MA, 2002.
[2] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, New York, NY, USA, 2004.
[3] N.V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, NY, 2000.
[4] B. Schölkopf, A.J. Smola, K.-R. Müller, Kernel principal component analysis, in: Advances in kernel methods: support vector learning, 1999, pp. 327–352.
[5] G. Wahba, Spline Models for Observational Data, CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 59, SIAM, Philadelphia, 1990.
[6] S. Fine, K. Scheinberg, Efficient SVM training using low-rank kernel representations, J. Mach. Learn. Res. 2 (2002) 243–264.
[7] C.S. Ong, A.J. Smola, R.C. Williamson, Learning the kernel with hyperkernels, J. Mach. Learn. Res. 6 (2005) 1043–1071.
[8] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, J. Kandola, On kernel-target alignment, in: T.G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, 2001.
[9] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L.E. Ghaoui, M.I. Jordan, Learning the kernel matrix with semidefinite programming, J. Mach. Learn. Res. 5 (2004) 27–72.
[10] J. Neumann, C. Schnorr, G. Steidl, Combined SVM-based feature selection and classification, Mach. Learn. 61 (1–3) (2005) 129–150.
[11] M. Wu, J. Farquhar, A subspace kernel for nonlinear feature extraction, in: M.M. Veloso (Ed.), International Conference on Artificial Intelligence, 2007, pp. 1125–1130.
[12] J.T. Kwok, I.W. Tsang, Learning with idealized kernels, in: International Conference on Machine Learning, 2003, pp. 400–407.
[13] K. Crammer, J. Keshet, Y. Singer, Kernel design using boosting, in: Advances in Neural Information Processing Systems, 2002, pp. 537–544.
[14] J. Kandola, N. Cristianini, J. Shawe-Taylor, Learning semantic similarity, Advances in Neural Information Processing Systems, vol. 15, 2003.
[15] J. Kandola, J. Shawe-Taylor, Refining kernels for regression and uneven classification problems, in: C. Bishop, B. Frey (Eds.), Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, 2003.
[16] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, New York, NY, USA, 2004.
[17] M. Meila, Data centering in feature space, in: C.M. Bishop, B.J. Frey (Eds.), Ninth International Workshop on Artificial Intelligence and Statistics, 2003.
[18] I.S. Gradshteyn, I.M. Ryzhik, Table of Integrals, Series, and Products, Academic Press, San Diego, CA, 2000.
[19] T. Gartner, J.W. Lloyd, P.A. Flach, Kernels and distances for structured data, Mach. Learn. J. 57 (3) (2004) 205–232.
[20] T. Jebara, R. Kondor, A. Howard, Probability product kernels, J. Mach. Learn. Res. 5 (2004) 819–844.
[21] W. Feller, An Introduction to Probability Theory and its Applications, vol. 2, Wiley, New York, 1971.
[22] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intell. Data Anal. 6 (5) (2002) 429–449.
[23] G.R.G. Lanckriet, L.E. Ghaoui, C. Bhattacharyya, M.I. Jordan, A robust minimax approach to classification, J. Mach. Learn. Res. 3 (2002) 555–582.
[24] L. Wang, K.L. Chan, Learning kernel parameters by using class separability measure, in: Advances in Neural Information Processing Systems, Sixth workshop on Kernel Machines, Canada, 2002.
[25] K. Fukunaga, Introduction to Statistical Pattern Recognition, second ed., Academic Press, New York, 1990.
[26] A. Globerson, S. Roweis, Metric learning by collapsing classes, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in Neural Information Processing Systems, vol. 18, MIT Press, Cambridge, MA, 2006, pp. 451–458.
[27] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, K.-R. Müller, Fisher discriminant analysis with kernels, in: Y.-H. Hu, J. Larsen, E. Wilson, S. Douglas (Eds.), Neural Networks for Signal Processing IX, IEEE, 1999, pp. 41–48.

About the Author—CANH HAO NGUYEN is a Ph.D. student at School of Knowledge Science, Japan Advanced Institute of Science and Technology, Japan. He received his Bachelor of Science in Computer Science (with Honors) from the University of New South Wales in 2002. His research interests lie in statistical machine learning, ranging from theoretical to application studies.

About the Author—TU BAO HO is a professor at School of Knowledge Science, Japan Advanced Institute of Science and Technology, Japan. He received his M.S. and Ph.D. from Marie and Pierre Curie University in 1984 and 1987, respectively. His research interests include knowledge-based systems, machine learning, data mining, medical informatics and bioinformatics.