Article

Not peer-reviewed version

KOS: Kernel-based Optimal Subspaces Method for Data Classification

Lakhdar Remaki *

Posted Date: 14 October 2025

doi: 10.20944/preprints202510.1014.v1

Keywords: Classification; Machine learning; SVM; kernel theory; POD

 doi:10.20944/preprints202510.1014.v1

Article

KOS: Kernel-Based Optimal Subspaces Method for Data Classification

Lakhdar Remaki

College of Science and General Studies, Departement of Mathematics and Computer Science, Alfaisal University, P.O. Box 5092, Riyadh 11533 Saudi Arabia; lremaki@alfaisal.edu

Abstract

Support Vector Machines (SVM) is a popular kernel-based method for data classification that have demonstrated high efficiency across a wide range of practical applications. However, SVM suffers from several limitations, including the potential failure of the optimization process,especially in high-dimensional spaces, the inherently high computational cost, the lack of a systematic approach to multiclass classification, difficulties in handling imbalanced classes, and the prohibitive cost of real-time or dynamic classification. This paper proposes an alternative method, referred to as Kernel-based Optimal Subspaces (KOS). The method achieves performance comparable to SVM while addressing the aforementioned weaknesses. It is based on computing a minimum distance to optimal feature subspaces of the mapped data. No optimization process is required, which makes the method robust, fast, and easy to implement. The optimal subspaces are constructed independently, enabling high parallelizability and making the approach well-suited for dynamic classification and real-time applications. Furthermore, the issue of imbalanced classes is naturally handled by subdividing large classes into smaller sub-classes, thereby creating appropriately sized sub-subspaces within the feature space.

Keywords: classification; machine learning; SVM; kernel theory; POD

1. Introduction

Machine Learning (ML), a branch of Artificial Intelligence (AI), is the science of designing algorithms that can automatically improve by learning from data (see, for instance, [1]). This multidisciplinary field has a wide range of applications, including handwriting recognition, automated disease detection, robotics, and others.

Among the core problems in AI is data classification, which relies heavily on ML techniques. Kernel methods are one of the most popular subsets of data classifiers, with the Support Vector Machine (SVM) being among the most widely used. The SVM algorithm is grounded in statistical learning theory, originating from Vapnik's work on the Structural Risk Minimization principle [2–4].

The fundamental idea of kernel methods is to map the original data,referred to as attributes, into a higher-dimensional space, known as the feature space, where the data can be more easily separated. While SVM was originally designed for linear separation, this mapping enables it to handle data that is non-linearly separable in the original space by achieving linear separation in the feature space. Note that, the mapping function is not explicitly defined, which is a tedious task, instead a suitable kernel function is used, as long as all operations can be expressed using dot products. The existence of such kernels is guaranteed under certain conditions by Mercer's theorem [4,5].

SVM is widely recognized for its high performance (see [6–13] for recent improvements). However, despite significant advancements, it still has several limitations:

- As an optimization problem, SVM may fail, particularly in high-dimension, even though the cost function is convex (quadratic).

 doi:10.20944/preprints202510.1014.v1

- Extending SVM to multiclass classification is not straightforward and becomes computationally expensive in high dimensions or when the number of classes is large. For instance, a common approach combining one-against-one and one-against-all strategies requires computing a total of $n + \frac{n(n-1)}{2}$ separating hyperplanes (i.e., solving n optimization problems), where n is the number of classes.
- Imbalanced classes poses another challenge. A popular approach to addressing this involves adding synthetic attributes to underrepresented classes. While effective, the process is not straightforward, and can alter the structure of the original class data.
- In dynamic classification, where classes may be added or removed, all separating hyperplanes must be recalculated, which can be prohibitively expensive for real-time applications.

To address these issues, a preliminary version of a novel method,proposed as an alternative to Support Vector Machines (SVMs), was introduced in a recent conference paper [30], along with basic initial results. In this extended journal version, we provide the complete formulation of the method for centered data, which offers more efficiency, along with a thorough theoretical analysis and corresponding proofs. Furthermore, we propose an effective strategy to tackle the inherent imbalanced classes problem. Extensive experiments are conducted on large, multi-class datasets for a thorough evaluation evaluation of the method's performance. In addition, a detailed analysis of time efficiency is included. Finally, when using the radial basis function (RBF) kernel, we introduce a robust learning procedure for selecting the parameter σ.

In this work, the new method is referred to as Kernel-based Optimal Subspaces (KOS). The approach relies on computing a minimum distance between an unseen feature vector and a set of optimal subspaces, in a sense that will be defined, in the feature space. It will be shown that subspaces constructed using Proper Orthogonal Decomposition (POD) are optimal.

For classification, the method projects an unseen feature vector onto each feature subspace, claculate the distnace, and then assigns the attribute vector to the class achieving a minimum distance. Fundamentally, this requires only the computation of the eigenpairs of the Mercer kernel matrix for each training class independently, and no optimization is involved, making the method simple to implement, significantly faster than SVM (with linear complexity), and robust.

KOS is highly parallelizable and well-suited for dynamic and real-time classification, as only the eigenpairs for new classes need to be computed. If a class is removed, no further computation is required.

The issue of imbalanced classes is handled naturally by splitting large attribute classes into balanced subclasses, then labeling all corresponding feature subspaces as belonging to a single class. An unseen feature vector is classified into a given class if it is close to any of the subspaces generated by that class's subclasses.

The performance of the proposed method is demonstrated through both 2D and high-dimensional, multi-class test cases. The results are compared against those of the Support Vector Machine (SVM), using the popular LIBSVM software package [21]. The findings show that KOS performs comparably to SVM, while offering several significant advantages as discussed earlier, making it a strong alternative to SVM.

The experiments use the Radial Basis Function (RBF) kernel, which, as is well known in the context of SVM, involves tuning two key parameters: the standard deviation σ (in the Gaussian form) or γ (in the RBF form), and the penalty parameter C. The most widely used method for selecting these parameters is cross-validation-based grid search, as described in [14–18], including both automated and manual search approaches.

In contrast, the KOS method does not require the penalty parameter C. Only one parameter, σ or γ, needs to be learned. Through a class-wise re-scaling process, this parameter is learned in a robust and effective manner, contributing to the overall performance of the method.

In Section 2, a short overview of the proper orthogonal decomposition (POD) is provided. In Section 3, details of the proposed KOS method are developed. Tests to validate the proposed method

 doi:10.20944/preprints202510.1014.v1

by comparison to SVM results are performed and reported in Section 4. Conclusions are drawn in Section 5.

2. KOS: Kernel-Based Optimal Subspaces Method

Before detailing the proposed method, we first present a brief overview of the proper orthogonal decomposition technique, which is a crucial component of the new approach.

2.1. Proper Orthogonal Decomposition (POD): Overview

Proper Orthogonal Decomposition (POD) [19], known as well as Karhunen-Loeve's expansion (KLE) [19], popular in data analysis field, is a technique that aims to represent a big amount of data by a reduced number of basis elements built from the data. More precisely, if the data are stored in a $m \times N$ matrix A with N being a large number, is it possible to represent the N columns of A by a p orthonormal vectors with p very small compared to N. The mathematical formulation can be summarized as follows, let $A = [Y_1, Y_2, ..., Y_N]$, the problem is to find $B = \left[V^1, V^2, ..., V^p\right]$ an orthonormal set of p vectors that better represent the N columns A in the following sense,

V^1 satisfying the optimization problem,

$$V^1 = Arg\left[Max_{V\in IR^m}\left(\sum_{i=1}^{N}(\langle V, Y_i\rangle)^2\right) \quad subject\ to \quad \|V\|^2 = 1\right]. \tag{1}$$

$$V^2 = Arg\left[Max_{V\in IR^m}\left(\sum_{i=1}^{N}|\langle V, Y_i\rangle|^2\right) \quad subject\ to \quad \|V\|^2 = 1\ and\ \left\langle V, V^1\right\rangle = 0\right]. \tag{2}$$

$$\vdots$$

$$V^k = Arg\left[Max_{v\in IR^m}\left(\sum_{i=1}^{N}|\langle V, Y_i\rangle|^2\right) \quad subject\ to \quad \|V\|^2 = 1 \quad and\ \left\langle V, V^1\right\rangle = 0, \left\langle V, V^2\right\rangle = 0, ..., \left\langle V, V^{k-1}\right\rangle = 0\right]. \tag{3}$$

The vectors $V^k, k = 1, p$ are the eigenvectors of AA^T. A simplified proof is provided in Appendix (A) for self-content. For more details, see [19].

2.1.1. POD Basis System

The POD basis system is built as follows:

1. : if $n \leq m$, let M be the correlation matrix $M_{i,j} = (A^T A)_{i,j}$ where $(,)$ is any inner product. Let V the matrix of eigenvectors of M, then the basis functions Ψ_i, called *modes*, are given by:

$$\Psi_i = \sum_{j=1}^{N} v_j^i Y_j \tag{4}$$

where v_j^i are the components of the i^{th} eigenvector of M

2. : if $n > m$, the POD basis is defined by the eigenvectors of the matrix $\overline{M}_{i,j} = (AA^T)_{i,j}$

 doi:10.20944/preprints202510.1014.v1

This paper will use the first definition since the original data will be mapped into an infinite dimension vector space.
Now calculate the inner product (Ψ_i, Ψ_j)

$$(\Psi_i, \Psi_j) = (\sum_{l=1}^{N} v_l^i Y_l, \sum_{k=1}^{N} v_k^j, Y_k) = \sum_{l,k} v_l^i v_k^j (Y_l, Y_k) = \sum_{l,k} v_l^i v_k^j M_{l,k} = MV^i.V^j = \lambda_i V^i.V^j \tag{5}$$

This shows that the functions $\Psi_i, 1 \leq i \leq N$ are orthogonal if and only the eigenvectors $V^i, 1 \leq i \leq N$ are, which is generally the case. For $i = j$ we get the norm $\|\Psi_i\| = \sqrt{\lambda_i}$, which shows that the norm of each mode represents part of the energy of the system. In conclusion, modes with low energy (small eigenvalues) can be neglected, reducing the POD basis to only modes with significant energy. Practically, the ratio of the modeled to the total energy contained in the system given by:

$$\varepsilon(l) = \frac{\sum_{i=1}^{l} \lambda_i}{\sum_{i=1}^{N} \lambda_i} \tag{6}$$

is calculated to decide if l modes are enough to describe the system.

This is an example of a quote.

2.2. KOS: Global Description and Some Theoretical Aspects

Similar to SVM, the first step in KOS is to map the original data (attributes) into a higher-dimensional feature space using a kernel function. For each class in the feature space, an optimal subspace, defined as the smallest and best fit subspace that contains the feature vectors of that class, is then constructed. The main difference

To classify an unseen attribute vector, its corresponding feature vector is projected onto each class subspace, and the class is determined based on the minimum distance to these subspaces. The fundamental difference from SVM is that the hyperplanes, responsible for many of the issues encountered with SVM, are removed and replaced by optimal subspaces that contain the feature data, as illustrated in Figure 1.

Note that, in some cases, different classes may be mapped to the same feature subspace, leading to a misclassification. Therefore, it is desirable for the subspaces to be as distinct as possible to reduce this risk. In theory, as proved later in the paper, it is always possible to construct subspaces that are even completely disjoint. However, in practice, this is cannot be guaranteed or controlled. For optimal performance, the kernel should be chosen such that the resulting feature space is of infinite dimension, which increases the likelihood of separability between class subspaces. It is well established, for example, that the RBF kernel meets this requirement.

Theorem 1. *There exist Mercer kernels such that the attribute classes can be mapped into disjoint feature subspaces.*

Proof. For simplicity, the proof is given for the two-class case; the result can then be extended recursively to the multiclass case. Let χ denote the attribute (original) space and let H be the feature space obtained through a mapping Φ associated with a Mercer kernel $K(x, y)$. Suppose that $K(x, y)$ is such that H is of infinite-dimensional.

$$\Phi : \chi \longrightarrow H \tag{7}$$

 doi:10.20944/preprints202510.1014.v1

Let C_1 and C_2 be two classes, and let E and F be two finite-dimensional subspaces of the feature space H such that $\Phi(C_1) \subset E$ and $\Phi(C_2) \subset F$. Let E^T denote the orthogonal complement of E in H. Since H is infinite-dimensional and E is finite-dimensional, E is also infinite-dimensional.

Let $G \subset E^T$ be any subspace with the same dimension as F. Then there exist a one-to-one mapping $\phi : F \to G$.

Now,define a new mapping $\psi : \chi \longrightarrow H$ as follows:

$$\psi(x) = \begin{cases} \phi(\Phi(x)) \;\; if \;\; x \in C_2 \\ \Phi(x) \;\; if \;\; not \end{cases} \tag{8}$$

We have $\Phi(C_1) \subset E$ and $\Phi(C_2) \subset G$ with E and G orthogonal then disjoint. The kernel defined by $\overline{K}(x,y) = \psi(x) * \psi(y)$ is a *Mercer* kernel by construction and satisfies the theorem, which ends the proof. □

2.3. POD Feature Subspaces

The POD technique is proposed to build the optimal subspaces in the feature vector space. Note that similar techniques, such as PCA (Principal Component Analysis), are commonly used in kernel theory for data classification, typically as a pre-processing step for feature extraction and enhancement [22]. The optimality of the POD feature subspaces will be discussed later.

Assume a suitable kernel $K(x,y)$ is given, and let ϕ be the associated mapping function. It is important to note that choosing a kernel such that the feature space H is of infinite dimension is highly recommended, as it helps map data into subspaces with minimal or no overlap.. Let $C_{l,l=1,P}$ denote the P attributes classes. Denote by X_i^l the i^{th} element of class C_l and by $Y_i^l = \phi(X_i^l)$ its corresponding mapped feature vector. To construct the POD feature subspaces, we only need to build the POD basis, also known as the modes, as described in Section 2.1.

In general, it is well established that centering the data is recommended when applying the POD method. Therefore, we provide formulations of the POD basis for both non-centered centered and centered data.

To lighten the notations, the superscript l , which refers to a specific class, is omitted and will be reintroduced as needed. In the following, we use $K_{i,j}$ to denote the *Mercer*' kernel K matrix, namely $K_{i,j} = Y_i Y_j = \phi(X_i)\phi(X_j)$, and $K(X,Y)$ when X or Y does not belong to the training dataset.

1. **None-Centered data formulation**: In this case, the data in the feature space are used without centering. Let $Y_i = \phi(X_i), i = 1, N$ be the mapped data points of a selected class of size N. First, we construct the correlation matrix:

$$M(i,j) = Y_i Y_j = \phi(X_i)\phi(X_j) = K(i,j), \tag{9}$$

where $K(i,j)$ denotes the kernel function evaluated at Y_i and Y_j. As seen, the POD correlation matrix is equivalent to the kernel matrix derived from the mapping.
The normalized POD modes are then given by:

$$\psi_i = \frac{1}{\sqrt{\sigma_i}} \sum_{k=1}^{N} V_k^i Y_k, \tag{10}$$

where V^i is the i^{th} eigenvector of the kernel matrix M and σ_i the associated eigenvalue.

2. **Centered data formulation**: In this case, the feature vectors Y_i are centered. Let $\overline{Y} = \frac{1}{N}\sum_{k=1}^{N} Y_k$ be the mean vector, and $Z_i = Y_i - \overline{Y}$ be the centered data. The correlation matrix is then given by:

$$M(i,j) = Z_i Z_j = (Y_i - \overline{Y})(Y_j - \overline{Y}) = Y_i Y_j - Y_i\overline{Y} - Y_j\overline{Y} + \overline{Y}\overline{Y}. \tag{11}$$

 doi:10.20944/preprints202510.1014.v1

Each of the inner products can be expressed in terms of the kernel function:

$$Y_i\overline{Y} = Y_i\frac{1}{N}\sum_{k=1}^{N} Y_k = \frac{1}{N}\sum_{k=1}^{N} Y_iY_k = \frac{1}{N}\sum_{k=1}^{N} K(i,k), \tag{12}$$

$$\overline{YY} = \frac{1}{N}\sum_{k=1}^{N} Y_k\frac{1}{N}\sum_{l=1}^{N} Y_l = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N} K(l,k). \tag{13}$$

Substituting back into the expression for $M(i,j)$, the centered correlation matrix becomes:

$$M(i,j) = K(i,j) - \frac{1}{N}\sum_{k=1}^{N}(K(i,k)+K(j,k)) + \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N} K(l,k). \tag{14}$$

As in the non-centered case, the correlation matrix can be entirely expressed using the kernel matrix.

The normalized POD modes for centered data are then given by:

$$\psi_i = \frac{1}{\sqrt{\sigma}_i}\sum_{k=1}^{N} V_k^i(Y_k - \overline{Y}) \tag{15}$$

where V^i is the i^{th} eigenvector of the correlation matrix M defined by 14 and σ_i the associated eigenvalue

Note that for calculating the minimum distance in classification (as will be shown later), it is not necessary to explicitly compute the feature vectors Y_k or the POD modes.

2.4. Optimality of the POD Feature Subspaces

The optimality of the feature subspaces is considered from both geometric and algebraic perspectives.

From a geometric standpoint, the basis elements of the subspace should be as close as possible to the original feature vectors. This ensures maximum representativeness and preserves the structure of the data within the feature space.

From an algebraic standpoint, the feature subspace should be of minimal dimension while still containing the feature vectors. This dimensional compactness guarantees maximum discrimination between different feature subspaces and, consequently, between the associated attribute classes.

In essence, the POD method provides a set of orthonormal basis functions that span the most "informative" subspace, allowing for an efficient and discriminative representation of each class in the feature space.

- Geometrical Optimality of POD Feature Spaces:
  As shown in [19], if Ψ is the POD basis matrix constructed to represent the columns of a matrix $A = [Y_1, Y_2, ..., Y_N]$, and $\overline{\Psi}$ is any other basis of a same size constructed for the same purpose, then the following inequality holds:
  $$\|A - \Psi\|_F \leq \|A - \overline{\Psi}\|_F, \tag{16}$$
  where $\|.\|_F$ is the *Frobenius* norm, defined by:
  $$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{N}|A_{i,j}|} = \sqrt{Trace(A^TA)} \tag{17}$$
  This result demonstrates the geometric optimality of the POD basis: among all possible orthonormal bases of the same dimension, the POD basis provides the best approximation (in the least-squares sense) of the original data matrix A. Therefore, the POD basis vectors are the closest

 doi:10.20944/preprints202510.1014.v1

possible representatives of the original feature vectors, ensuring maximum fidelity in capturing the data structure.

- Algebraic Optimality of POD Feature Spaces:
  Lemma: The POD subspace is algebraically optimal; it is the smallest subspace containing the mapped data.
  Proof: The proof is straightforward. To show that the POD subspace is the smallest subspace containing the mapped (feature) data, it is sufficient to show that any subspace containing the feature data must also contain the POD subspace.
  Let F be a subspace that contains the given feature data. To prove that F also contains the POD subspace, it suffices to show that F ontains the POD modes (i.e., the basis of the POD subspace). Since the POD modes are linear combinations of the feature data, and F s a subspace (i.e., closed under linear combinations), the modes necessarily lie in F. Hence, the POD subspace is contained in a ny subspace that contains the mapped data, proving its algebraic optimality.

2.5. Decision Criterion

The decision criterion for an unseen element $\widehat{X}$, as introduced at the beginning of this section, is based on the minimum distance of the feature vector $\widehat{Y} = \phi(\widehat{X})$ to the POD feature subspaces.To achieve this, the POD coordinates of $\widehat{Y}$ within each subspace are computed. The details for both non-centered and centered data are provided.

1. **None-Centered data formulation**: The coordinates $\alpha_i, i = 1, N$ are given by the projection of the element on the POD modes

$$\alpha_i =< \psi_i, \widehat{Y} > \tag{18}$$

with

$$\psi_i = \frac{1}{\sqrt{\sigma}_i} \sum_{k=1}^{N} V_k^i Y_k, \tag{19}$$

Then

$$\alpha_i = \frac{1}{\sqrt{\sigma}_i} < \sum_{k=1}^{N} V_k^i Y_k, \widehat{Y} >= \frac{1}{\sqrt{\sigma}_i} \sum_{k=1}^{N} V_k^i (Y_k \widehat{Y}) = \frac{1}{\sqrt{\sigma}_i} \sum_{k=1}^{N} V_k^i K(X_k, \widehat{X}) \tag{20}$$

Using the Pythagoras rule, the distance of $\overline{Y}$ to the POD subspace F, is given by

$$dis(\widehat{Y}, F) = \sqrt{\|\widehat{Y}\|^2 - \sum_{i=1}^{N} \alpha_i^2}. \tag{21}$$

With

$$\|\widehat{Y}\|^2 = \widehat{Y}\widehat{Y} = \phi(\widehat{X})\phi(\widehat{X}) = K(\widehat{X}, \widehat{X}). \tag{22}$$

2. **Centered data formulation**: For the centered case, $\widehat{Y}$ and Y_i are replaced by $\widehat{Y} - \overline{Y}$ and $Y_i - \overline{Y}$ respectively. That is,

$$\alpha_i =< \psi_i, (\widehat{Y} - \overline{Y}) >, \tag{23}$$

with

$$\psi_i = \frac{1}{\sqrt{\sigma}_i} \sum_{k=1}^{N} V_k^i (Y_k - \overline{Y}), \tag{24}$$

 doi:10.20944/preprints202510.1014.v1

Then

$$\alpha_i = \frac{1}{\sqrt{\sigma_i}} < \sum_{k=1}^{N} V_k^i (Y_k - \overline{Y}), (\widehat{Y} - \overline{Y}) >= \frac{1}{\sqrt{\sigma_i}} \Big[\sum_{k=1}^{N} V_k^i (Y_k \widehat{Y}) - \sum_{k=1}^{N} V_k^i (Y_k \overline{Y}) - \sum_{k=1}^{N} V_k^i \overline{Y} \widehat{Y} + \sum_{k=1}^{N} V_k^i \overline{Y} \overline{Y} \Big]. \tag{25}$$

Substituting $\overline{Y} = \frac{1}{N} \sum_{j=1}^{N} Y_j$ we obtain

$$\alpha_i = \frac{1}{\sqrt{\sigma_i}} \Big[\sum_{k=1}^{N} V_k^i K(X_k, \widehat{X}) - \frac{1}{N} \sum_{k=1}^{N} \sum_{j=1}^{N} V_k^i K_{k,j} - (\frac{1}{N} \sum_{j=1}^{N} K(X_j, \widehat{X})) \sum_{k=1}^{N} V_k^i + (\frac{1}{N^2} \sum_{k=1}^{N} \sum_{j=1}^{N} K_{k,j}) \sum_{k=1}^{N} V_k^i \Big]. \tag{26}$$

Then

$$dis(\widehat{Y}, F) = \sqrt{\|\widehat{Y} - \overline{Y}\|^2 - \sum_{i=1}^{N} \alpha_i^2}. \tag{27}$$

With

$$\|\widehat{Y} - \overline{Y}\|^2 = \widehat{Y}^2 - 2\widehat{Y}\overline{Y} + \overline{Y}^2 = K(\widehat{X}, \widehat{X}) - \frac{2}{N} \sum_{j=1}^{N} K(\widehat{X}, X_j) + \frac{1}{N^2} \sum_{k=1}^{N} \sum_{j=1}^{N} K_{k,j}. \tag{28}$$

The unseen attribute vector is assigned to the class that corresponds to the minimum distance calculated above.

2.6. Imbalanced Classes

The issue of imbalanced classes is commonly encountered in many classifiers, including SVM. Several techniques address this problem, with the two most popular being weight adjustment between large and small classes (as used in LIBSVM) and the SMOTE (Synthetic Minority Over-Sampling Technique) algorithm [20], which generates synthetic data to augment smaller classes. Both methods yield acceptable results; however, there is no systematic approach for defining the weight distribution or the positioning of synthetic vectors, which may alter the data structure.

The proposed KOS method naturally handles this problem by subdividing large attribute classes into smaller subclasses of similar sizes. These subclasses will share the same class label. Specifically, a reference size is defined, which can be the size of the smallest class in the attribute space. Any class with a size at least double the reference size can be split into subsets, each roughly equal in size to the reference class. For the classification decision, if the minimum distance is obtained from one of the POD subspaces of a subdivided class, the unseen element is assigned to that class. This strategy is schematized in Figure 2 and will be validated in the test section.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 October 2025 doi:10.20944/preprints202510.1014.v1

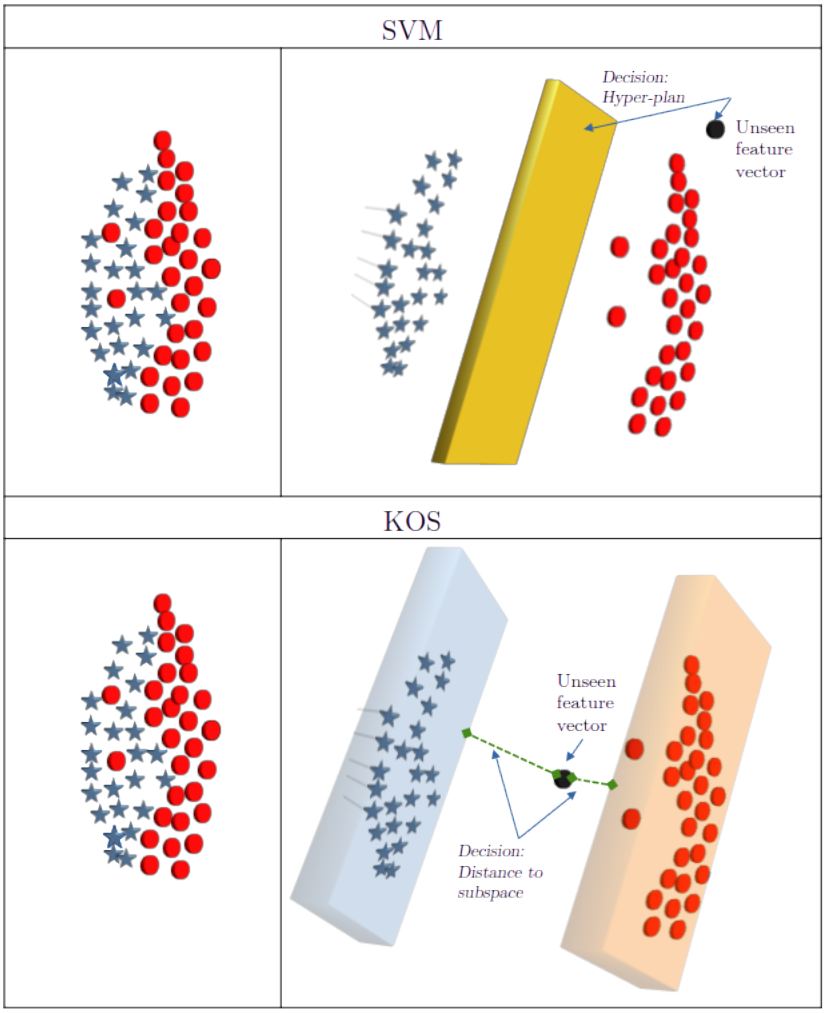

Figure 1. SVM vs KOS explanatory illustration.

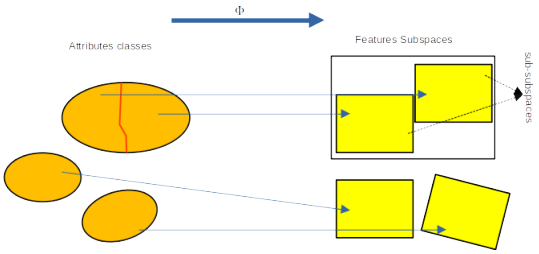

Figure 2. Imbalanced classes strategy.

2.7. RBF Parameter Learning

As previously mentioned, when using the RBF kernel, only the parameter σ (equivalently γ) must be learned, as in the case of hard-margin SVM. There is need to the penality parameter. We propose a strategy to learn σ that significantly reduces the range of values to be tested. The key idea is to scale σ according to the size of each attribute class.

The RBF kernel (also known as the Gaussian kernel) is widely used in multi-scale image analysis, where small σ values correspond to fine image details, and larger values capture more global structures. This makes it natural to scale σ according to the size of the input set. We decompose σ as follows:

$$\sigma = \omega D, \tag{29}$$

 doi:10.20944/preprints202510.1014.v1

where D is the diameter of the set, computed as the maximum distance between any two elements in the set. The scaling factor ω is then varied from 0.1 to 1 in steps of 0.1. This approach significantly reduces the complexity of the parameter search. For comparison, in standard SVM, approximately 250×250 grid searches are typically required to determine the optimal pair (σ, C), where C is the penalty parameter.

2.8. Summary of the Algorithm

Let $C_{k,\ k=1,P}$, denote P attributes sets representing P different classes used during the learning stage. Let $B_{k=1,P}$, denote the corresponding mapped sets in the feature space. The KOS classification procedure is as follows:

1. Check whether the classes C_k are imbalanced. If so, split the larger classes into smaller subclasses according to procedure described in Section 2.6.
2. Select a *Mercer* kernel K, then compute the kernel matrices K_k for each feature class C_k (of feature subclasses if applicable).
3. Compute the POD correlation matrices M_k for each feature class (or subclass) C_k using either the non-centered formulation 9 or the centred formulation 14.
4. Compute the eigenvalues and eigenvectors of each matrix $M_{k,\ k=1,P}$.
5. For each unseen vector $\widehat{X}$, compute its coordinates $\alpha_i^k, i = 1, size(C_k)$ in each POD feature subspace using Formula (20) for non-centered case or 26 for centered case.
6. Compute the distance of $\widehat{X}$ to each POD feature subspace using Formula (21) for non-centered case or 27 for centered case.
7. **Decision**: Assign $\widehat{X}$ to the class corresponding to the minimum computed distance.

3. Results: Validation and Discussion

The proposed KOS method is verified and validated using the RBF kernel. Initial tests are conducted on 2D cases to visually assess its ability to handle non-linear classification. Subsequently, KOS is tested on selected real-world datasets from the LIBSVM repository [21], and the results are compared against those obtained using SVM implemented in the widely used LIBSVM software.

It is important to note that no specific preprocessing is applied to the data for KOS, apart from rescaling the input features to the interval $[-1, 1]$.

3.1. 2D Test Cases

To evaluate KOS's capability in separating non-linearly separable data, three increasingly challenging scenarios are tested: (i) connected sets, (ii) non-connected sets, and (iii) a spiral configuration. Figures 3–5 demonstrate that KOS successfully separates the data in all cases. A fixed kernel parameter value of $\sigma = 1.2$ is used for all three tests.

3.2. Higher-Dimension Test Cases

Two-classes and multi-class test cases are selected from the database [21]. The proposed KOS and SVM (using LIBSVM software) are then tested, and their performances compared. Note that for KOS, there is no penalty parameter C; only the standard deviation σ is learned. This is done by splitting the training set into two parts: $2/3^{rd}$ for building the KOS model (eigenpairs of the correlation matrices) and $1/3^{rd}$ for validation. The value of σ is varied from $\sigma = 0.1$, to $\sigma = 1.4$ in increments of 0.1. Once the optimal σ is found, the entire training set is used to build the KOS model. The tests are briefly described in Table 1, and Table 2. Table 3 compares the performance of KOS and SVM. Overall, the table shows that KOS performs comparably to SVM, while offering significant advantages outlined in the next subsection.

The effectiveness of the proposed imbalanced strategy is demonstrated, for instance, in the Leukemia and Shuffle tests, which contain the most imbalanced classes. Specifically, the performance

 doi:10.20944/preprints202510.1014.v1

without splitting the classes was 67.64% and 80.6%, respectively. After applying the splitting strategy, these scores improved to 82.35% and 99.9%, respectively.

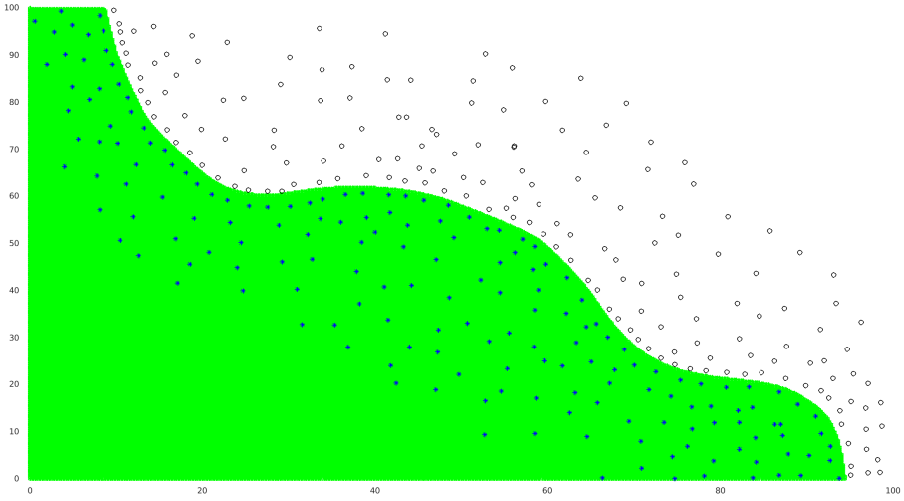

Figure 3. Connected sets case.

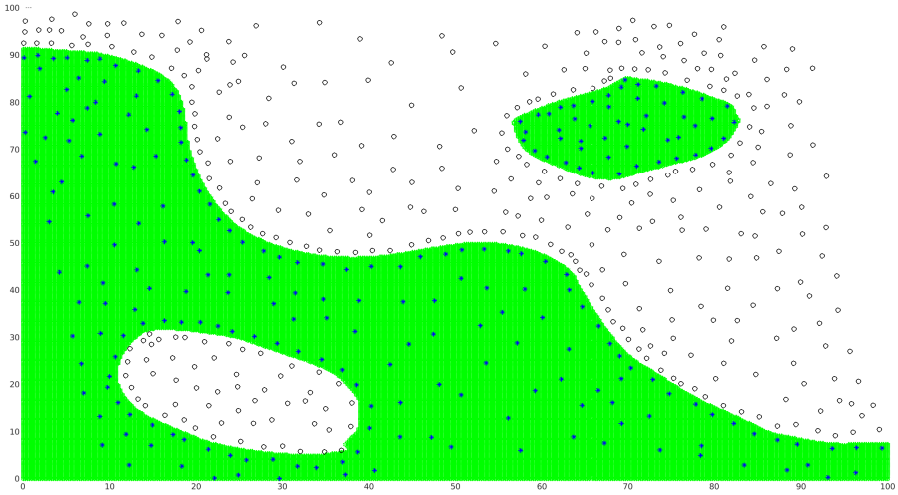

Figure 4. Non-connected sets case.

Figure 5. Spiral case.

 doi:10.20944/preprints202510.1014.v1

3.3. KOS main Advantages Compared to SVM

As shown by the tests, KOS is performing similarly to SVM, with many significant advantages that can be listed as:

1. The KOS algorithm is easier to implement: it only requires the computation of the correlation matrix eigenpairs (9 or 14), followed by a direct application of formulas (20, 21 or 26, 27).
2. Robustness: KOS algorithm is robust and faster; No optimization process is required that can slow down the algorithm in high dimensions and can fail even if the cost function is quadratic because of rounding-off errors as in the case of SVM.
3. Complexity: SVM complexity grows quadratically with the number of classes (requiring $n + \frac{n(n-1)}{2}$ hyperplanes), while KOS scales linearly, needing only n sets of eigenpairs.
4. Time efficiency: As a result of the two previous proprieties, KOS is significantly faster than SVM, up to more than 60 times faster in certain tests as shown in Table 4.
5. Parallelization: KOS is highly parallelizable since all subspaces are independent..
6. Dynamic classification: In the case of class creation or cancellation, SVM requires recalculating all feature-space hyperplanes. By contrast, KOS only requires computing eigenpairs of the Mercer kernel matrix for the new class, with no update needed for canceled classes.
7. Imbalanced classes: In KOS, this issue is naturally handled by subdividing large classes into smaller, balanced subclasses (2.6). This avoids the need for artificial attributes, which may distort the data structure, or the use of balancing weights, which often lack a clear and systematic procedure.

Table 1. Tests Information (two classes).

Test name	NB of Classes	Description/Reference	Training set size/ Class sizes	Testing set size/ Feature vector size
Leukemia	2	Molecular classification of cancer [23]	38/ $Size(C_1) = 27$ $size(C_2) = 11$	34/ 7129
svmguide1	2	Astroparticle application [24]	3089/ $Size(C_1) = 200$ $size(C_2) = 200$	400/ 4
splice	2	Splice junctions in a DNA sequence [25]	1000/ $Size(C_1) = 483$ $size(C_2) = 517$	2175/ 60
austrian	2	Credit Approval dataset [26]	349/ $Size(C_1) = 191$ $size(C_2) = 158$	341/ 14
madelon 2	Analysis of the NIPS 2003 [27]	200/ $Size(C_1) = 1000$ $size(C_2) = 1000$	600/ 14	

 doi:10.20944/preprints202510.1014.v1

Table 2. Tests Information (multiple classes).

Test name	NB of Classes	Description/Reference	Training set size/ Class sizes	Testing set size/ Feature vector size
DNA	3	DNA [28]	2000/ $Size(C_1) = 464$ $size(C_2) = 485$ $size(C_3) = 1051$	1186/ 180
Satimage	6	Satelit images [28]	4435 $Size(C_1) = 1072$ $size(C_2) = 479$ $ize(C_3) = 961$ $Size(C_4) = 415$ $Size(C_5) = 470$ $Size(C_6) = 1038$	2000 36
USPS	10	handwritten text recognition dataset [29]	7291/ $Size(C_1) = 1194$ $size(C_2) = 1005$ $ize(C_3) = 731$ $Size(C_4) = 658$ $Size(C_5) = 652$ $Size(C_6) = 556$ $Size(C_7) = 664$ $Size(C_8) = 645$ $Size(C_9) = 542$ $Size(C_{10}) = 644$	2007/ 256
letter	26	letter recognition dataset [29]	15000/ $Size(C_1) = 789$ $size(C_2) = 766$ $ize(C_3) = 736$ $Size(C_4) = 805$ $Size(C_5) = 768$ $Size(C_6) = 775$ $Size(C_7) = 773$ $Size(C_8) = 734$ $Size(C_9) = 755$ $Size(C_{10}) = 747$ $Size(C_{11}) = 739$ $size(C_{12}) = 761$ $ize(C_{13}) = 792$ $Size(C_{14}) = 783$ $Size(C_{15}) = 753$ $Size(C_{16}) = 803$ $Size(C_{17}) = 783$ $Size(C_{18}) = 758$ $Size(C_{19}) = 748$ $Size(C_{20}) = 796$ $Size(C_{21}) = 813$ $Size(C_{22}) = 764$ $Size(C_{23}) = 752$ $Size(C_{24}) = 787$ $Size(C_{25}) = 786$ $Size(C_{26}) = 734$	500/ 16
shuttle	7	space shuttle' sensors [29]	43500/ $Size(C_1) = 34108$ $size(C_2) = 37$ $ize(C_3) = 132$ $Size(C_4) = 6748$ $Size(C_5) = 2458$ $Size(C_6) = 6$ $Size(C_7) = 11$	14500/ 9

 doi:10.20944/preprints202510.1014.v1

Table 3. KOS-SVM performance comparison.

name/ Classes	Test size	SVM-rate of success	KOS-rate of success
Leukemia/2	*Train*.#38 *Test*.#34	82.35%	82.35%
svmguide1/2	*Train*.#3089 *Test*.#4000	96.87%	96.3%
splice/2	*Train*.#1000 *Test*.#2175	90.43%	89.42%
austrian/2	*Train*.#349 *Test*.#341	85.04%	85.63%
madelon/2	*Train*.#2000 *Test*.#600	61.16%	61.5%
DNA/3	*Train*.#2000 *Test*.#1186	94.43%	92.32%
Satimage/6	*Train*.#4435 *Test*.#200	91.85%	91.35%
USPS/10	*Train*.#7291 *Test*.#2007	95.26%	95.76%
letter/26	*Train*.#15000 *Test*.#5000	97.9%	97.4%
shuttle/7	*Train*.#43, 500 *Test*.#14, 500	99.9%	99.9%

Table 4. KOS-SVM processing time comparison.

name/ Classes	Test size	SVM processing time	KOS processing time
Leukemia/2	*Train*.#38 *Test*.#34	0*m*8.646*s*	0*m*0.005*s*
svmguide1/2	*Train*.#3089 *Test*.#4000	0*m*33.327*s*	0*m*1.680854*s*
splice/2	*Train*.#1000 *Test*.#2175	0*m*29.190*s*	0*m*0.262854*s*
austrian/2	*Train*.#349 *Test*.#341	0*m*1.721*s*	0*m*0.025804*s*
madelon/2	*Train*.#2000 *Test*.#600	16*m*50.721*s*	0*m*1.571*s*
DNA/3	*Train*.#2000 *Test*.#1186	4*m*24.300*s*	0*m*0.614*s*
Satimage/6	*Train*.#4435 *Test*.#200	3*m*53.042*s*	0*m*1.468212*s*
USPS/10	*Train*.#7291 *Test*.#2007	81*m*18.931*s*	0*m*6.347*s*
letter/26	*Train*.#15000 *Test*.#5000	44*m*11.996*s*	0*m*21.330*s*
shuttle/7	*Train*.#43, 500 *Test*.#14, 500	62*m*28.783*s*	0*m*55.077*s*

 doi:10.20944/preprints202510.1014.v1

4. Conclusions

In this paper, a novel classification method is proposed, referred to as KOS (kernel-based optimal subspaces). The approach involves constructing optimal subspaces in the feature space, and classifying an unseen attribute vector based on its minimum distance to these subspaces. Some theoretical foundations are discussed, and proofs provided. The POD subspaces are proved to be optimal in the sense of being the smallest subspaces containing the feature classes, and the basis systems being the best features data representatives. All necessary formulations for practical implementation were derived.

The issue of imbalanced classes is addtressed a simple and intuitive mechanism, eliminating the need for additional preprocessing or resampling techniques. Experimental results on both 2D and high-dimensional multiclass datasets drawn from real-world scenarios show that KOS achieves performance comparable to optimized SVMs using LIBSVM. Beyond accuracy, KOS offers several practical advantages: it is easier to implement, significantly faster, more robust, imbalanced classes issue is systematically handled, suitable for dynamic classification tasks, and highly parallelizable. These strengths make KOS a promising alternative to traditional kernel-based classifiers in a variety of applications.

Data Availability Statement: All used data are public and available at: https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Acknowledgments: This research is supported by Alfaisal University grant IRG with reference IRG20411.

Conflicts of Interest: The author declare no conflicts of interest.

Appendix A. Proof of the POD Best Representatives

Let Y^k be the columns of the matrix A, the upper index is used here instead of lower index, which will be used to refer to the columns components. Y^1 satisfies the optimization problem,

$$V^1 = Arg\Big[Max_{V\in IR^n}\Big(\sum_{i=1}^{N}\Big(\big\langle V, Y^i\big\rangle\Big)^2\Big) \quad subject\ to \quad \|V\|^2 = 1\Big]. \tag{A1}$$

This constrained optimization problem can be solved using Lagrange multipliers:

$$L(V,\lambda) = \sum_{i=1}^{N}\Big(\big\langle V, Y^i\big\rangle\Big)^2 + \lambda(1-\|V\|^2) \quad (V,\lambda)\in IR^{m+1}, \tag{A2}$$

and the solution is obtained by nullifying the gradient

$$\nabla L(V,\lambda) = 0 \quad in \quad IR^m \times IR.$$

Differentiating L with respect of V

 doi:10.20944/preprints202510.1014.v1

$$\begin{aligned}
\frac{\partial L}{\partial V_j} &= \frac{\partial}{\partial V_j}\left(\sum_{i=1}^{N}\left(\sum_{k=1}^{m} V_k Y_k^i\right)^2 + \lambda(1-\sum_{k=1}^{m}(V_k)^2)\right) \\
&= \sum_{i=1}^{N}\frac{\partial}{\partial V_j}\left(\sum_{k=1}^{m} V_k Y_k^i\right)^2 + \frac{\partial}{\partial V_j}\left(\lambda(1-\sum_{k=1}^{m}(V_k)^2)\right) \\
&= \sum_{i=1}^{N} 2\left(\sum_{k=1}^{m} V_k Y_k^i\right) Y_j^i - 2\lambda V_j \\
&= 2\left(\sum_{k=1}^{m}\left(\sum_{i=1}^{N} Y_k^i Y_j^i\right) V_k - \lambda V_j\right) \\
&= 2\left(\sum_{k=1}^{N}\left(AA^T\right)_{k,j} V_k - \lambda V_j\right) \\
&AA^T \text{ is symmetric we can permute the indexes} \\
&= 2\left(\sum_{k=1}^{N}\left(AA^T\right)_{j,k} V_k - \lambda V_j\right).
\end{aligned} \tag{A3}$$

Then

$$\frac{\partial L}{\partial V_j} = 0 \quad \text{implies} \quad \sum_{k=1}^{N}\left(AA^T\right)_{j,k} V_k = \lambda V_j$$

and then, by setting $V = V^1$ and $\lambda = \lambda_1$

$$AA^T V^1 = \lambda_1 V^1.$$

This shows that V^1 is the eigenvector of the matrix AA^T and λ_1 is the associated eigenvalue. Now evaluate the maximum by substituting V^1,

$$\begin{aligned}
\sum_{i=1}^{N}\left\langle V^1, Y^i\right\rangle^2 &= \sum_{i=1}^{N}\left(\sum_{k=1}^{m} V_k^1 Y_k^i\right)\left(\sum_{j=1}^{m} V_j^1 Y_j^i\right) \\
&= \sum_{i=1}^{N}\left(\sum_{k=1}^{m}\sum_{j=1}^{m} V_k^1 Y_k^i V_j^1 Y_j^i\right) \\
&= \sum_{k=1}^{m}\sum_{j=1}^{m} V_k^1 V_j^1 \sum_{i=1}^{N} Y_k^i Y_j^i \\
&= \sum_{k=1}^{m}\sum_{j=1}^{m} V_k^1 V_j^1 (A^T A)_{k,j} \\
&= \left\langle AA^T V^1, V^1\right\rangle = \lambda_1\left\langle V^1, V^1\right\rangle = \lambda_1.
\end{aligned} \tag{A4}$$

We seek V^2 to be the second best representative with V^2 perpendicular to V^1. We seek V^2 in the orthogonal space to the one spanned by V^1. This can be formulated as follows:

$$\begin{aligned}
V^2 = Arg\Big[&Max_{V\in IR^N}\left(\sum_{i=1}^{N}|\langle V, Y_i\rangle|^2\right) \\
&\textit{subject to} \quad \|V\|^2 = 1 \ \textit{and} \ \left\langle V, V^1\right\rangle = 0\Big].
\end{aligned} \tag{A5}$$

 doi:10.20944/preprints202510.1014.v1

Note the matrix AA^T is symmetric, positive semi-definite, then it has m eigenvalues $\lambda_1 \geq \lambda_2 \geq ...\lambda_m \geq 0$ and m corresponding eigenvectors $V^1, ..., V^2$ that can be chosen to be orthonormal, that is,

$$\|V^i\| = 1 \quad \text{for} \quad i = 1, m \quad \text{and} \quad \left\langle V^i, V^j \right\rangle = 0, \quad \text{if} \quad i \neq j.$$

Since we look for a V orthogonal to V^1, $V \in span\{V^2, ..., V^m\}$ it can be expended as

$$V = \sum_{k=2}^{k=m} \left\langle V, V^k \right\rangle V^k.$$

Now estimate the quantity to maximize at an arbitrary V,

$$\begin{aligned}
\sum_{i=1}^{N} \left\langle V, Y^i \right\rangle^2 &= \sum_{i=1}^{N} \left\langle \sum_{k=2}^{m} \left\langle V, V^k \right\rangle V^k, Y^i \right\rangle^2 \\
&= \sum_{i=1}^{N} \left\langle \sum_{k=2}^{m} \left\langle V, V^k \right\rangle V^k, Y^i \right\rangle \left\langle \sum_{k=2}^{m} \left\langle W, Y^j \right\rangle V^j, Y^i \right\rangle \\
&= \sum_{i=1}^{N} \left(\sum_{k=2}^{m} \left\langle V, V^k \right\rangle \left\langle V^k, Y^i \right\rangle \right) \left(\sum_{k=2}^{m} \left\langle V, V^j \right\rangle \left\langle V^j, Y^i \right\rangle \right) \\
&= \sum_{i=1}^{N} \sum_{k=2}^{m} \sum_{j=2}^{m} \left\langle V, V^k \right\rangle \left\langle V^k, Y^i \right\rangle \left\langle V, V^j \right\rangle \left\langle V^j, Y^i \right\rangle \\
&= \sum_{i=1}^{N} \sum_{k=2}^{m} \sum_{j=2}^{m} \left\langle V, V^k \right\rangle \left\langle V, V^j \right\rangle \left\langle V^j, Y^i \right\rangle \left\langle V^k, Y^i \right\rangle \\
&= \sum_{k=2}^{m} \sum_{j=2}^{m} \left\langle V, V^k \right\rangle \left\langle V, V^j \right\rangle \sum_{i=1}^{N} \left\langle V^j, Y^i \right\rangle \left\langle V^k, Y^i \right\rangle.
\end{aligned}$$

We can check by expending in terms of components that,

$$\sum_{i=1}^{N} \left\langle V^j, Y^i \right\rangle \left\langle V^k, Y^i \right\rangle = \left\langle A^T A V^j, V^k \right\rangle,$$

and by orthogonality

$$\begin{aligned}
\left\langle AA^T V^j, V^k \right\rangle = 0 \quad \text{if} \quad j \neq k, \\
\text{and} \quad \left\langle AA^T V^j, V^j \right\rangle = \lambda_j,
\end{aligned}$$

therefore

$$\begin{aligned}
\sum_{i=1}^{N} \left\langle V, Y^i \right\rangle^2 &= \sum_{j=2}^{m} \left\langle V, V^j \right\rangle \left\langle V, V^j \right\rangle \lambda_j \\
&= \sum_{j=2}^{m} \left\langle V, V^j \right\rangle^2 \lambda_j \\
&\leq \sum_{j=2}^{m} \left\langle V, V^j \right\rangle^2 \lambda_2 \\
&= \lambda_2 \sum_{j=2}^{m} \left\langle V, V^j \right\rangle^2 = \lambda_2 \|V\|^2 = \lambda_2.
\end{aligned} \tag{A6}$$

 doi:10.20944/preprints202510.1014.v1

Now if we substitute V by V^2 following the same steps as in (A4) we obtain

$$\sum_{i=1}^{N} \left\langle V^2, Y^i \right\rangle^2 = \lambda_2,$$

which proves that the maximum is reached at V^2. We can repeat the same process and show that the p vectors we looking for are the eigenvectors of AA^T. For more details see [19].

References

1. Alpaydin, E. Introduction to Machine Learning (Fourth ed.). MIT. pp. xix, 1–3, 13–18. (2020). ISBN 978-0262043793.
2. Vapnik, V. The Nature of Statistical Learning Theory. *Springer-Verlag*, NY, USA, 1995.
3. Vapnik, V. Statistical Learning Theory. *Wiley, NY,* USA, 1998.
4. Cortes, C.; Vapnik, V. Support vector networks, *Machine Learning*, 20:1–25, 1995
5. Mercer, J. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society*, A(209):415–446, 1909.
6. Wang, H.; Li, G.; Wang, Z. Fast SVM classifier for large-scale classification problems. *Information Sciences*, Volume 642, September 2023.
7. Shao, Y.H.; Lv, X.J.; Huang, L.W.; Bai Wang, L. Twin SVM for conditional probability estimation in binary and multiclass classification. *Pattern Recognition*, Volume 136, April 2023.
8. Wang, H.; Shao, Y. Fast generalized ramp loss support vector machine for pattern classification. *Pattern Recognition*, Volume 146, February 2024.
9. Wang, B.Q.; Guan, X.P.; Zhu, J.W.; Gu, C.C.; Wu, K.J.; Xu, J.J. SVMs multi-class loss feedback based discriminative dictionary learning for image classification. *Pattern Recognition*, Volume 112, April 2021.
10. Borah P.; Gupta, D. UFunctional iterative approaches for solving support vector classification problems based on generalized Huber loss. *Neural Computing and Applications*, vol. 32, no. 1, pp. 1135–1139, 2020
11. Gaye, B.; Zhang, D.; Wulamu, A. Improvement of Support Vector Machine Algorithm in Big Data Background. *Hindawi Mathematical Problems in Engineering* Volume 2021,June 2021, Article ID 5594899, 9 pages DOI:10.1155/2021/5594899
12. Tian, Y.; Shi, Y.; Liu, X. Advances on support vector machines research. *Technological and Economic development of economy*. iSSN 2029-4913 print/iSSN 2029-4921 online, 2012 Volume 18(1): 5–33, doi:10.3846/20294913.2012.661205
13. Ayat, N.E.; Cheriet, M.; Remaki, L.; Suen, C.Y. KMOD-A New Support Vector Machine Kernel With Moderate Decreasing for Pattern Recognition. Application to Digit Image Recognition. P*roceedings of Sixth International Conference on Document Analysis and Recognition*, pp.1215-1219, 2001.
14. Lin, S.W.; Ying, K.C; Chen, S.C.; Lee, ZJ. Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert Systems with Applications*. 2008; 35: 1817-1824.
15. Syarif, I.; Prugel-Bennett, A.; Wills, G. SVM Parameter Optimization Using Grid Search and Genetic Algorithm to Improve Classification Performance. *TELKOMNIKA*, Vol.14, No.4, December 2016, pp. 1502-1509 ISSN: 1693-6930. DOI: 10.12928/TELKOMNIKA.v14i
16. Shekar, B.H.;G. Dagnew, G. Grid Search-Based Hyperparameter Tuning and Classification of Microarray Cancer Data. *Second International Conference on Advanced Computational and Communication Paradigms* (ICACCP)2019.
17. Hinton, E.; Osindero, S.; Teh, Y. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.
18. Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research* 13 (2012) 281-305
19. Volkwein, S. Proper orthogonal decomposition: Theory and reduced-order modelling Lecture Notes, University of Konstanz, 2013, 4
20. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. Smote: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321–357 (2002).
21. Chang, C.C.; Lin, C.J. LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at https://www.csie.ntu.edu.tw/ cjlin/libsvm/

 doi:10.20944/preprints202510.1014.v1

22. Wang, W.; Zhang, M.; Wang, D.; Jiang, Y. Kernel PCA feature extraction and the SVM classification algorithm for multiple-status, through-wall, human being detection. *EURASIP Journal of wireless communication and Networking* (2017) 2017:151. DOI 10.1186/s13638-017-0931-2.
23. Golub, T.R.; Slonim, D.K.; Tamayo, P.; Huard, C.; Gaasenbeek, M.; Mesirov, J.P.; Coller, H.; Loh, M.L.; Downing, J.R.; Caligiuri, M.A.; Bloomfield, C.D.; Lander, E.S. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531, 1999.
24. Hsu, C.W.; Chang, C.C,; C.J Lin. A practical guide to support vector classification. *Technical report, Department of Computer Science, National Taiwan University*, 2003.
25. Available online:http://archive.ics.uci.edu/ml/index.php
26. Available online: https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/
27. Guyon, I.; Gunn, S.; Ben Hur, A.; Dror, G. Result analysis of the NIPS 2003 feature selection challenge. *Advances in Neural Information Processing Systems*, volume 17. 2005.
28. Hsu, C.W.; Lin, C.J. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.
29. Hull, J.J. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, May 1994.
30. Remaki, L. Efficient Alternative to SVM Method in Machine Learning. *In: Arai, K. (eds) Intelligent Computing*. 2025. *Lecture Notes in Networks and Systems*, vol 1426. Springer, Cham.