LP SVM with a Novel Similarity Function Outperforms Powerful LP-QP-Kernel-SVM Considering Efficient Classification

Rezaul Karim[1,*], Mahmudul Hasan[2], Amit Kumar Kundu[1] and Ali Ahmed Ave[1]
[1]Department of Electrical and Electronic Engineering, Uttara University, Dhaka, Bangladesh
[2]Department of Electrical Engineering, University of British Columbia, Okanagan Campus
E-mail: rezkar@uttarauniversity.edu.bd, mahmud67@student.ubc.ca, amit31416@gmail.com, ali.ave@uttarauniversity.edu.bd
*Corresponding author

Keywords: classification, SVM, kernel, similarity function, QP, sparse, LP, efficiency, machine accuracy time (MAT), performance

Received: March 27, 2023

While the core quality of SVM comes from its ability to get the global optima, its classification performance also depends on computing kernels. However, while this kernel-complexity generates the power of machine, it is also responsible for the computational load to execute this kernel. Moreover, insisting on a similarity function to be a positive definite kernel demands some properties to be satisfied that seem unproductive sometimes raising a question about which similarity measures to be used for classifier. We model Vapnik's LPSVM proposing a new similarity function replacing kernel function. Following the strategy of "Accuracy first, speed second", we have modelled a similarity function that is mathematically well-defined depending on analysis as well as geometry and complex enough to train the machine for generating solid generalization ability. Being consistent with the theory of learning by Balcan and Blum [1], our similarity function does not need to be a valid kernel function and demands less computational cost for executing compared to its counterpart like RBF or other kernels while provides sufficient power to the classifier using its optimal complexity. Benchmarking shows that our similarity function based LPSVM poses test error 0.86 times of the most powerful RBF based QP SVM but demands only 0.40 times of its computational cost.

Povzetek: Za SVM je predlagana je nova funkcija podobnosti, ki zamenja funkcijo jedra, zahteva manj računanja in dosega visoko natančnost in hitrost.

1 Introduction

A frequent contemporary method in the problems of data classification in machine learning is to encode prior knowledge about data patterns (objects) through a kernel operation that takes in two patterns, maps into a higher dimensional feature space and outputs a number representing similarity or dissimilarity with the condition that it must form a positive semi definite (PSD) matrix after applied to all pairs of patterns. To assess the true structure and input-output relation of the data for solving many real-world problems efficiently with Support Vector Machine, appropriate selection of this kernel is a crucial issue where the PSD property of kernel similarity matrix ensures for the SVM to be solved efficiently by a convex quadratic programming. For further about kernel SVM based classification and sparse learning one could see these papers [2–10] . However, while the kernel theory is quite well-designed, there are some other issues that sometimes make this kernel less interesting to be used such as i) The formal statement of Mercer's condition is tough to verify. ii) While designing a kernel function for any learning problem, the usual perception is that the better kernel for working on a dataset would also serve as the better similarity function with respect to that data. On contrary, the SVM kernel deals with margin in a possibly very high-dimensional space that is also implicit and generally not perceptible in the innate demonstration of data. This turns out to be not so helpful for a designer to have the intuition to model or select an appropriate kernel for the learning task at hand. iii) There could be a question about the usefulness of an algorithm to be controlled by the characteristics of an implicit mapping that might not even be calculated by someone. Additionally, if functionally a kernel is just a black-box method taking two examples as input and giving an output number that gives some concept of how similar they are, it is not so clear if or why the positive semi-definiteness property is certainly needed— that is the physical significance of positive semi definiteness is not obvious although we know that positive semi definiteness is needed for the problem to be convex and have a unique minima. Moreover, the constraint of positive semi-definiteness may reject many natural similarity functions for a given problem and there are also many similarity function that are naturally not directly PSD but are proved to be potential, for example, the sigmoid function.

This motivates to work with possibly proper similarity

function relaxing kernel-constraints for data learning such as learning with non-PSD similarity or distance function. Learning with indefinite kernel or non-PSD similarity matrix has attracted huge concentration [11–19]. However, [20] have divided recent work on training SVM with indefinite kernels into three main kinds: PSD kernel approximation, non-convex optimization, and learning in Krein spaces with a conclusion that all methods are not fully adequate as they have either hosted bases of inconsistency in handling training and test patterns using kernel approximation which harms generalization guarantees or established for approximate local minimum solutions by non-convex optimization, or generated nonsparse solutions. But there is another approach that has been studied in a sequence of papers [1], [21], [13], [22] that adopt a certain "goodness" property, which is formally defined for the similarity function and provide both generalization guarantees in terms of how well-suited the similarity function is to the classification task at hand as well as the capability to use fast algorithmic techniques. Informally, a similarity function can be considered as good if patterns of same classes are closer to each other than patterns of different classes in some sense. The model proposed by [1], [22] developed a general theory of learning with pairwise similarity function that may not necessarily be a valid positive semi-definite kernel and sufficient conditions for the function for learning well that does not involve reference to implicit spaces, and nor demands the function to be PSD. Being inspired by this, in this paper we follow the formulation of SVM by modelling a Manhattan distance based similarity function replacing the kernel function.

The rest of paper is organized as follows. Section 2 provides a short related basics about SVM and kernel whereas Section 3 explains the proposed similarity function with analysis. In Section 4, the experimental results are shown with the description of data and model while conclusion with future work is presented in Section 5.

2 Related basics

As SVM and kernel are the core of a detector, we provide a short introduction about these here.

2.1 Support vector machine (SVM)

Support Vector Machines (SVMs) are advanced classifiers using a higher dimensional feature space and powerful tools for supervised classification. Two methods for constructing support vector machines are illustrated here where the first one is the conventional and standard method based on the quadratic programming (QP), which we term as QPSVM whereas the second one is based on the linear programming (LP) and we call this VLPSVM.

2.2 Quadratic programming SVM (QPSVM)

QPSVM finds the optimal separating hyperplane using margin maximization between two classes [23]. For the instance-label pairs of a training data set is $(x_i, y_i), i = 1, ..., N$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{1, -1\}^N$, the SVM classifier offers a decision function for finding class of the pattern x in the following form: $f(x) = sgn\big(w \cdot \phi(x) + b\big)$, where weight vector w, bias $b \in \mathbb{R}$ and $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$ is a kernel function and solves the following primal problem

$$\min_{w,b,\zeta} f_P(w) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\zeta_i \tag{1}$$

$$s.t.\ y_i\big(w \cdot \phi(x_i) + b\big) \geq 1 - \zeta_i; \tag{2}$$

$$\zeta_i \geq 0;\ i = 1, 2, ..., N \tag{3}$$

where ζ_i are a measure of the miss classification errors. The objective function above is a convex programming problem, where, C is a characteristic parameter of the classifier to be defined by the user that adjusts the trade off between optimal maximization of margin and to minimize the number of overall error on the training vectors. After few calculation, corresponding dual problem, becomes,

$$\max_{\alpha} f_D(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \phi(x_i) \cdot \phi(x_j) \tag{4}$$

$$s.t \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{5}$$

$$0 \leq \alpha_i \leq C;\ \ i = 1, 2, ..., N \tag{6}$$

The solution to the QP maximization problem (4)-(6) are used to obtain the values of the variables α_i and primal variable w is determined by using the α_i value from the expression $w = \sum_{i=1}^{N}\alpha_i y_i \phi(x_i)$. In addition, some KKT conditions are used to determine the bias b.

2.3 Vapnik's LP SVM (VLPSVM)

Vapnik suggested VLPSVM to build a separating hyperplane with support vectors minimization in number (heuristic SVM). An elaborate introduction along with mathematical background of both QP and LP based SVM could be found in [4].Vapnik formulated an alternative approach implementing linear programming to find a separating hyperplane similar to QPSVM considering the trade off between minimizing the summation of coefficients associated with the KCV (kernel computing vector, which plays very similar role as the Support Vectors (SVs) in QPSVM) and min-

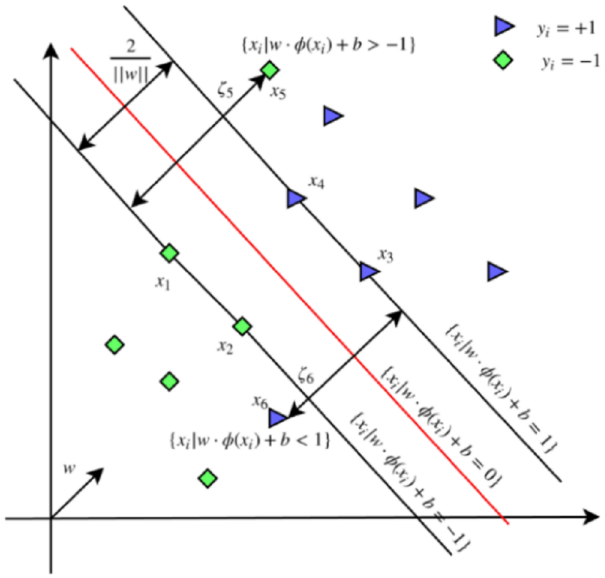

Figure 1: Pictorial representation of maximum margin concept based SVM [7]. Positive and negative patterns are represented by the triangles and the squares respectively. The red straight line is the optimal separating hyperplane and the black lines indicate the margin of the respective classes. Patterns that stay outwards from their classes are called outward-deviated patterns and amount of this deviation is represented by a non-negative variable, ζ. The objective of the QPSVM is to maximize the margin with minimum training error. The red line is also called the decision boundary of the training output.

imizing the error by finding w and b as below;

$$\min_{\lambda,\xi,b_V} \quad \sum_{i=1}^{N} \lambda_i \; + \; C_V \sum_{i=1}^{N} \xi_i \tag{7}$$

$$s.t. \;\; y_i\Big(\sum_{j=1}^{N} \lambda_j y_j \phi(x_j) \cdot \phi(x_i) + b_V\Big) \;\; \geq \;\; 1 - \xi_i \tag{8}$$

$$\lambda_j \;\geq\; 0; \; j = 1, 2, ..., N \tag{9}$$

$$\xi_i \;\geq\; 0; \; i = 1, 2, ..., N \tag{10}$$

where $\lambda_1, \lambda_2, ..., \lambda_N, \xi_1, \xi_2, ..., \xi_N, b_V$ are the optimization variable. The penalty parameter (C_V) in VLPSVM is defined by the user and $C_V > 0$ is in control of overfitting and learning the data. On the other hand, the slack variables (ξ_i) > 0 are used for the absolute-unity-outward-deviated patterns [6]. After solving the LP optimization problem of (7)-(10), the optimum λ values are used to calculate the weight vector w_V by using the equation $w_V = \sum_{j=1}^{N} \lambda_j y_j \phi(x_j)$ and the decision function are in the form of $f(x) = sgn\big(w_V \cdot \phi(x) + b_V\big)$.

2.4 Kernels

As the popularity of Support Vector Machines (SVMs) is increased in recent years, kernel methods have got major attention. Kernel functions, which can be expressed as dot product, are used to bridge from linearity to non-linearity in many applications. In addition, these methods map the data in a structured way into a higher dimensional space so that the data could be separated easily. Besides, kernel functions must have some important properties such as continuous, symmetric and most preferably, a positive semi-definite (meaning kernel matrices must have no negative Eigen values). Kernels satisfying Mercer's theorem are positive semi-definite that ensures the optimization problem is convex and has a unique solution. However, many so called kernel functions perform very well and are not positive semi-definite. For example, Sigmoid kernel which is not positive semi-definite for certain values of the parameters but it is used in wide range of applications. Some of the conventional kernels are Gaussian kernel, Polynomial Kernel, Bessel kernel among which Gaussian (also known as RBF) kernel is the most powerful and popular, which is given as $K(x, y) = \exp -\frac{\|x-y\|^2}{2\sigma^2}$ where, $\|x-y\|^2$ is recognized as the squared Euclidean distance (between the two feature vectors) and σ is an adjustable parameter, plays an important role in performance of the kernel.

3 Proposed distance based similarity function (DSF)

As in supervised learning, training examples are used to tell the classifier about the amount of dis/similarity among examples from opposite/own class, learning a sound classifier seems dubious if the given similarity function misses any innate "goodness" property where, naturally the goodness of a similarity function is needed be suitable to the classification problem at hand. To say more explicitly and specifically, the discriminators both from QPSVM and VLPSVM can be rewritten in the following common form by using a common term "kernel computing vector (KCV)" for SV or EV or such other kernel operating patterns: $f(z) = \sum_{i \in KCV set} \lambda_i y_i S(x_i, z) + b$ where z is any pattern and $S(x_i, z)$ is the kernel or similarity measure or strength of some sort of attraction force (between the patterns x_i and z), which is the source of main computational expenses and our main goal is to find a suitable mathematical expression (function) for this which is optimally complex(powerful) while not being much expensive to be executed. Thus, it is worthwhile to review notions of similarity, dissimilarity, and distance where we discuss about distance at first as below

3.0.1 Different distances

There are many distance functions out of which Euclidean distance and Manhattan distance seem to give the best results in distance based data learning [25]. They are given below

Euclidean Distance (ED): The Euclidean distance is the straight-line distance between two points in Euclidean

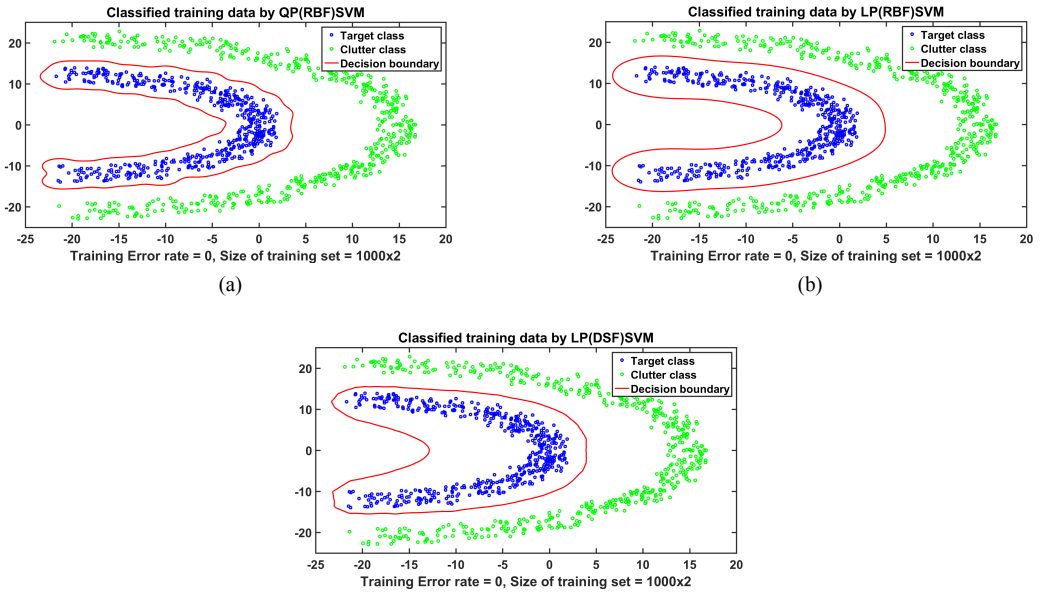

Figure 2: Decision boundaries with training error rates & data size from (a) QPSVM (RBF kernel), (b) VLPSVM (RBF kernel), & (c) VLPSVM (DSF) on an artificial dataset named Half kernel [24] before adding noise.

space and an extension to the Pythagorean Theorem. It is also known as L_2 norm or Ruler distance. The Euclidean distance between points x and y is expressed as: $ED(x,y) = \sqrt{\sum_{i=1}^{d} |x_i - y_i|^2}$. This distance is used in the so far the most powerful and popular kernel in machine learning, that is RBF kernel, with which we compare our similarity function considering classification efficiency basing on similarity measure.

Manhattan Distance (MD): The Manhattan distance represents the distance between two points as the sum of the absolute differences in Cartesian coordinates. It is also known as L_1 norm or City block distance. The Manhattan distance between points x and y is expressed as: $MD(x,y) = \sum_{i=1}^{d} |x_i - y_i|$ and we use this distance to model our similarity function.

3.0.2 Main idea

We start by considering the conclusion from [18], that is, objects that are similar in their representation are also similar in reality and belong, thereby, to the same class. The general idea is to transform a similarity into a dissimilarity function or vice versa by applying a monotonically decreasing function. This is according to the general intuition that a distance is small if the similarity is large, and vice versa.

Theorem: Similarity decreases as distance increases.

Proof: Let three points x, y, z are on the same straight path with distance between x and y, $d(x,y) = a > 0$, distance between y and z, $d(y,z) = b > 0$, and distance between x and z, $d(x,z) = a + b$. Then similarity between x and y, $s(x,y) \propto \frac{1}{a} \Rightarrow s(x,y) = \frac{K}{a}$, similarity between y and z, $s(y,z) \propto \frac{1}{b} \Rightarrow s(y,z) = \frac{K}{b}$ and similarity between x and z, $s(x,z) \propto \frac{1}{a+b} \Rightarrow s(x,z) = \frac{K}{a+b}$ for a constant K. Now as $a, b > 0$, we get $a^2, b^2, ab > 0$. Thus $a^2 + b^2 + ab > 0 \Rightarrow (a+b)^2 > ab \Rightarrow \frac{a+b}{ab} > \frac{1}{a+b} \Rightarrow \frac{1}{a} + \frac{1}{b} > \frac{1}{a+b} \Rightarrow \frac{K}{a} + \frac{K}{b} > \frac{K}{a+b} \Rightarrow s(x,z) < s(x,y) + s(y,z)$. Now, let p be another point on the same straight path further outside of z and distance between z and p, $d(z,p) = c$. then the similarity between x and p, $s(x,p) < s(x,z) + s(z,p) < s(x,y) + s(y,z) + s(z,p)$ and $s(x,p) = \frac{K}{a+b+c} < s(x,z) = \frac{K}{a+b} < s(x,y) = \frac{k}{a}$,which proves the statement basing on the induction method.

3.0.3 Proposed DSF (distance-similarity function) expression

We form our similarity function in the following way: For any two patterns, x, y, first we select a well reputed metric distance (or equivalently, dissimilarity) function using L_1-distance , $D(x,y)$ and transform this dissimilarity (distance) into a similarity. So, we define our similarity function S that generates a numeric value to represent similarity between x and y such that $S(x,y) = \frac{1}{\frac{D(x,y)}{\sigma}+1}$, where σ is a data related parameter and $\sigma > 0$. Although there are a number of ways to convert between a distance metric and a similarity measure, we select this one as to have i) Non-linear mapping ii) Less computational cost iii) scaling to keep value between 0 and 1 for numerical efficiency.

Hence, our proposed Distance Similarity Function (DSF) is expressed as- $S(x,y) = \frac{1}{\frac{\sum_{i=1}^{d} |x_i - y_i|}{\sigma}+1}$ If x coincides with y or $x \to y$ then $|x_i - y_i| \to 0 \Rightarrow S(x,y) \to 1$, representing the highest similarity between x and y whereas if x moves infinitely far away from y, then

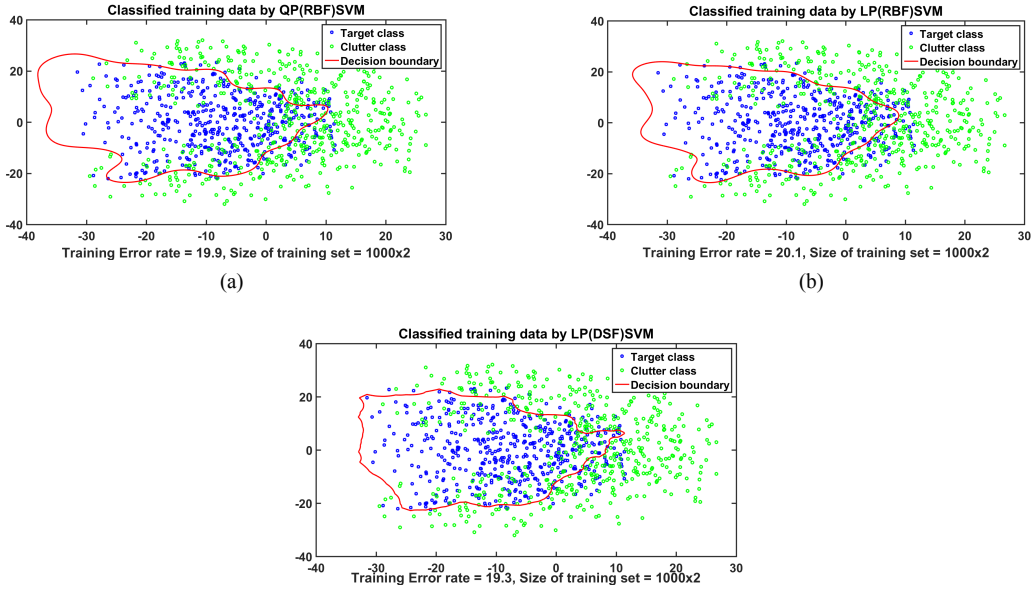

Figure 3: Decision boundaries with training error rates & data size from (a) QPSVM (RBF kernel), (b) VLPSVM (RBF kernel), & (c) VLPSVM (DSF) on an artificial dataset named Half kernel [24] after adding noise.

$|x_i - y_i| \to \infty \Rightarrow S(x,y) \to 0$ representing the lowest similarity between x and y

From the expression of similarity function above we see that $\sigma \to \infty \Rightarrow S(x,y) \to 1$ representing x and y have the highest similarity and $\sigma \to 0 \Rightarrow S(x,y) \to 0$ representing x and y have the lowest similarity. So, if σ is overestimated, the expression will behave almost linearly and non-linear mapping will start to lose its power while on contrary if underestimated, the function will lack regularization and the decision boundary will be highly sensitive to noise in training data. Thus, this adjustable parameter σ plays a very significant role in the performance of the similarity function and should be carefully tuned according to the problem at hand. And for any $\epsilon > 0$ increment in the distance, the distance will become $d + \epsilon$ and our similarity function S will decrease as $1/(d+\epsilon)$ will go down and the opposite for the decrement of the distance, that is $1/(d-\epsilon)$ will rise from the decrement of the distance by $\epsilon > 0$. This proves that our similarity function is a monotonic function whose value increases/decreases with the decrements/increment of distance between two patterns. A rough graphical representation of our similarity function (DSF) could be seen in figure below (Fig.4).

Non-linear mapping with mathematical expression: The motivation for such an embedding comes with the hope that the nonlinear transformation of input data into higher dimensional H allows for using linear techniques in H. Non-linear classification in lower dimension is linear in higher dimension. Our similarity function (f) does so in the following way:

$f(d) = \frac{1}{\frac{d}{constant}+1}$ which could be written as $f(d) = \frac{1}{z+1}$ where $z = \frac{d}{constant}$ which could be expanded following two cases as below:

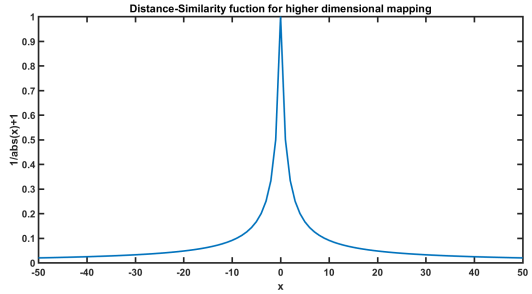

Figure 4: Distance-Similarity Function (DSF) using parameter, $\sigma = 1$.

Case I: For $|z| < 1$

$\frac{1}{z+1} = 1 - z + z^2 - z^3 ...$ and $\frac{1}{2} < \frac{1}{z+1} \leq 1$

Case II: For $|z| > 1$

$\frac{1}{z+1} = \frac{1}{z}\frac{1}{1+\frac{1}{z}} = \frac{1}{z}\left(1 - z^{-1} + z^{-2} - z^{-3} ...\right) = z^{-1} - z^{-2} + z^{-3} - z^{-4} ...$ and $0 \leq \frac{1}{z+1} < \frac{1}{2}$

3.1 Goodness of our similarity function

[1] provided a criterion for a good similarity function to be used in a discriminator where it is said approximately that a similarity function is good if the generated mean innerclass similarity is sufficiently large compared to the mean interclass similarity. After calculating the pattern-similarities on benchmark data using the following way, we have found out that our proposed function successfully obeys this in general without any deviation.

Calculating Similarity: Sum of similarities of a pattern

x_j with all patterns x_i for $i = 1, 2, ..., N$, $S(x_j, all) = \sum_{i=1}^{N} S(x_i, x_j)$. Then average similarity (of all patterns with other patterns) = $mean_i(mean_j(S(x_i, x_j)))$. On Breast Cancer dataset [26] (or [27]), the average similarity of the training patterns using our proposed (DSF) function 0.3614 to own class and 0.3235 to opposite class, which is consistent with the theorem of Balcan-Blum in [1].Whereas, on this dataset, the average kernel (or, similarity) value of the training patterns using Radial Basis Function (RBF) is 0.6267 to own class and 0.5748 to opposite class, which is also consistent with the theorem of Balcan-Blum in [1]. However, comparing these values from these two functions we see that DSF gives comparatively a bit higher Class Variational Similarity Deviation (= $\frac{similarity\ to\ own\ class - similarity\ to\ opposite\ class}{similarity\ to\ own\ class}$), which is the higher the better for classification by recognising discrepancy of patterns from different classes) as it is (0.3614 - 0.3235) / 0.3614 = 0.1049 in case of DSF and (0.6267 - 0.5748) / 0.6267 = 0.0828 in case of RBF based kernel function. The similarity values from these two functions can be seen more detailed from the figure below (Fig. 5)

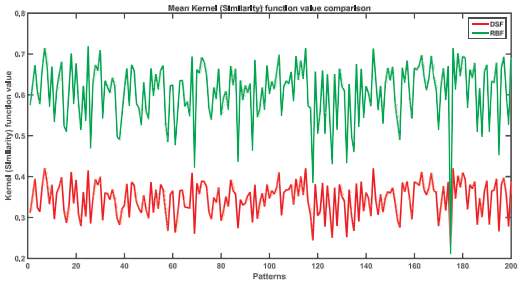

Figure 5: Mean Kernel (Similarity) function value comparison between RBF kernel and DSF.

However, values generated by similarity (or kernel) function operated on patterns heavily depends on pattern-scattering or data complexity. If a pattern from one class is very near to the patterns of the other class, its overall distance to patterns of opposite class is smaller than such of the patterns of its own class. Consequently, similarity behaves inversely. Thus we further investigate these similarity values by going into a bit deeper basing on numerical and functional analysis of pattern.

3.1.1 Numerical and functional analysis of pattern

We can determine Patterns' position from decision function value and the slack variable. Considering the geometric position of the patterns in the pattern space, we can divide them into two main parts as inlier and outlier . A pattern is said to be "inlier" if it belongs to the region of its own class boundary and similarly, an "outlier" pattern of a training set is the one that stays outside of its own class boundary. Hence a training pattern, x_j with class label y_j and decision function value $f(x_j)$, will be an inlier if $y_j = sgn(f(x_j)) \implies 1 = y_j sgn(f(x_j)) \implies y_j f(x_j) > 0$ and outlier if $y_j = -sgn(f(x_j)) \implies -1 = y_j sgn(f(x_j)) \implies y_j f(x_j) < 0$. Now, re-formatting the error constraints of SVM in (2) and in (8) into a single form using the slack variable ξ we get $y_j f(x_j) \geq 1 - \xi_j$ which gives $\xi_j \geq 1 - y_j f(x_j)$ leading to $0 \leq \xi_j \leq 1$ in case of inlier patterns and $\xi_j > 1$ for outlier patterns. We discuss further about them below.

In case of inlier patterns: As these patterns stay inside of the class-decision boundaries of their own classes, generally their overall distances to the patterns of their own classes are smaller compared to the distances to the patterns of their opposite classes and the similarity values will behave inversely that is their overall similarities to the patterns of their own classes are higher compared to the similarities to the patterns of their opposite classes. For each inlier, we calculate the ratio of the sum of its similarity to the patterns of its own class to the sum of its similarity to the patterns of its opposite class(that is, own-class/opposite class) in the way given below and we expect it to be larger than 1. For any inlier pattern x_i with class label y_i, sum of similarities to patterns of its own class is, $SumSimOwn_i = \sum_m S(x_i, x_m) | (sgn(f(x_i)) = y_i = y_m)$ and sum of similarities to patterns of its opposite class is, $SumSimOps_i = \sum_n S(x_i, x_n) | (sgn(f(x_i)) = y_i = -y_n)$. Then their similarity ratio,$RatSumSim_i^{mn} = \frac{SumSimOwn_i}{SumSimOps_i} = \frac{\sum_m S(x_i, x_m) | (sgn(f(x_i)) = y_i = y_m)}{\sum_n S(x_i, x_n) | (sgn(f(x_i)) = y_i = -y_n)}$ The inlier related similarity ratio values from these two functions can be seen more detailed from the following figures (Fig. 6 and 7)

In case of outlier patterns:

In case of a complex or very noisy dataset, a well generalized SVM is expected to introduce more outliers.

As these patterns stay outside of their own classes, they behave nearly in the oppoiste way compared to inlier patterns in case of pattern distance or similarity. So, for these patterns, generally their overall distances to the patterns of their own classes are larger compared to the distances to the patterns of their opposite classes consequently, the similarity values behave inversely that is their overall similarities to the patterns of their own classes are smaller compared to the similarities to the patterns of their opposite classes. For each outlier, we calculate the ratio of the sum of its similarity to the patterns of its opposite class to the sum of its similarity to the patterns of its own class(that is, opposite class/own class) in the way given below and we expect it to be larger than 1. For any outlier pattern x_o with class label y_o, sum of similarities to patterns of its opposite class is, $SumSimOps_o = \sum_n S(x_o, x_n) | (sgn(f(x_o)) = -y_o = y_n)$ and sum of similarities to patterns of its own class is, $SumSimOwn_o = \sum_m S(x_o, x_m) | (sgn(f(x_o)) = -y_o = -y_m)$. Then their similarity ratio,$RatSumSim_o^{nm} = \frac{SumSimOps_o}{SumSimOwn_o} = \frac{\sum_n S(x_o, x_n) | (sgn(f(x_o)) = -y_o = y_n))}{\sum_m S(x_o, x_m) | (sgn(f(x_o)) = -y_o = -y_m)}$ The outlier related similarity ratio values from these two functions can be seen more detailed from the following figures (Fig. 8 and 9)

Why Manhattan distance based kernel and

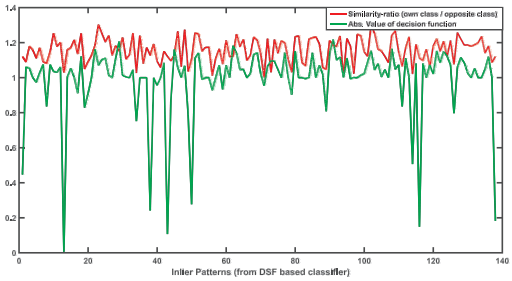

Figure 6: Figure showing similarity ratio of inlier patterns (with their corresponding pattern function (absolute) values; higher function value means the pattern stays deeper in its own class) to the patterns of its own class with respect to the similarities of the patterns of opposite class (that is, own class/opposite class) on Breast Cancer dataset using **our similarity function (DSF)**. As inlier patterns stay inside the decision boundary of their own classes, for them, generally, similarities to the patterns of their own classes are higher compared to the similarities to the patterns of their opposite classes; hence ratio of the sum of similarities to own class with respect to opposite class should be higher for a good similarity function. This is also the case for our similarity function. However, for very few patterns, which are placed in very special and isolated places of their own class, this may be different depending on their geometric positions. Amazingly, it can also be seen in the figure that these similarity ratio values are higher for the patterns staying deeper inside of their classes; quite as expected.

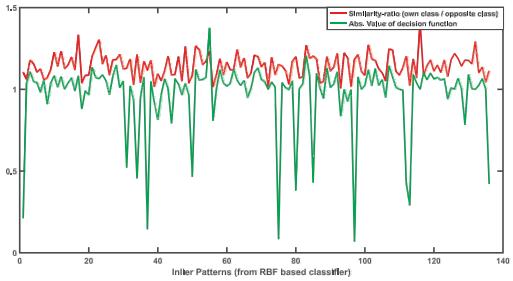

Figure 7: Figure showing similarity ratio of inlier patterns (with their corresponding pattern function (absolute) values) to the patterns of its own class with respect to the similarities of the patterns of opposite class (that is, own class/opposite class) on Breast Cancer dataset using **RBF kernel**. Interestingly, it behaves nearly in the very same way as our similarity function. Note that, RBF and DSF based SVMs are two different machines, so they have two different discriminators. Thus, they may produce different function values for the same patterns. By this, these two machines may lead to different outlier, inlier numbers from the same data set and patterns that are detected as inliers/outliers by RBF SVM may not be detected as inliers/outliers by DSF SVM. This is why number of variables in the horizontal axis of the two plots are different.

LPSVM: Comparing to the L2 norm (also known as Euclidean norm, which gives the ordinary distance using Pythagorean theorem from the origin to the point) used in QPSVM, L1 norm leads to a much sparser solution [28] in case of VLPSVM. This is useful to minimize the kernel computation by minimizing the number of kernel computing vector (KCV), the bases patterns that build the discriminator function of a machine by executing kernel operations (Support Vectors or Expansion Vectors or Basis Vectors or Machine Vectors, however it is named by individual author for various machines). Moreover, some other issues regarding distance have inspired to select this distance as i) Problem with Euclidean distance as a family of the Minkowski metric is that the largest-scaled feature would dominate the others [29] ii)Euclidean distance is noise sensitive [30] iii) Manhattan distance function, requires less Computation [31] iv) high dimensionality is sensitive to the value of k using the L_k based distance learning, which means that the Manhattan distance metric (L_1 norm) is consistently more preferable than the Euclidean distance metric (L_2 norm) for high dimensional data mining applications [32] v)In case of Manhattan Distance, distance between two exterior points a and z through another interior point y can be written as $d(x, z) = d(x, y) + d(y, z)$. So, the distances are linearly related, which matches more with our distance-inverted similarity measure. vi) By taking just the summation of the absolute values Manhattan Distance considers only the real topological distance but by taking squared values Euclidean distance sometimes may lead to miss interpretation, for example, when $0 < a < 1$, $a^2 < a$

4 Experiments & results

4.1 Machine's performance indicating terms

Test error rate is the most important term to evaluate a machine's performance in a conventional way. To get the complete evaluation and information regarding performance of a machine, classification time is also considered to be an important factor along with the test error rate. Therefore, a novel term called "Machine Accuracy Time (MAT)" is introduced to consider the computation time coupled with test accuracy. A machine with higher test accuracy could take much longer time to classify the test data which is not always useful in practical purposes. In contrast, a machine with faster classification allowing slightly higher or similar error rate could be acceptable and more practical considering the computational time as well as many real time applications.

Machine accuracy time (MAT): In classification problem, specially, in sparse learning, the prime objective is to build a machine efficiently having high accuracy and compatible computational cost. The number of computing kernel should be kept minimum to reduce this classification cost in such machine. Along with the minimum classifica-

Table 1: No. of KCVs (Kernel Computing Vectors), & Test Error Rate on Benchmark Data for Different Machines

Dataset	No. of train pat-tern	No. of test pat-tern	Data Di-men-sion	QPSVM mean SVs(SD) RBF kernel	QPSVM mean TeER (SD) RBF kernel	VLPSVM mean SVs(SD) RBF kernel	VLPSVM mean TeER (SD) RBF kernel	VLPSVM mean SVs(SD) DSF	VLPSVM mean TeER (SD) DSF	TeER $\frac{DSF(LP)}{RBF(QP)}$	TeER $\frac{DSF(LP)}{RBF(LP)}$
BREAST CANCER	200	77	9	200 (0)	28.53 (4.32)	18.89 (2.44)	26.05 (4.50)	27.1 (5.01)	25.47 (4.65)	0.89	0.98
FLARE SOLAR	666	400	9	609.98 (27.57)	32.41 (1.80)	247.87 (182.50)	32.56 (1.72)	157.38 (34.92)	32.23 (1.83)	0.99	0.99
HEART	170	100	13	68.23 (6.01)	16.6 (3.05)	21.94 (2.61)	17.44 (3.49)	33.84 (3.87)	16.96 (3.61)	1.02	0.97
IMAGE	1300	1010	18	237.4 (24.32)	4.19 (0.61)	75.05 (5.54)	4.07 (0.57)	170.3 (6.65)	2.19 (0.58)	**0.52**	**0.54**
SPLICE	1000	2175	60	943.65 (7.86)	12.34 (0.73)	292.15 (15.68)	12.39 (0.96)	308.45 (11.52)	7.59 (0.73)	**0.62**	**0.61**
THYROID	140	75	5	43.51 (3.10)	5.2 (2.08)	8.97 (1.59)	5.09 (2.11)	77.09 (3.66)	3.55 (2.04)	**0.68**	**0.70**
TWO-NORM	400	7000	20	299.18 (7.00)	2.42 (0.14)	32 (5.03)	3.71 (0.55)	123.05 (6.51)	3.1 (0.24)	1.28	0.84
Average				343.14 (10.84)	14.53 (1.82)	99.55 (30.77)	14.47 (1.99)	128.17 (10.31)	13.01 (1.95)	0.86	0.80

In Table 1, the number KCVs(Kernel Computing Vector) of different state-of-the-art machines along the proposed DSF based machine with the test error rates due to these machines on different datasets of benchmark data [27]. It is worth to note that while these machines work for sparsification of SVM based classifier using any specific kernel, our this method works for finding a good replacement of a kernel function.

tion time, the test error is also kept at a minimum rate to get the optimum performance from the machine. To calculate such type of property, MAT (Machine Accuracy Time) is defined, where, $MAT = \frac{TestAccuracyRate}{MeanClassificationTime}$. Therefore, to get a machine with the maximum test accuracy along with having the minimum classification time, will have the maximum MAT which is the most desirable case. Conversely, a machine with the highest classification time and the lowest accuracy rate will have the minimum MAT, which is never desired. In our case, conventional SVM is implemented to compare the effect on performance using RBF kernel (in QPSVM & VLPSVM) and the proposed Distance Similarity Function (in LPSVM).

4.2 Experimental setup & results

In machine learning, model selection refers to the problem of choosing a good kernel function while the class of model is a parametric problem with each value corresponds to some model. In our case, similarity function is fixed. Hence, choosing both σ and C parameters are the model selection problem and in our experimental work, the results are collected by implementing our distance-similarity based function (DSF) using Vapnik's Linear Programming SVM (VLPSVM) and compared with the RBF kernel on both QPSVM and VLPSVM. All the experiments were done on seven benchmark data sets [26] (or [27]) i.e. Breast Cancer, Flare Solar, Heart, Image, Splice, Thyroid, Twonorm. In both the DSF and RBF kernel, the penalty parameters C, C_V and kernel parameters σ and σ_V were obtained by cross validation using the range $C, C_V \in \{2^{-2}, 2^0, 2^2, ..., 2^{12}\}$ and $\sigma, \sigma_V \in \{2^{-2}, 2^0, 2^2, ..., 2^6\}$.

In Table 1, the comparison was between RBF kernel (using both QPSVM and LPSVM) and the DSF (using LPSVM) regarding the test error rate and the number of KCVs. In terms of the RBF kernel, the average test error rate of all the data sets was accounted for 14.53 with SD (Standard Deviation) of 1.82 using QPSVM and 14.47 (SD 1.99) using LPSVM. Conversely, the average value of the test error rate was noted as 13.01 (SD 1.95) for the DSF. Hence, after counting the error rate ratio (i.e. DSF-error rate/X-machine error rate) it is found that DSF one is 86% of QPSVM (i.e. 14% reduced) and 80% of LPSVM (i.e. 20% reduced) on average (column 11, 12 of table 1).

Table 2 illustrates the classification times and their ratios implementing DSF to the RBF-based SVM using both QP and LP (i.e. DSF-classification time/X-machine classification time). With regards to the RBF kernel using QP, DSF performs better in terms of classification time for all the data sets except THYROID and overall, the novel function performs 60% faster. On the other hand, DSF computes faster for 4 out of 7 data sets than RBF kernel in VLPSVM

Table 2: Classification Time For Different Machines

Dataset	QP(RBF)	LP(RBF)	LP(DSF)	$\frac{LP(DSF)}{QP(RBF)}$	$\frac{LP(DSF)}{LP(RBF)}$
BREAST CANCER	0.024177	0.003560	0.003323	**0.137446**	0.933351
FLARE SOLAR	0.358267	0.150766	0.066410	**0.185366**	**0.440487**
HEART	0.011770	0.005137	0.004635	**0.393824**	0.902411
IMAGE	0.384581	0.130776	0.179741	**0.467368**	1.374411
SPLICE	3.254468	1.068450	0.740747	**0.227609**	0.693291
THYROID	0.006084	0.002074	0.006718	1.104236	3.238635
TWONORM	3.251135	0.443706	0.959134	**0.295015**	2.161642
Average	1.041497	0.257781	0.280101	0.401552	1.392032

Table 2 presents the classification time by different machines along the proposed DSF based one.

Table 3: Machine Accuracy Time (MAT) for Different Machines

Dataset	QP(RBF)	LP(RBF)	LP(DSF)	$\frac{LP(DSF)}{QP(RBF)}$	$\frac{LP(DSF)}{LP(RBF)}$
BREAST CANCER	2955.9565	20769.6	22428.5488	**7.5876**	1.0799
FLARE SOLAR	188.6721	447.3165	1020.4734	**5.4087**	**2.2813**
HEART	7085.8258	16072.97	17914.6633	2.5282	1.1146
IMAGE	249.1334	733.5476	544.1563	2.1842	0.7418
SPLICE	26.9366	81.9944	124.7514	**4.6313**	**1.5215**
THYROID	15582.71	45754.27	14357.8516	0.9214	0.3138
TWONORM	30.0133	217.0204	101.0276	**3.3661**	0.4655
Average	3731.3211	12010.9598	8070.2103	3.8039	1.0741

Table 3 presents the MAT values of different machines along the proposed DSF based one.

and overall it takes around 39% more time for classification.

To get a clear picture about the performance of the machine, MAT (Machine Accuracy Time) have been calculated considering the test accuracy rate and classification time of the test set. Table 3 contains the mean classification time on test data sets and corresponding MAT values for both RBF kernel based SVM and DSF based LPSVM. The DSF based machine performs better in terms of MAT for 6 out of 7 data sets except for Thyroid compared to RBF kernel in QP. Overall, DSF performs 280% better than RBF kernel using QP. In comparison with the RBF kernel in LP, the DSF shows slightly better performance (higher MAT value) for most of the data sets (4 out of 7) and in average the performance is improved over 7%.

5 Conclusion and future work

In this paper, we presented a novel similarity function replacing kernel function for classification with SVM. We have investigated about its ”goodness” using some numerical analysis and compared it with the most powerful and popular kernel function, RBF. It is found that while both of these satisfy the theory of learning with similarity functions by Balcan-Blum [1], our one gives a bit higher ”class variational similarity deviation”, which is more useful for better classification. We have also analyzed the behavior of these (kernel and similarity) functions using some Pattern Analysis considering the geometric position of patterns, their functional values by the discriminators, and kernel or similarity values in pairs.It is realised that the two functions (RBF, and our similarity function, DSF) behave quite expectedly and consistently as well as in similar fashion.Further, the effectiveness of the proposed function was also tested on benchmark Machine Learning datasets, which showed a countable improved performance compared to RBF kernel with respect to both accuracy and computational cost. However, while our method beats others most of the time and consistently, it also slightly fails such in few cases proving the no free lunch theorem that is, no method performs better 100% times.

Moreover, [20] discussed about SVM with indefinite kernels where it is showed that similarity function used for L_1-norm or LP SVM does not need to be positive semi/definite and we use LP SVM which remains convex even if the similarity matrix is indefinite. Although in some cases compared to the sparse machines, our similarity based SVM in-

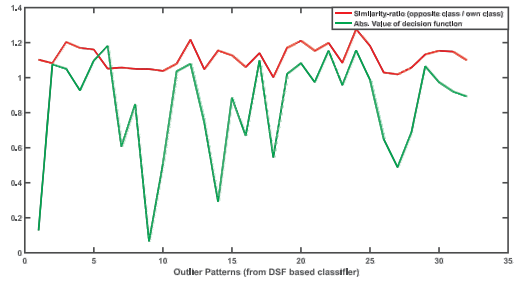

Figure 8: Figure showing similarity ratio of outlier patterns (with their corresponding pattern function (absolute) values; higher function value means the pattern stays deeper in its opposite class) to the patterns of its opposite class with respect to the similarities of the patterns of own class (that is, opposite class/own class) on Breast Cancer dataset using **our similarity function(DSF)**. As outlier patterns stay outside the decision boundary of their own classes, for them, generally, similarities to the patterns of their opposite classes are higher compared to the similarities to the patterns of their own classes; hence, ratio of the sum of similarities to opposite class with respect to own class should be higher for a good similarity function. This is also the case for our similarity function. However, for very few patterns that are placed in very special and isolated place of their opposite class, this may be different depending on their geometric positions. Amazingly, it can also be seen in the figure that these similarity ratio values are higher for the patterns staying deeper outside of their classes; quite as expected.

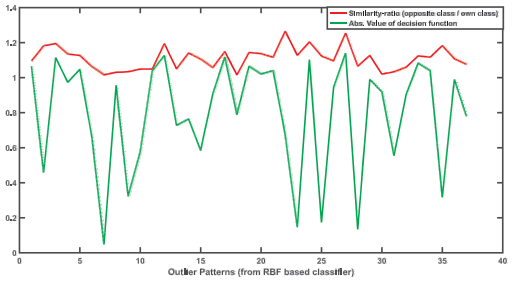

Figure 9: Figure showing similarity ratio of outlier patterns (with their corresponding pattern function (absolute) values) to the patterns of its opposite class with respect to the similarities of the patterns of own class (that is, opposite class/own class) on Breast Cancer dataset using **RBF kernel**. Interestingly, it behaves nearly in the very same way as our similarity function. Note that, RBF and DSF based SVMs are two different machines, so they have two different discriminators. Thus, they may produce different function values for the same patterns. By this, these two machines may lead to different outlier, inlier numbers from the same data set and patterns that are detected as inliers/outliers by RBF SVM may not be detected as inliers/outliers by DSF SVM. This is why number of variables in the horizontal axis of the two plots are different.

volves more number of kernel computing vectors to classify each pattern, each computation with our similarity function demands less execution cost compared to these or other conventional kernel functions. So far, we have noticed the following issues relating our work, which we want to study with deep investigation and evaluation in future i) Although we have checked our similarity function using Manhattan distance measure, there are many other distance measures that could be modelled to construct similarity function to develop SVM or other machine to test and evaluate for optimal performance. ii) Datasets with much higher dimension and sample size are also important to be used to execute experiments in order to find significant conclusions about performance iii) Similarity measures are not used only with the SVM, but also with other machine learning algorithms, which are needed to be evaluated under different distance measures iv) Go into deeper theoretical analysis to find relation between our similarity function and PSD kernel.

Acknowledgement

Thanks to the reviewers for constructive guidance.

References

[1] M.-F. Balcan, A. Blum, and N. Srebro, "A theory of learning with similarity functions," *Machine Learning*, vol. 72, no. 1, pp. 89–112, 2008. [Online]. Available: https://doi.org/10.1007/s10994-008-5059-5

[2] N. Cristianini, J. Shawe-Taylor *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000. [Online]. Available: https://doi.org/10.1017/cbo9780511801389

[3] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Adaptive Computation and Machine Learning Series, 2018. [Online]. Available: https://doi.org/10.7551/mitpress/4175.001.0001

[4] V. Vapnik, *Statistical learning theory*. Wiley, 1998. [Online]. Available: https://www.scribd.com/doc/191179292/Vladimir-N-Vapnik-Statistical-Learning-Theory-BookFi-org

[5] R. Karim, M. Bergtholdt, J. H. Kappes, and C. Schnörr, "Greedy-based design of sparse two-stage svms for fast classification," in *Proc. of the 29th DAGM Symposium on Pattern Recognition*, 2007, pp. 395–404. [Online]. Available: https://doi.org/10.1007/978-3-540-74936-3_40

[6] R. Karim and A. K. Kundu, "Efficiency and performance analysis of a sparse and powerful second order svm based on lp and qp," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 9, no. 2, pp. 311–318, 2018. [Online]. Available: https://doi.org/10.14569/ijacsa.2018.090244

[7] ——, "Computational analysis to reduce classification cost keeping high accuracy of the sparser lpsvm," *International Journal of Machine Learning and Computing*, vol. 9, no. 6, pp. 728–733, 2019. [Online]. Available: https://doi.org/10.18178/ijmlc.2019.9.6.865

[8] T. Joachims and C. N. J. Yu, "Sparse kernel svms via cutting-plane training," in *Proc. of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML)*, 2009, p. 8. [Online]. Available: https://doi.org/10.1007/978-3-642-04180-8_8

[9] S. S. Keerthi, O. Chapelle, and D. DeCoste, "Building support vector machines with reduced classifier complexity," *Journal of Machine Learning Research, (JMLR)*, vol. 7, no. Jul, pp. 1493–1515, 2006. [Online]. Available: https://dl.acm.org/doi/10.5555/1248547.1248602

[10] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998. [Online]. Available: https://doi.org/10.1023/A:1009715923555

[11] Y. Ying, C. Campbell, and M. Girolami, "Analysis of svm with indefinite kernels," *Advances in neural information processing systems*, vol. 22, pp. 2205–2213, 2009. [Online]. Available: https://seis.bristol.ac.uk/ enicgc/pubs/2009/indefiniteSVM_NIPS09_final.pdf

[12] J. Chen and J. Ye, "Training svm with indefinite kernels," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 136–143. [Online]. Available: https://doi.org/10.1145/1390156.1390174

[13] T. Graepel, R. Herbrich, P. Bollmann-Sdorra, and K. Obermayer, "Classification on pairwise proximity data," *Advances in neural information processing systems*, pp. 438–444, 1999. [Online]. Available: https://doi.org/10.1007/978-1-4615-5029-7_7

[14] B. Haasdonk, "Feature space interpretation of svms with indefinite kernels," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 4, pp. 482–492, 2005. [Online]. Available: https://doi.org/10.1109/tpami.2005.78

[15] H.-T. Lin and C.-J. Lin, "A study on sigmoid kernels for svm and the training of non-psd kernels by smo-type methods," *submitted to Neural Computation*, vol. 3, pp. 1–32, 2003. [Online]. Available: https://www.researchgate.net/publication/2478380

[16] R. Luss and A. d'Aspremont, "Support vector machine classification with indefinite kernels," *Mathematical Programming Computation*, vol. 1, no. 2, pp. 97–118, 2009. [Online]. Available: https://doi.org/10.1007/s12532-009-0005-5

[17] C. S. Ong, X. Mary, S. Canu, and A. J. Smola, "Learning with non-positive kernels," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 81. [Online]. Available: https://doi.org/10.1145/1015330.1015443

[18] E. Pekalska, P. Paclik, and R. P. Duin, "A generalized kernel approach to dissimilarity-based classification," *Journal of machine learning research*, vol. 2, no. Dec, pp. 175–211, 2001. [Online]. Available: https://dl.acm.org/doi/pdf/10.5555/944790.944810

[19] G. Wu, E. Y. Chang, and Z. Zhang, "An analysis of transformation on non-positive semidefinite similarity matrix for kernel machines," in *Proceedings of the 22nd International Conference on Machine Learning*, vol. 8. Citeseer, 2005. [Online]. Available: https://www.researchgate.net/publication/228341630

[20] I. Alabdulmohsin, X. Gao, and X. Z. Zhang, "Support vector machines with indefinite kernels," in *Asian Conference on Machine Learning*. PMLR, 2015, pp. 32–47. [Online]. Available: http://proceedings.mlr.press/v39/alabdulmohsin14.pdf

[21] L. Wang, C. Yang, and J. Feng, "On learning with dissimilarity functions," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 991–998. [Online]. Available: https://doi.org/10.1145/1273496.1273621

[22] M.-F. Balcan, A. Blum, and N. Srebro, "Improved guarantees for learning via similarity functions," 2008. [Online]. Available: https://www.researchgate.net/publication/221497236

[23] A. Nefedov, J. Ye, C. Kulikowski, I. Muchnik, and K. Morgan, "Experimental study of support vector machines based on linear and quadratic optimization criteria," *DIMACS Technical Report 2009-18*, 2009. [Online]. Available: https://doi.org/10.1109/icmla.2009.52

[24] J. Kools. (2013) 6 functions for generating artificial datasets. https://www.mathworks.com/matlabcentral/

fileexchange/41459-6-functions-for-generating-artificial-datasets. [Online; accessed 22-September-2020].

[25] H. A. Abu Alfeilat, A. B. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, H. S. Eyal Salman, and V. S. Prasath, "Effects of distance measure choice on k-nearest neighbor classifier performance: a review," *Big data*, vol. 7, no. 4, pp. 221–248, 2019. [Online]. Available: https://doi.org/10.1007/s10994-008-5059-5

[26] G. Rätsch. Benchmark data sets. [Online]. Available: http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm

[27] T. Diethe, "13 benchmark datasets derived from the UCI, DELVE and STATLOG repositories," https://github.com/tdiethe/gunnar_raetsch_benchmark_datasets/, 2015.

[28] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004. [Online]. Available: https://doi.org/10.1017/CBO9780511804441

[29] A. S. Shirkhorshidi, S. Aghabozorgi, and T. Y. Wah, "A comparison study on similarity and dissimilarity measures in clustering continuous data," *PloS one*, vol. 10, no. 12, p. e0144059, 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0144059

[30] V. Prasath, H. A. A. Alfeilat, A. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, and H. S. E. Salman, "Distance and similarity measures effect on the performance of k-nearest neighbor classifier–a review," *arXiv preprint arXiv:1708.04321*, 2017. [Online]. Available: https://doi.org/10.1089/big.2018.0175

[31] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *Journal of artificial intelligence research*, vol. 6, pp. 1–34, 1997. [Online]. Available: https://arxiv.org/pdf/cs/9701101.pdf

[32] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *International conference on database theory*. Springer, 2001, pp. 420–434. [Online]. Available: https://doi.org/10.1007/3-540-44503-x_27