ISSN: 2320-1363

Advantages of using Laplacian Kernel over RBF in a Support Vector Machine

P. Chandrasekhar*[1], P.Md. Akhthar*[2]
[1] *Research Scholar, Dept. of Computer Science & Technology, S.K. University, A.P., INDIA*
[2] *Professor, Dept. of Statistics, S.K. University, Anantapur, A.P., INDIA*

Abstract

Support Vector Machines are popular because of their promising performance in classification and prediction. Experiments on a collection of benchmark data sets demonstrate the efficiency and effectiveness of the SVM algorithm. The success of SVM lies in suitable kernel design and selection of its parameters. There is no formal way to decide, which kernel function is suited to a class of classifier problem. While most commonly used kernels are Radial Basis Function (RBF), polynomial, spline, sigmoid etc, we have explored an un-conventional kernel function, namely the Laplacian kernel. Contrary to the usual thinking that Gaussian RBF is well suited for experimentation, we found that the Laplacian kernel behaves better than the conventional RBF for some data sets though not always. This will make calculations simpler and reduce the time required to implement the SVM problem.

Keywords **--- Support Vector Machine, hyperplane, non-linear classifier, Kernel trick**

I. INTRODUCTION

Support Vector learning is based on simple ideas which originated in statistical learning theory (Vapnik [1]). The simplicity comes from the fact that Support Vector Machines (SVMs) apply a simple linear method to the data but in a high-dimensional feature space which is non-linearly related to the input space. Moreover, even though we can think of SVMs as a linear algorithm in a high-dimensional space, in practice, it does not involve any computations in that higher dimensional space. This simplicity combined with state of the art performance on many learning problems (classification, regression, and novelty detection) has contributed to the popularity of the SVM. Although the training time of even the fastest SVMs can be extremely slow, they are highly accurate, owing to their ability to model complex nonlinear decision boundaries. They are much less prone to over fitting than other methods. The support vectors found also provide a compact description of the learned model.

ISSN: 2320-1363

SVMs can be used for prediction as well as classification. They have been applied to a number of areas, such as geo and environmental sciences, medical sciences and biological sciences. SVM domain relates to real world problems including handwritten digit recognition, object recognition, and speaker identification, as well as benchmark time-series prediction tests.

SVMs use an implicit mapping φ of the input data into a high-dimensional feature space defined by a kernel function, i.e., a function returning the inner product $\langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle$ between the images of two data points $\mathbf{x}$, $\mathbf{x}'$ in the feature space. The learning then takes place in the feature space. This is often referred to as the "kernel trick" (Sch¨olkopf and Smola [2]). More precisely, if a projection $\varphi: X \rightarrow H$ is used, the dot product $\langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle$ can be represented by a kernel function k given by $k(\mathbf{x}, \mathbf{x}') = \langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle$ which is computationally simpler than explicitly projecting $\mathbf{x}$ and $\mathbf{x}'$ into the feature space H. One interesting property of support vector machines and other kernel-based systems is that, once a valid kernel function has been selected, one can practically work in spaces of any dimension without any significant additional computational cost, since feature mapping is never effectively performed. In fact, one does not even need to know which features are being used.

Another advantage of SVMs and kernel methods is that one can design and use a kernel for a particular problem that could be applied directly to the data without the need for a feature extraction process. This is particularly important in problems where a lot of structure of the data is lost by the feature extraction process (e.g., text processing).

The benefit of using a support vector machine to extract complex patterns from the data is that it is not necessary to have a prior understanding of the behavior of the data. A support vector machine is able to analyze the data and extract its own insights and relationships.

One of the best features of support vector machines is that they are able to deal with errors and noise in the data very well. They are often able to see the underlying pattern within the data and filter out data outliers and other complexities.

The success of classification of data using a support vector machine depends largely on the choice of the kernel. When training an SVM the practitioner needs to make a number of decisions on how to process the data, what kernel to use and

ISSN: 2320-1363

setting the parameters of the SVM and the kernel.The polynomial kernels and the Gaussian RBF are the most popular kernels and they are widely used in the literature to classify data and often a comparision of the results using various kernels is made showing the effectiveness and suitability of a kernel to a given data set (Srivastava and Bhambhu [3], Prajapati and Patle[4], Sangeetha and Kalpana [5], Ben Ayed Mezghani, Zribi Boujelbene, and Ellouze [6]). Efforts are often made in literature to device or construct new kernels to suit specific large data sets (Boolchandani and Vineet Sahula [7], Ayat *et al.*,[8]). Such attempts are made in literature when the well known kernels are unable to classify a specific dataset of a real world problem efficiently.

Before describing our problem, it is necessary to describe some properties of the Gaussian RBF kernel. Normally a Gaussian will be used as the RBF kernel. The output of the kernel is dependent on the Euclidean distance of $\mathbf{x}$ from $\mathbf{x}'$ (one of these will be the support vector and the other will be the testing data point). The support vector will be the centre of the RBF and γ will determine the area of influence this support vector has over the data space. The Gaussian RBF kernel is $k(\mathbf{x},\mathbf{x}') = \exp(-\gamma\|\mathbf{x}-\mathbf{x}'\|^2)$. A smaller value of γ will give a smoother decision surface and more regular decision boundary. This is because an RBF with small γ will allow a support vector to have a strong influence over a larger area. This property is less pronounced in the case of the Laplacian Kernel $k(\mathbf{x},\mathbf{x}') = \exp(-\gamma\|x-x'\|)$. Intuitively, the *gamma* parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. Similar results are true for the Laplacian kernel also though the area of influence differs.

II. Related Work

The purpose of this paper is to introduce the Laplacian kernel which is also known as the Exponential kernel. In the Gaussian kernel the square of the distance between the vectors $\mathbf{x}$ and $\mathbf{x}'$ is considered in the exponential whereas here we take the distance alone. This kernel is also well known kernel but least attention is paid in using this kernel to train data in a Support Vector machine. The kernel is mathematically simpler and when used in a SVM it can reduce the complexity of algorithm. It is the purpose of this paper to show that the exponential kernel produces

ISSN: 2320-1363

very good classification accuracy and indeed it will serve better purpose than the well known Gaussian kernel for certain data sets. The secret behind this success is that the Gaussian kernel very slowly decreases from 1 whereas the exponential kernel is faster. Similarly, the Gaussian kernel falls to zero very fast whereas the exponential kernel is slower in reaching zero compared to the Gaussian kernel. This avoids clustering of points at 1 and 0 and disperses points wider enabling smooth separation in the feature space. Also much depends on parameterγ. Motivated by these ideas we consider the problem of classification of data sets using a Support Vector Machine. We use the exponential kernel. The data sets considered are described during the discussion. Now we present the quadratic programming problem in brief.

III. SVM CLASSIFIERS (Methodology)

A. Linear Classifiers

Support vector machines are an exa mple of a linear two-class classifier. The data for two class learning problem consists of objects labeled with one of two labels corresponnding to the two classes; for convenience we assume the labels are +1 (positive examples) or -1 (negative examples). In what follows boldface $\mathbf{x}$ denotes a vector with components x_i. The notation $\mathbf{x}_i$ will denote the i th vector in a data set $\{(\mathbf{x}_i, y_i); i = 1,2, \ldots, n\}$ where y_i is the label associated with $\mathbf{x}_i$ The objects $\mathbf{x}_i$ are called patterns or examples. We assume that the examples belong to some set X. Initially we assume that the examples are vectors, but once we introduce kernels, this assumption will be relaxed at which point they could be any continuous or discrete objects.

A key concept required for defining a linear classifier is the dot product betwe en two vectors, also referred to as an inner product or scalar product defined as $\mathbf{w}^{\mathrm{T}}\mathbf{x} = \sum_i^n w_i x_i$. A linear classifier is based on a linear discriminant function of the form

$$f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b. \quad (1)$$

The vector $\mathbf{w}$ is known as the weight vector, and b is called the bias. Consider the case $b = 0$ first. The set of points $\mathbf{x}$ such that $\mathbf{w}^T\mathbf{x} = 0$ are all points that are perpendicular to $\mathbf{w}$ and go through the origin – a line in two dimensions, a plane in three dimensions, and more generally a hyperplane. The bias b translates the hyperplane away from the origin. The hyperplane

ISSN: 2320-1363

$$\{\mathbf{x} : f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b\} = 0$$

(2)

divides the space into two.

The sign of the discriminant function $f(\mathbf{x})$ in equation (1) denotes the side of the hyperplane a point is on. The boundary between regions classified as positive and negative is called the decision boundary of the classifier. The decision boundary defined by a hyperplane is said to be linear because it is linear in the input (c.f. Equation 1). A classifier with a linear decision boundary is called a linear classifier. Conversely, when the decisioin boundary of a classifier depends on the data in a non-linear way the classifier is said to be non-linear.

B. From linear to non-linear classifiers

The machinery of linear classifiers can be extended to generate non-linear decision boundaries. The naïve way of making a non-linear classifier out of linear classifier is to map our data from the input space X to a feature space F using a non-linear function $\varphi: \mathrm{X} \rightarrow \mathrm{F}$ .In the feature space F the discriminant function is

$$f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}) + b$$

(3)

Kernel methods avoid the step of explicitly mapping the data to a higher dimensional feature space. Suppose the weight vector can be expressed as a linear combination of the training examples, that is, $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i$. Then, $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i^{\mathrm{T}}\mathbf{x} + b$. In the feature space F, this expression takes the form $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \varphi(\mathbf{x}_i)^{\mathrm{T}}\varphi(\mathbf{x}) + b$. The representation in terms of the variables α_i is known as the dual representation of the decision boundary. As indicated above, the feature space F may be high dimensional, making this trick impractical unless the kernel function $k(\mathbf{x}, \mathbf{x}')$ is defined as $k(\mathbf{x}, \mathbf{x}') = \varphi(\mathbf{x})^{\mathrm{T}}\varphi(\mathbf{x}')$ - a dot product – that can be computed efficiently. In terms of the kernel function, the discriminant function is $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}, \mathbf{x}') + b$

We saw that a linear decision boundary can be "kernalized", i.e., its dependence on the data is only through dot product. In order for this to be useful, the training algorithms need to be kernalized as well. It turns out that a large number of machine learning algorithms can be expressed using kernels – including ridge regression, the perceptron algorithm and SVMs [2,9].

We use the term linearly seperable to denote data for which there exists a linear decision bounday that seperates positive from negative examples. Initially

ISSN: 2320-1363

we assume linearly seperable data and later indicate how to handle data which is not linearly seperable.

We define the notion of a margin. For a given hyperplane, we denote by $\mathbf{x}_+$ and $\mathbf{x}_-$ the closest points to the hyperplane among the positive and negative example points respectively. The norm of the vector $\mathbf{w}$ denoted by $\|\mathbf{w}\|$ is its length and is given by $\sqrt{\mathbf{w}^T\mathbf{w}}$. Let H be the optimal hyperplane – for which the margin is maximum. Let $\mathbf{H}_+$ and $\mathbf{H}_-$ be the two hyperplanes equidistant on either side defining the margin. We may call them as "sides" defining the margin. Then $\mathbf{x}_+$ and $\mathbf{x}_-$ lie on $\mathbf{H}_+$ and $\mathbf{H}_-$ and hence we have $\mathbf{w}\mathbf{x}_+ + b = +1$ and $\mathbf{w}\mathbf{x}_- + b = -1$. The points on the planes $\mathbf{H}_+$ and $\mathbf{H}_-$ are support vectors. From geometric considerations, the distance between the planes $\mathbf{H}_+$ and $\mathbf{H}_-$ is given by $\frac{|\mathbf{w}\mathbf{x}+b|}{\|\mathbf{w}\|}\| = 1/\|\mathbf{w}\|$. Therefore the distance between $\mathbf{H}_+$ and $\mathbf{H}_-$ is $2/\|\mathbf{w}\|$

We have the concept of a margin and now we can formulate the maximum margin classifier. We will first define the hard margin SVM, applicable to a linearly seperable data set, and then modify it to handle non-seperable data. The maximum classifier is the discriminant function that maximizes the geometric margin $1/\|\mathbf{w}\|$ which is equivalent to minimizing $\frac{1}{2}\|\mathbf{w}\|^2$. This leads to the following constrained optimization problem.

$\text{Minimize}_{\mathbf{w},b}\frac{1}{2}\|\mathbf{w}\|^2$ subject to:

$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \ i = 1, \ldots, n.$

(4)

The constraints in this formulation ensure that the maximum margin classifier classifies each example correctly, which is possible since we assumed that the data is linearly seperable. In practice, data is often not linearly seperable; and even if it is, a greater margin can be achieved by allowing the classifier to misclassify some points. To allow errors we replace the inequality constraintes in Eq. (4) with $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \ \ i = 1, \ldots, n,$ where $\xi_i \geq 0$ are slack variables that allow an example to be in the margin ($0 \leq \xi_i \leq 1$, also called a margin error) or to be misclassified ($\xi_i \geq 1$). Since an example is misclassified if the value of its slack variable is greater than 1, $\sum_i \xi_i$ is a bound on the number misclassified examples. Our object of maximizing the margin *i.e.*, minimizing $\frac{1}{2}\|\mathbf{w}\|^2$ will be augmented with a term $C\sum_i \xi_i$ to penalize misclassification and

ISSN: 2320-1363

margin errors. The optimization problem now becomes

$$\text{Minimize}_{w,b}\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{subject to}$$

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i,\ \ \xi_i \geq 0.$$

(5)

The constant $C>0$ sets the relative importance of maximizing the margin and minimizing the amount of slack. This formulation is called the soft margin SVM, and was introduced by Cortes and Vapnik (10). Using the method of Lagrange multipliers, we can obtain the dual formulation which is expressed in terms of variables α_i[10, 2, 9].

$$\underset{\alpha}{\text{Maximize}} \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$

Subject to $\sum_{i=1}^{n} y_i \alpha_i = 0,\ 0 \leq \alpha_i \leq C$

(6)

The dual formulation leads to an expansion of the weight vector in terms of the input examples: $\mathbf{w} = \sum_{i=1}^{n} y_i \alpha_i \mathbf{x}_i$. The examples $\mathbf{x}_i$ for which $\alpha_i \geq 0$ are those points that are on the margin, or within the margin when a soft margin SVM is used. These are the so called support vectors. The expansion in terms of the support vectors is often sparse, and the level of sparsity (fraction of the data serving as support vectors) is an upper bound on the error rate of the classifier[2].

The dual formulation of the SVM optimization problem depends on the data only through the dot products. The dot product can therefore be replaced with a nonlinear kernel function, thereby performing large margin seperation in the feature space of the kernel. The SVM optimization problem was traditionally solved in the dual formulation, and only recently it was shown that the primal formulation, Equation (5) can lead to efficient kernel-based learning [11].

The ability of a machine to learn will depend on the mapping function or in other words on the kernel function. So, the choice of kernel function and its parameters will play an important role in the generalization ability of a machine. The other important component of SVM is the support vectors. On these vectors, the placement of separating hyperplane depends upon. It is preferable that the number of support vectors should be less to enable misclassification.

IV. RESULTS AND DISCUSSION

Support Vector Machines require parameter tuning in order to improve the

ISSN: 2320-1363

Data set	IRISH		HEART		DIBETIC	
γ Value	RBF	LAPLACE	RBF	LAPLACE	RBF	LAPLACE
0.001	96.6667	68.6667	65.5556	55.5556	72.7865	75.1592
0.005	92.6667	68.6667	60.3704	55.5556	69.2708	74.9469
0.01	94	68.6667	55.5556	55.5556	65.8854	75.1592
0.05	95.3333	98.6667	55.5556	58.5185	65.1042	75.3715
0.1	96	78	60.3212	55.5556	65.1042	75.1592
0.5	96.6667	98.6667	55.5556	58.5185	65.1042	73.4607
1.0	94	99.3333	55.5556	75.9259	65.1042	81.8182
5.0	95.333	100	55.5444	83.3333	65.1066	80.3922
10	92.6667	100	55.5556	83.7037	65.1042	75.7962
20	85.3333	100	55.5556	84.0741	65.1042	75.7962
100	56.667	100	55.5556	83.7037	65.1042	75.7962

accuracy of their test results. To do this parameter tuning, 5-fold cross validation was performed for each of kernels in SVMLib's Matlab implementation on the training data. A logarithmic"grid search" method was performed to find the best choices for C and γ ; those that had the best average cross-validation accuracy are the ones chosen for use on the test data.

We will use the following data sets originating from the UCI machine learning database. The experiments were done using three data sets. We have selected the data sets ranging from smaller number of instances to larger numbers. Binary SVM has been used in the experiments. The three data sets used were irish, heart, and diabetes. Irish data set has 148 samples and each sample has 14 attributes. Heart data set has 270 instances with 14 attributes. The diabetic data set has 768 samples each having 9 attributes.

Table 1 gives the results obtained for the cases of the three data sets for various values of the parameter γ which appears in the kernels.

The results show the classification accuracies. It is already remarked that the exponential kernel would reduce the clustering of the points at the points 0 and 1. Thus we expect a quick decrease in the neighborhood of zero and a moderate decrease toward infinity. This forces the

ISSN: 2320-1363

images of the original points to be linearly separable in the augmented space. The parameter γ varies over a wide range from very small values to large values (see table) showing the behavior of the kernel functions in effectively tackling the separation of points in the feature space.

In the case of heart data base we have 270 instance and 14 attributes, we can find that higher values of γ are able to accomplish the job of classification very well. We find that the classification efficiency will go to 83 or 84 for larger values of γ whereas for smaller values of γ both the kernels produce similar values of round 55 to 60. The diabetic data set consists of 768 examples and 9 attributes. The exponential kernel uniformly gives better results at all values of the parameter γ. The best accuracy is reached when γ =1. The Irish data set has 148 instances and 14 attributes. Both the kernels give very good results but the Exponential kernel is highly successful scoring an accuracy of 100 for larger values of γ.

V. CONCLUSIONS

While dealing with training data on a support vector machine, we felt that the sparseness of the data inside the original space can vary widely depending on its distribution and we believe that kernels preserving the whole data closeness information while still penalizing the far neighborhood are more reliable, especially in case of sparse data. And simpler kernels can not the underestimated to do this job. We proposed and experimented with a non conventional kernel, namely the Laplace kernel that allows such a behavior by ensuring at once a quick decreasing from zero and a slow decrease toward infinity (compared to RBF). This tends to uncorrelated (disperse) as much as possible very close points into the augmented space. Hybrid kernels can be created using systematic methodology and optimization technique only after ensuring that the conventional and simpler kernels do not work on a data set.

REFERENCES

1. V. Vapnik, Statistical Learning Theory. Wiley, New York, 1998.

2. B.Sch¨olkopf, and A. Smola, Learning with Kernels. MIT Press, Cambridge, 2002.

3. Durgesh K. Srivastava and Lekha Bhambhu, “Data classification using support vector machine,” Journal of Theoretical and Applied Information Technology, JATIT, pp.1-6. 2009.

4. Gend Lal Prajapati, and Arti Patle, “On Performing Classification Using SVM with Radial Basis and Polynomial Kernel Functions,” in Third International Conference on

ISSN: 2320-1363

Emerging Trends in Engineering and Technology, ICETET, 2010, pp. 512-515.

5. R. Sangeetha, and B. Kalpana, "Performation Evaluation of kernels in Multiclass Support Vector Machines," International Journal of Soft Computing and Engineering, IJSCE, Vol.1, Issue 5, pp 138-145. Nov 2011.

6. D. Ben Ayed Mezghani, S. Zribi Boujelbene, and N. Ellouze, "Evaluation of SVM Kernels and Conventional Machine Learning Algorithms for Speaker Identification," International Journal of Hybrid Information Technology, Vol3, No.3, pp. 23-33. July 2010.

7. D. Boolchandani, and Vineet Sahula, "Exploring Efficient Kernel Functions for Support Vector Machine Based Feasibility Models for Analog Circuits," International Journal of Design, Analysis and Tools for Circuits and Systems, Vol 1, No. 1, pp. 1-8. 2011.

8. N.E. Ayat, M. Cheriet, L. Remaki, and C.Y. Suen, " KMOD-A New Support Vector Machine Kernel with Moderate Decreasing for Pattern Recognition. Application to Digit Image Recognition," in Proceedings of the Sixth International Conference on Document Analysis and Recognition, Sept. 2011, pp. 1215-1219.

9. N. Cristianini and J. Shawe-Taylor. An Introduction to Support Vector Machines. Cambridge University Press, Cambridge, UK, 2000.

10. C. Cotes and V. Vapnik. Sapport Vector Networks. Machine Learning, 20:273-297, 1995.

11. O. Chapelle. Training a support vector machine in the primal. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, Editors, Large Scale Kernel Machines. MIT Press, Cambridge, MA., 2007.