# Using K-NN SVMs for Performance Improvement and Comparison to K-Highest Lagrange Multipliers Selection

Sedat Ozer[1], Chi Hau Chen[2], and Imam Samil Yetik[3]

[1] Electrical & Computer Eng. Dept, Rutgers University, New Brunswick, NJ, USA
sozer@umassd.edu
[2] Electrical & Computer Eng. Dept, University of Massachusetts,
Dartmouth, N. Dartmouth, MA, USA
cchen@umassd.edu
[3] Electrical & Computer Eng. Dept, Illinois Institute of Technology, Chicago, IL, USA
yetik@iit.edu

**Abstract.** Support Vector Machines (SVM) can perform very well on noise free data sets and can usually achieve good classification accuracies when the data is noisy. However, because of the overfitting problem, the accuracy decreases if the SVM is modeled improperly or if the data is excessively noisy or nonlinear. For SVM, most of the misclassification occurs when the test data lies closer to the decision boundary. Therefore in this paper, we investigate the effect of Support Vectors found by SVM, and their effect on the decision when used with the Gaussian kernel. Based on the discussion results we also propose a new technique to improve the performance of SVM by creating smaller clusters along the decision boundary in the higher dimensional space. In this way we reduce the overfitting problem that occurs because of the model selection or the noise effect. As an alternative SVM tuning method, we also propose using only K highest Lagrange multipliers to summarize the decision boundary instead of the whole support vectors and compare the performances. Thus with test results, we show that the number of Support Vectors can be decreased further by using only a fraction of the support vectors found at the training step as a post-processing method.

**Keywords:** Support Vector Machine, KNN SVM, Post-processing, Support Vector Reduction.

## 1 Introduction

Support Vector Machine (SVM) is a well known learning algorithm that has been widely used in many applications including classification, estimation and tracking as in [1], [2], [3] and [4]. SVM finds the closest data vectors called support vectors (SV), to the decision boundary in the training set and it classifies a given new test vector by using only these closest data vectors [5],[6].

In order to find the optimal nonlinear decision boundary, SVM uses kernel functions, along the optimization step to find the optimal hyperparameters, [5]. However, in practice, the iterative techniques used at the optimization step, can also affect the classification accuracy of SVM within the margin.

E.R. Hancock et al. (Eds.): SSPR & SPR 2010, LNCS 6218, pp. 532–539, 2010.

Besides the SVM algorithm, the K nearest neighbor (KNN) technique is another well known learning technique and being used in several pattern recognition applications as in [7]. There have been some previous studies where KNN technique was combined with SVM as in [8], [9] and [10].

The combination of these two techniques by switching between them could perform better only for certain cases in which the new data is close to the decision boundary. In [8], the KNN algorithm is applied directly onto those data vectors which are within the margin. However, in [8], it is claimed that previously proposed KSVM cannot reduce the generalization error.

Also, in studies such as in [9] and in [10], KNN idea is used in a different way combined with SVs. In [9], authors study the effects of using K nearest SVs by focusing on query time rather than improving the accuracy. They propose using a varying K value for each test data till they reach to a certain threshold. Thus they search for an appropriate K value for each given test data. In [10], instead of training the SVM only once, the authors propose using the K nearest data values to train SVM separately for each given test data.

Both of the papers [9] and [10] uses the KNN idea in a different way, while [9] requires to search for an appropriate K value for each single test data, the authors of [10] require to train SVM for each given new test data. Moreover, although these papers do not clearly indicate in them, they can perform better when used with Gaussian kernel because of the Gaussian kernel function's shape.

In this study, we propose more naive yet efficient way of using KNN SVs when used with Gaussian kernels for a given dataset. We train the SVM only once and after that we require only one K value to be found. Our approach is applicable to all new data points regardless of their distance to the decision boundary. In this approach, we use the entire training data to find the SVs. However after this point, instead of using the all SVs that have been found on the training step, we propose to use only the K nearest SVs. Since the Gaussian kernel is also using the Euclidian distance, there is not much computational cost to find distances to each SVs.

The classification with SVM, besides its high accuracy, also provides sparseness which is another advantage of SVM, thus we do not need to save all the training data. Therefore, in this study, we also propose using only the K highest Lagrange multipliers ($\alpha$) instead of all the nonzero Lagrange multipliers found at the training step of SVM. Section 4 tests and investigates if all the SVs found by the classifier, are necessary to classify the new data. Experimental results show that, even if the non-zero $\alpha$ values has closer value to each other, there can be some redundancy where we can reduce the SV number by choosing only the K highest SVs and corresponding $\alpha$ values.

Consequently, the SV number can be reduced by using the method presented on this paper for a similar performance. Besides, we also show that it is possible to increase the efficiency of the SVM by using only a fraction of SV numbers. Preliminary test results provide us interesting results about SVs which we discuss at the Section 5.

## 2 Support Vector Machine

SVM searches for the optimal decision boundary between two classes [5], [6]. Although SVM is mainly designed as a linear binary classifier, it is widely being used for nonlinear data efficiently as well, by the use of kernel functions [5].

SVM uses the following formula for the classification, for a given new data vector **x**:

$$f(\mathbf{x}) = \text{sgn}\left( \sum_{i=1}^{m} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b \right) \tag{1}$$

where $\alpha$ is the Lagrange multiplier for each SV that needs to be found in the training step, $m$ the support vector number, $b$ the biasing term, $y$ the class labels, $K(\mathbf{x},\mathbf{x_i})$ the kernel function, and $\mathbf{x_i}$ are the support vectors. The parameters $b$ and $\alpha_i$ need to be found in the training step. The Lagrange multipliers, $\alpha_i$, can be found by maximizing the following equation:

$$w(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x_i}, \mathbf{x_j}) \tag{2}$$

subject to $\sum_{i=1}^{n} \alpha_i y_i = 0$ and $\alpha_i \geq 0$, where $n$ is the training sample number.

Thus the $\mathbf{x_i}$ input vectors with nonzero $\alpha_i$ values, are called support vectors (SV). Although several kernel functions have been proposed to be used with SVMs, as in [5], [11] and [12], the kernel function used in this study is the Gaussian kernel which is defined as:

$$K(\mathbf{x}, \mathbf{y}) = \exp\left( -\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2} \right) \tag{3}$$

where $\sigma$ is the kernel parameter that needs to be found for a satisfactory classification performance.

## 3 The Proposed Method

K-NN SVM: If the Gaussian kernel is being used, then SVM can be considered as a binary clustering algorithm. However, in contrast to the other clustering algorithms, instead of finding the centroids of the clusters, SVM uses the edge information of the clusters where the two clusters are the closest to each other.

Our assumption in this study is that for a given new data vector, we do not need to use all the support vectors as in the traditional SVM. That is because the hyperplane can be more linear in some regions of the whole data space, and can be highly nonlinear in other regions. Therefore using only the K nearest support vectors within the same local region can increase the performance. Let us re-arrange the equation (1) as follows:

$$f(\mathbf{x}) = \text{sgn}\left( b + \sum_{i=1}^{h} \alpha_i K(\mathbf{x}, \mathbf{x}_i) - \sum_{j=1}^{g} \alpha_j K(\mathbf{x}, \mathbf{x}_j) \right) \tag{4}$$

where $h$ is the number of support vectors for the (+1) zone and similarly $g$ is the number of the support vectors for the (-1) zone.

When the Gaussian kernel is being used for SVM, the Equation (4) simply becomes a weighted subtraction of $\alpha$ values with a biasing term b, treating the $K(\mathbf{x},\mathbf{x_i})$ values as weights. Here the weights $K(\mathbf{x},\mathbf{x_i})$ are mapped to a value based on the Euclidian distance between the new data and the support vector.

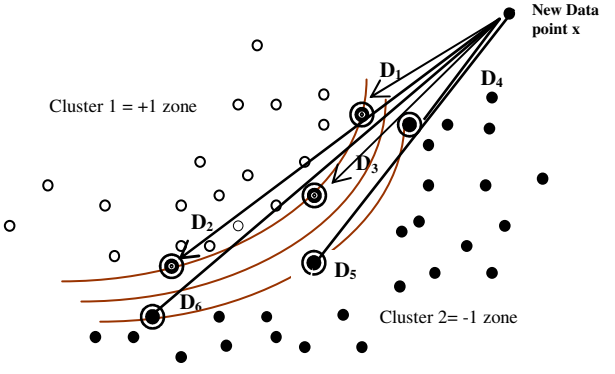


**Fig. 1.** Distances to all Support Vectors for a given new data

As shown in Equation (5), the Gaussian kernel maps the distance values between 0 and 1, where the closer distance is mapped to a higher value. Here, the kernel parameter $\sigma$ decides after which value the mapping decays to 0 more faster. Thus, for a given test data vector, some of the $\alpha$ values in Equation (4) can vanish because of their weights goes to zero, then only the $b$ value and the closest $\alpha$ values decide for the sign of the new test data. That means all SVs have some local effect on the whole decision boundary.

$$0 < K(\mathbf{x},\mathbf{x}_i) \leq 1 \tag{5}$$

As illustrated in Figure 1, for a given new data the distances to all support vectors are shown as $D_1$, $D_2$, $D_3$, $D_4$, $D_5$, $D_6$. For the classification of the new data point, $D_2$ and $D_3$ will be more effective than $D_5$ and $D_6$, as these distance values are smaller. This may yield an incorrect classification of the data. This situation can be more important as the test data gets closer to the decision boundary.

This can reduce the effect of the noise on forming the decision boundary. As the overfitting problem yields a complicated nonlinear decision surface, and usually requires more support vectors.

As a result, the classifier for a given new test data can be constructed as:

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{k} \alpha_i y_i \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}\right) + b\right) \tag{6}$$

where $k$ is the nearest support vector number and $k \leq m$ for an improved accuracy of SVM.

## 4 Experimental Results

In this section we perform experiments to illustrate the performance of the proposed method.

For the experiments, we use the image segmentation dataset which has 7 different classes of certain images. Each instant is a 3 by 3 region and randomly chosen from a database of 7 outdoor images. Each image is hand segmented for classification purpose. The dataset is available at [13].

We use one against all rule for each 7 classes. The first 210 data are used as training dataset and the remaining 2100 data are used for testing. Each vector has 18 features. We calculate the highest performance values by finding the appropriate Gaussian kernel parameters. Before the training the SVM, we first normalized all the data between the range [-1,1]. For each class, we first find the best kernel parameter that gives the lowest generalization error, and then by using this kernel parameter, we find the support vectors and corresponding Lagrange multipliers. For the experiments, we used and modified the code available at [14].

Table 1 shows best classification results for the test data with the corresponding Gaussian kernel parameters and support vector numbers for each class. Then by keeping the same support vectors and the corresponding α values, we applied $K$ nearest SVM technique on the same dataset and the results are shown on Table 2. The best classification percentages are obtained by using the lowest $K$ nearest support vectors, and are shown on Table 2 where $K$ is the nearest Support vector numbers.

On Table 3 we first sorted the $\alpha$ values in descending order and then have chosen the $K$ highest α values with the corresponding support vectors. The remaining $\alpha$ values are set to zero. Therefore the number of SVs is reduced in each test.

**Table 1.** The SVM training and best kernel parameters with SV numbers for the best classification results

*Traditional SVM Test Results*

| Class name: | cement | brickface | Grass | foliage | sky | path | window |
|---|---|---|---|---|---|---|---|
| Best %: | 96.95 | 99.48 | 99.86 | 96.71 | 100 | 99.71 | 94.57 |
| Parameter: | 0.53 | 0.44 | 0.5 | 1.43 | 1.45 | 0.43 | 0.40 |
| SV Number | 84 | 93 | 94 | 28 | 18 | 106 | 122 |

**Table 2.** K nearest SVM classification results for the image segmentation dataset

*K Nearest SVM Test Results*

| Class name: | cement | brickface | Grass | foliage | Sky | Path | Window |
|---|---|---|---|---|---|---|---|
| Best %: | 97 | 99.52 | 99.86 | 96.71 | 100 | 99.95 | 94.62 |
| σ value: | 0.53 | 0.44 | 0.5 | 1.43 | 1.45 | 0.43 | 0.4 |
| $K$ | 37 | 29 | 7 | 28 | 3 | 9 | 25 |

**Table 3.** Using only the highest K number of α values and its results for different classes

*Using only the K highest α values for SVM*

| Class name: | cement | brickface | Grass | foliage | Sky | Path | Window |
|---|---|---|---|---|---|---|---|
| Best %: | 97 | 99.52 | 99.86 | 96.71 | 100 | 99.86 | 94.57 |
| $\sigma$ value: | 0.53 | 0.44 | 0.5 | 1.43 | 1.45 | 0.43 | 0.4 |
| SV Number | 41 | 23 | 8 | 28 | 3 | 23 | 51 |

**Table 4.** Showing the maximum and minimum α values that are used and discarded in the "K highest α values for SVM" experiment

*The Maximum and Minimum α values used in Table 3*

| Class name: | cement | brickface | Grass | foliage | Sky | Path | Window |
|---|---|---|---|---|---|---|---|
| Used max $\alpha$ | 12.16 | 15.31 | 1.42 | 464.3 | 1.27 | 1.61 | 14.02 |
| Used min $\alpha$ | 0.36 | 0.38 | 0.57 | 1.13 | 1.17 | 0.28 | 0.21 |
| Nonused max | 0.34 | 0.21 | 0.38 | 0 | 0.99 | 0.28 | 0.21 |
| Nonused min | 0.002 | 0.001 | 0.001 | 0 | 0.02 | 0 | 0.004 |

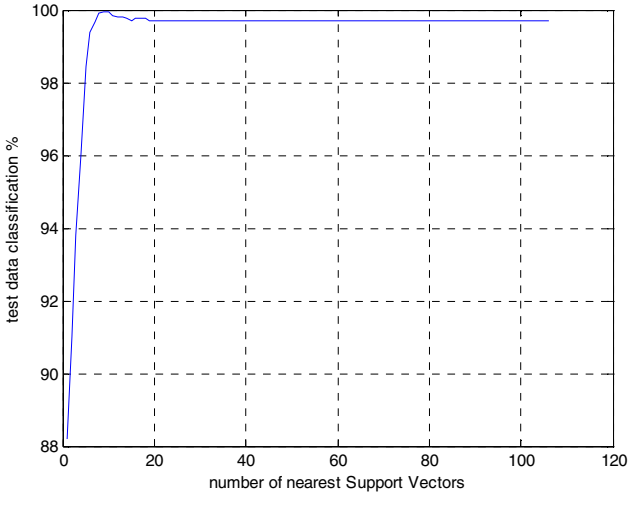


**Fig. 2.** For the Path class the K nearest SV number vs test classification percentage plot

The classification percentage with the same kernel parameters are shown on Table 3 for each class separately with the best $K$ values. The maximum and minimum $\alpha$ values that are used and discarded for each class are shown on Table 4.

In Figure 2, we plotted the change on classification percentage versus the nearest support vector numbers used in Equation (7) when the kernel parameter is kept as 0.43. It can clearly be seen that the best classification result is not obtained by using all the support vectors. The peak value for the plot is obtained when the K is chosen as 9 as it is shown on the plot.

Comparing Table 1, Table 2 and Table 3, we can see that K nearest SVM gives the best results for Path and Window classes when the same kernel parameters are used. Cement and Brickface classes show the same improved performance on Table 2 and Table 3. For Foliage, Grass and Sky classes we find the same results as in the regular SVM case. However for the Grass and Sky classes the same percentage values are obtained by using lower support vector numbers on experiments.

## 5 Conclusion and Discussion

In this paper, as an alternative SVM tuning method, we propose using the KNN idea to decrease generalization error, when the optimum kernel parameter is used with the Gaussian kernel. Moreover, we also show that the SV number can be reduced gradually by using only the highest K number of $\alpha$ values for the same or an increased performance for many applications.

Based on the experimental results on Table 1, and Table 2 we can conclude that, on SVM generalization, learning with the lowest Support vector numbers is not always the best way of learning the training data when the accuracy is the main concern. Although SVM is called a "sparse learning algorithm", it is better to keep sparseness at an optimum value (which is not the minimum value always) so that, it does not reduce the generalization ability of the SVM. Especially for highly nonlinear data structures, it is safer and better to learn with more support vectors. And then by using K nearest SVM technique, the generalization error can be decreased.

As shown on Table 1 and Table 2, the preliminary experimental results indicate us that, if the training step is completed with a small number of support vectors, then the generalization error may not be decreased with K nearest SVM as the support vectors are not close enough to form a proper smaller clusters to smoothen the decision boundary, thus we may not capture the nonlinearity of the space in a better way by using less support vectors.

Table 3 shows that, there may be some redundancy on support vector number which can be further reduced for SVM classification with the Gaussian kernel. Even the $\alpha$ values may have similar values (not closer to zero), by choosing only the K highest $\alpha$ values, and setting all the remaining ones to zero, we can obtain the same generalization performance. Finding this K value is an important step and it can be found heuristically. This result can be quite useful where the SV number is more important such as in feature selection and feature extraction applications. Less support vector also means less computation time for a given test data.

The information we obtain from this study when combined with previous similar works, shows us that there are interesting properties with the Gaussian kernel, and

there is a relation between the decision boundary and the kernel parameter as well as the K value. We will use these preliminary results in our next study to obtain a novel method that finds its own parameters automatically during the training step.

## References


[1] El-Naqa, I., Yang, Y., Wernick, M.N., Galatsanos, N.P., Nishikawa, R.M.: A support vector machine approach for detection of microcalcifications. IEEE Trans. on Medical Imaging 21(12), 1552–1563 (2002)

[2] Artan, Y., Huang, X.: Combining multiple 2*v*-SVM classifiers for tissue segmentation. In: Proc. of ISBI 2008, pp. 488–491 (2008)

[3] Lucey, S.: Enforcing non-positive weights for stable support vector tracking. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2008, pp. 1–8 (2008)

[4] Ozer, S., Haider, M.A., Langer, D.L., van der Kwast, T.H., Evans, A.J., Wernick, M.N., Trachtenberg, J., Yetik, I.S.: Prostate Cancer Localization with Multispectral MRI Based on Relevance Vector Machines. In: ISBI 2009, pp. 73–76 (2009)

[5] Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons, Chichester (1998) ISBN: 0-471-03003-1

[6] Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006)

[7] Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13(1), 21–27 (1967)

[8] Ming, T., Yi, Z., Songcan, C.: Improving support vector machine classifier by combining it with k nearest neighbor principle based on the best distance measurement. IEEE Intelligent Transportation Systems 1, 373–378 (2003)

[9] De Coste, D., Mazzoni, D.: Fast query-optimized kernel machine classification via incremental approximate nearest support vectors. In: 20th International Conference on Machine, Learning - ICML, Washington, DC (2003)

[10] Zhang, H., Berg, A.C., Maire, M., Malik, J.: Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In: IEEE Conference on Computer Vision and Pattern Recognition CVPR (2006)

[11] Zhang, L., Zhou, W., Jiao, L.: Wavelet Support Vector Machine. IEEE Trans. On Systems, Man, and Cybernetics-Part B: Cybernetics 34(1), 34–39 (2004)

[12] Ozer, S., Chen, C.H.: Generalized Chebyshev Kernels for Support Vector Classification. In: 19th International Conference on Pattern Recognition, ICPR (2008)

[13] Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[14] Canu, S., Grandvalet, Y., Guigue, V., Rakotomamonjy, A.: SVM and Kernel Methods Matlab Toolbox, Perception Systèmes et Information, INSA de Rouen (2005) `http://asi.insa-rouen.fr/enseignants/~arakotom/toolbox/index.html`