Preservation of Neighborhood Relation under Input to Feature Space Mapping in SVM Training

HyunJung Shin* and Sungzoon Cho
Dept. of Industrial Engineering, Seoul National University,
Shillim-dong, Kwanak-gu, Seoul, 151-744, Korea
Email: {hjshin72, zoon@snu.ac.kr}

ABSTRACT Training support vector machine (SVM) requires large memory and long time when a pattern set is large. To relieve the computational burden, we proposed neighborhood property based pattern selection algorithm (NPPS) which selects only the patterns near the decision boundary ahead of SVM training (Shin & Cho, 2002, 2003a-2003d). NPPS tries to identify those patterns that are likely to become support vectors in feature space. It was very effective: SVM training time was reduced by two orders of magnitude with almost no loss in accuracy for various datasets. It has to be noted, however, that decision boundary of SVM and support vectors are all defined in feature space while NPPS described above operates in input space. If neighborhood relation in input space is not preserved in feature space, NPPS may not always be effective. Since running NPPS in feature space is impractical, we show that the neighborhood relation is invariant under input to feature space mapping. The result now assures that the NPPS will identify those patterns near decision boundary in feature space.

KEYWORDS Support Vector Machine (SVM), Pattern Selection, Neighborhood Relation, Input to Feature Space Mapping

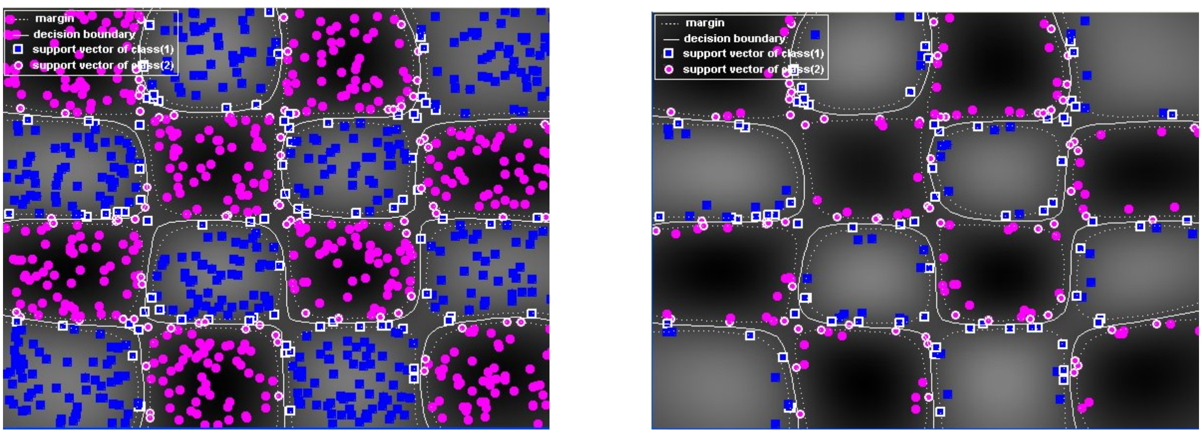
1. Introduction

In support vector machine (SVM) quadratic programming (QP) formulation, the dimension of kernel matrix ($M \times M$) is equal to the number of training patterns (M). A standard QP solver has time complexity of order $O(M^3)$: MINOS, CPLEX, LOQO, and MATLAB QP routines. And the solvers using decomposition methods approximately have time complexity of $T \cdot O(Mq + q^3)$ where T is the number of iterations and q is the size of the working set: Chunking, SMO, SVMlight, and SOR (Hearst, Schölkopf, Dumais, Osuna, & Platt, 1997; Platt, 1999). Needless to say, T increases as M increases. One way to circumvent this computational burden is to select some of training patterns in advance which contain most information given to learning. One of the merits of SVM theory distinguishable from other learning algorithms is that it is clear that which patterns are of importance to training. Those are called support vectors (SVs), distributed near the decision boundary, and fully and suc-

* Corresponding Author, hjshin72@snu.ac.kr

cinctly define the classification task at hand (Cauwenberghs & Poggio, 2001; Pontil & Verri, 1998; Vapnik, 1999). Furthermore, on the same training set, the SVMs trained with different kernel functions, i.e., RBF, polynomial, and tanh, have selected almost identical subset as support vectors (Schölkopf, Burges & Vapnik, 1995). Therefore, it is worth finding such *would-be* support vectors prior to SVM training.

Recently, we proposed *neighborhood property based pattern selection algorithm* (NPPS) (Shin & Cho, 2002, 2003a). The time complexity of NPPS is O(vM) where v is the number of patterns in the *overlap* region around decision boundary (Shin & Cho, 2003b). We utilized k nearest neighbors to look around the pattern's periphery. The *first* neighborhood property is that "a pattern located near the decision boundary tends to have more heterogeneous neighbors in their class-membership." The *second* neighborhood property dictates that "an overlap or a noisy pattern tends to belong to a different class from its neighbors." And the *third* neighborhood property is that "the neighbors of a pattern located near the decision boundary tend to be located near the decision boundary as well." The first one is used for identifying those patterns located near the decision boundary. The second one is used for removing the patterns located on the wrong side of the decision boundary. And the third one is used for skipping calculation of unnecessary distances between patterns, thus accelerating the pattern selection procedure. Fig. 1 visualizes one of the experimental results of artificial problems previously reported. The decision boundaries in both figures look quite similar, thus, generalization performance is similar. Table I summarizes the empirical results of NPPS reported in (Shin & Cho, 2002, 2003a-2003d). The table includes the results obtained from test with artificial datasets and real-world bench-marking datasets (http://www.ics.uci.edu/mlearn/, http://yann.lecun.com/exdb/mnist) as well as a marketing dataset (http://www.kernelmachines.org/). The results show that NPPS reduced SVM training time up to almost two orders of magnitude with virtually no loss of accuracy.



(a) SVM result with *all* patterns (b) SVM result with *selected* patterns

Fig. 1. 4x4 CHECKERBOARD PORBLEM: decision boundary is depicted as a solid line and the *margins* are defined by the dotted lines in both sides of it. Support vectors are outlined. Figure (a) indicates a typical SVM result of *all* patterns while (b) stands for that of *selected* patterns by NPPS.

In short, NPPS uses only local neighbor information to identify those patterns likely to be located near decision boundary. It has to be noted, however, that decision boundary of SVM and support vectors are all defined in feature space while NPPS described above operates in input space. Since the mapping from input space to feature space is highly nonlinear and dimension expanding, distortion of neighborhood relation could occur. In other words, neighborhood relation in input space may not be preserved in feature space. If that is the case, local information in input space may not be correct in feature space, thus impairing the effectiveness of NPPS. There are two approaches to solve this problem. The first involves running NPPS in feature space, and the second involves proving that the neighborhood relation is invariant under the input to feature space mapping. Let us consider the first approach. In order to compute the distance between two patterns, one has to have the optimal kernel function and hyper-parameters, which are usually found by trial-and-error involving multiple trials of SVM training with all patterns. Obviously, that is not acceptable since the purpose of pattern selection is to avoid training SVM with all patterns. On the other hand, NPPS can be executed only once in input space since it does not involve searching for optimal kernel and hyper-parameters. Thus, we take the second approach in this paper: to show that the neighborhood relation is invariant under the input to feature space mapping.

Table 1: EMPIRICAL RESULT COMPARISON: '*' stands for that the SVM training of corresponding problems was conducted with a standard QP solver, i.e., Gunn's SVM MATLAB Toolbox. On the contrary, because of heavy memory burden and lengthy training time caused by large training set, others were trained with an iterative SVM solver known as one of the fastest solvers, i.e., OSU SVM Classifier Toolbox (http://www.kernelmachines.org/). The column, 'SELECTED' of Execution Time includes SVM training time as well as NPPS running time.

	Num. Of Trn. Patterns		Num. of SVs		Execution Time (sec)		Test Error (%)	
	ALL	SELECTED	ALL	SELECTED	ALL	SELECTED	ALL	SELECTED
Continuous XOR *	600	179	167	84	454.83	4.06	9.67	9.67
Sine Function *	500	264	250	136	267.76	8.96	13.33	13.33
4x4 Checkerboard	1000	275	172	148	3.81	0.41	4.03	4.66
Pima Indian Diabetes	615	311	330	216	203.91	28.00	29.90	30.30
W-Breast Cancer	546	96	87	41	2.14	0.13	6.80	6.80
MNIST: 3-8	11982	4089	1253	1024	477.25	147.73	0.50	0.45
MNIST: 6-8	11769	1135	594	421	222.84	58.96	0.31	0.31
MNIST: 9-8	11800	1997	823	631	308.73	86.23	3.74	3.85
DMEF4	81226	8871	35529	6624	4820.06	129.29	34.83	35.13

2. Proofs on Validity of Pattern Selection in Input Space

In this section, we prove that the k nearest neighbors of a pattern in the input space I are also the k nearest neighbors of the pattern in the feature space Φ.

Definition 1 (*k*NN Invariance): Let *kNNI* ($\vec{x}$) be the set of *k* nearest neighbors of a pattern $\vec{x}$ in the input space *I*, and *kNN*Φ ($\vec{x}$) be that of the pattern Φ($\vec{x}$) in the feature space *Φ*. If both sets are identical

$$kNN_I(\vec{x}) = kNN_\Phi(\vec{x}) \;\; , \forall\, k > 0, \;\; \forall\, \vec{x}\,,$$

the invariance of the *k* nearest neighbors holds.

Finding the nearest neighbors implies distance calculation. In terms of the squared Euclidean distance which is the most commonly used distance measure, the distance among patterns in the input space *I* is

$$\|\vec{x}-\vec{y}\|^2 = \vec{x}\cdot\vec{x} + \vec{y}\cdot\vec{y} - 2\vec{x}\cdot\vec{y}. \tag{1}$$

The distance in the feature space $\varPhi$ is similarly drawn as

$$\|\varPhi(\vec{x})-\varPhi(\vec{y})\|^2 = \varPhi(\vec{x})\cdot\varPhi(\vec{x}) + \varPhi(\vec{y})\cdot\varPhi(\vec{y}) - 2\varPhi(\vec{x})\cdot\varPhi(\vec{y}) \tag{2}$$

where $\varPhi(\cdot)$ is a mapping function from the input space to the feature space, $\varPhi(\cdot): I \mapsto \varPhi$. One might obtain $\varPhi(\vec{x})$ directly but the formula is extremely complicated. Thanks to the fact that the mapping $\varPhi(\cdot)$ always appears within a form of inner product during SVM QP calculation, one thus uses *kernel trick* which substitutes the inner product to a kernel function, $\varPhi(\vec{x})\cdot\varPhi(\vec{y}) = K(\vec{x},\vec{y})$. If this kernel trick is applied to Eq.(2), then the distance in the feature space becomes

$$\|\varPhi(\vec{x})-\varPhi(\vec{y})\|^2 = K(\vec{x},\vec{x}) + K(\vec{y}\cdot\vec{y}) - 2K(\vec{x},\vec{y}) \tag{3}$$

As long as the relative distance magnitude of the input space is preserved in the feature space $\varPhi$ for all patterns, the composition of the *k* nearest neighbors of a pattern will be invariant. We now define *proximity invariance* and then prove that proximity invariance implies *kNN invariance*.

Definition 2 (Proximity Invariance): For the patterns $\vec{x}$, $\vec{y}_1$, and $\vec{y}_2$ ($\vec{x}\neq\vec{y}_1, \vec{x}\neq\vec{y}_2$, and $\vec{y}_1\neq\vec{y}_2$) in the input space *I* satisfying

$$\|\vec{x}-\vec{y}_1\|^2 < \|\vec{x}-\vec{y}_2\|^2,$$

the invariance of proximity holds if they preserve their relative distances in the feature space *Φ,*

$$\|\varPhi(\vec{x})-\varPhi(\vec{y}_1)\|^2 < \|\varPhi(\vec{x})-\varPhi(\vec{y}_2)\|^2.$$

Lemma 1: Proximity invariance implies *k*NN invariance.

$$\left\{\|\vec{x}-\vec{y}_1\|^2 < \|\vec{x}-\vec{y}_2\|^2 \Rightarrow \|\varPhi(\vec{x})-\varPhi(\vec{y}_1)\|^2 < \|\varPhi(\vec{x})-\varPhi(\vec{y}_2)\|^2\right\} \Rightarrow \left\{\, kNN_I(\vec{x}) = kNN_\varPhi(\vec{x})\,\right\}.$$

Proof: To avoid complication, assume that there do not exist two different neighbors of $\vec{x}$, $\vec{y}_1$, and $\vec{y}_2$, such that $\|\vec{x}-\vec{y}_1\|^2 = \|\vec{x}-\vec{y}_2\|^2$ *or* $\|\varPhi(\vec{x})-\varPhi(\vec{y}_1)\|^2 = \|\varPhi(\vec{x})-\varPhi(\vec{y}_2)\|^2$. Let $\vec{k}_i^I$ and $\vec{k}_i^\varPhi$ denote the i^{th} nearest neighbor of the pattern $\vec{x}$ in the input space *I* and that in the feature space *Φ*, respectively. Then, the *k* nearest neighbors' set of the pattern $\vec{x}$ in each space is defined from training set D as

$$kNN_I(\vec{x}) = \{\; \vec{k}_i^I \in D \mid i = 1,\ldots,k \;\},$$

$$kNN_\varPhi(\vec{x}) = \{\; \vec{k}_i^\varPhi \in D \mid i = 1,\ldots,k \;\}.$$

Now, suppose that the k nearest neighbors' set of the pattern $\vec{x}$ is not invariant,

$$kNN_I(\vec{x}) \neq kNN_\Phi(\vec{x}) \ , \tag{4}$$

Eq.(4) implies that there exists i such that $\vec{k}_i^I \neq \vec{k}_i^\Phi$ and $\vec{k}_j^I = \vec{k}_j^\Phi$, $j = 1,\ldots,i-1$. In the input space I, the following inequality of the distances from $\vec{x}$ to $\vec{k}_i^I$ and to $\vec{k}_i^\Phi$ holds

$$\left\|\vec{x}-\vec{k}_i^I\right\|^2 < \left\|\vec{x}-\vec{k}_i^\Phi\right\|^2, \tag{5}$$

since $\left\|\vec{x}-\vec{k}_i^I\right\|^2$ produces the minimum distance out of the pattern set $D' = D - \{\vec{x},\ \vec{k}_1^I, \vec{k}_2^I, \ldots, \vec{k}_{i-2}^I, \vec{k}_{i-1}^I\}$ which $\vec{k}_i^\Phi$ belongs to. Similarly, in the feature space Φ, the following inequality holds

$$\left\|\Phi(\vec{x})-\Phi(\vec{k}_i^I)\right\|^2 > \left\|\Phi(\vec{x})-\Phi(\vec{k}_i^\Phi)\right\|^2 \tag{6}$$

Proximity invariance leads Eq.(5) into

$$\left\|\Phi(\vec{x})-\Phi(\vec{k}_i^I)\right\|^2 < \left\|\Phi(\vec{x})-\Phi(\vec{k}_i^\Phi)\right\|^2$$

which is contradictory to Eq.(6). Thus, non-invariance assumption made in Eq.(4) is false. Therefore, kNN invariance holds if proximity invariance holds. ∎

Lemma 2 (Proximity Invariance for RBF Kernel): Proximity invariance holds when the mapping function $\Phi(\vec{x})$ is defined such that

$$\Phi(\vec{x}) \cdot \Phi(\vec{y}) = K(\vec{x},\vec{y}) = \exp\{-\|\vec{x}-\vec{y}\|^2 / 2\sigma^2\} .$$

Proof: Let $\vec{y}_1$ and $\vec{y}_2$ are two distinct neighbors of $\vec{x}$ with $\|\vec{x}-\vec{y}_1\|^2 < \|\vec{x}-\vec{y}_2\|^2$, i.e., $\vec{y}_1$ is closer to $\vec{x}$ than $\vec{y}_2$. Suppose the invariance of proximity does not hold for a mapping function $\Phi(\vec{x})$, i.e., $\|\Phi(\vec{x})-\Phi(\vec{y}_1)\|^2 \geq \|\Phi(\vec{x})-\Phi(\vec{y}_2)\|^2$. Using Eq.(3), one can rewrite the inequality as

$$K(\vec{x},\vec{x}) + K(\vec{y}_1 \cdot \vec{y}_1) - 2K(\vec{x},\vec{y}_1) \geq K(\vec{x},\vec{x}) + K(\vec{y}_2 \cdot \vec{y}_2) - 2K(\vec{x},\vec{y}_2) .$$

Since $K(\vec{a},\vec{a}) = 1$ and $K(\vec{a},\vec{b}) > 0$, the inequality is simplified as

$$K(\vec{x},\vec{y}_1) \leq K(\vec{x},\vec{y}_2) .$$

Plugging the definition of RBF kernel, we obtain

$$\exp\{-\|\vec{x}-\vec{y}_1\|^2 / 2\sigma^2\} \leq \exp\{-\|\vec{x}-\vec{y}_2\|^2 / 2\sigma^2\} ,$$

which in turn can be simplified into

$$\|\vec{x}-\vec{y}_1\|^2 \geq \|\vec{x}-\vec{y}_2\|^2 .$$

This is contradictory to our initial assumption that $\vec{y}_1$ is closer to $\vec{x}$ than $\vec{y}_2$. Thus the assumption that *the invariance of proximity does not hold* is not true. ∎

Theorem 1: kNN invariance holds for RBF kernel.

Proof: In Lemma 2, we proved that proximity invariance holds for RBF. Due to Lemma 1, kNN invariance also holds for the kernel.

3. Conclusion

In this paper, we proved the k-nearest neighbor invariance under input to feature space mapping. The result leads us to conclude that the patterns selected in input space are identical to the patterns selected in feature space if neighborhood relation is used. Thus, selecting patterns in input space that are likely to be support vectors in feature space is justified. We only provided proof for the case of RBF kernel, but proofs for other kernels should be similar in nature.

Acknowledgment

This research was supported by Brain Science and Engineering Research Program sponsored by Korean Ministry of Science and Technology.

References

Cauwenberghs, G. & Poggio, T. (2001). Incremental and Decremental Support Vector Machine Learning, *A*dvances in Neural Information Processing Systems, Cambridge MA: MIT Press, 13, 409–415.

Hearst, M. A., Schölkopf, B., Dumais, S., Osuna, E., & Platt, J. (1997). Trends and Controversies Support Vector Machines, *IEEE Intelligent Systems,* 13, 18–28.

Platt, J. C. (1999). Fast Training of Support Vector Machines Using Sequential Minimal Optimization, In: *Advances in Kernel Methods: Support Vector Machines* (pp. 185–208), Cambridge MA: MIT press.

Pontil, M. & Verri, A. (1998). Properties of Support Vector Machines, *Neural Computation,* 10, 955–974.

Schölkopf, B., Burges, C., & Vapnik, V. (1995). Extracting support data for given task, In: *Proc. of 1st International Conference on Knowledge Discovery and Data Mining* (pp. 252–257), AAAI.

Schölkopf, B. & Smola, A. J. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond,* MIT press.

Shin, H. J. & Cho, S. (2002). Pattern Selection for Support Vector Classifiers, In: *Proc. of the 3rd International Conference on Intelligent Data Engineering and Automated Learning* (pp. 469–474), Manchester: LNCS 2412.

Shin, H. J. & Cho, S. (2003a). Fast Pattern Selection for Support Vector Classifiers, In: *Proc. of the 7th Pacific Asia Conference on Knowledge Discovery and Data Mining* (pp.376–387), Seoul: LNAI 2637.

Shin, H. J. & Cho, S. (2003b). Fast Pattern Selection Algorithm for Support Vector Classifiers: Time Complexity Analysis, In: *Proc. of the 3rd International Conference on Intelligent Data Engineering and Automated Learning* (pp.1008–1015), Hong Kong: LNCS 2690.

Shin, H. J. & Cho, S. (2003c). How Many Neighbors To Consider in Pattern Pre-selection for Support Vector Classifiers?, In: *Proc. of the International Joint Conference on Neural Networks* (pp. 565–570), Portland.

Shin, H. J. & Cho, S. (2003d). Response Modeling with Support Vector Machines, (submitted).

Vapnik, V. (1999). *The Nature of Statistical Learning Theory* (2nd ed.), Springer.