신경망을 이용한 빠른 서포트 벡터 분류

김광인
kimki@ai.kaist.ac.kr

Fast Support Vector Classification based on Artificial Neural Networks

Kwang In Kim
A. I. Lab, CS Dept., Korea Advanced Institute of Science and Technology

요 약

본 논문에서는 빠른 서포트 벡터 분류를 위해 신경망을 사용하는 방법을 제안한다. 주어진 학습 데이터를 통해 낮은 학습 오류를 가지는 다단계 신경망을 얻으면 출력층을 제외한 은닉층은 주어진 문제를 선형분리 가능하게 하는 특징 추출기로 간주 할 수 있다. 많은 계산시간을 요하는 커널 맵 대신 이를 사용해서 빠른 서포트 벡터 분류를 가능하게 하였다.

1. Introduction

Support vector machines (SVMs) are recently introduced as a method for pattern classification and regression. They have shown impressive performance in many problems [1], [2] and have been widely accepted as one of the strongest classifiers. The application areas of the SVM are, however, limited because of its high run-time complexity which is mainly caused by expanding the solution in terms of *kernel map*. Here we propose to use an artificial neural network (ANN) as a replacement of kernel map. Training an ANN on a given problem to achieve low training error and taking up to the last hidden layer makes the given problem linearly separable and enables application of linear SVMs. To avoid the resulting overfitting of the ANN *feature extractor*, optimal brain surgeon (OBS) algorithm is adopted as an implicit regularization. The method is evaluated based on the color texture-based license plate (LP) detection problem.

2. Brief Overview of Support Vector Machine Classifier

For the pattern classification problem, from a given set of labeled training examples $(\mathbf{x}_i,y_i)\in\mathbf{R}^N\times\{\pm1\}$, $i=1,\ldots,l$, an SVM constructs a linear classifier by determining the separating hyperplane that has maximum distance to the closest points of the training set (called *margin*). The appeal of SVMs lies in their strong connection to the underlying statistical learning theory. According to the structural risk minimization principle [3], a function that can classify training data accurately and which belongs to a set of functions with the lowest capacity (particularly in the VC-dimension) will generalize best, regardless of the dimensionality of the input space. In the case of separating hyperplanes, the VC-dimension h is upper bounded by a term depending on the margin Δ and the radius of the smallest sphere R including all the data points as follows [3]

$$h \le \min\left(\left\lceil \frac{R^2}{\Delta^2} \right\rceil, N\right)+1. \quad [1] \tag{1}$$

Accordingly, SVMs approximately implement SRM by maximizing Δ for a fixed R (since the example set is fixed). The solution of an SVM is obtained by solving a QP problem which shows that the SVM as a linear classifier is represented equivalently as either

$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{x}\cdot\mathbf{w}+b) \tag{2}$$

or

$$f(\mathbf{x}) = \mathrm{sgn}\left(\sum_{i=1}^{l^*} y_i\alpha_i\mathbf{x}_i^*\cdot\mathbf{x}+b\right), \tag{3}$$

where $\mathbf{x}_i^*$'s are a subset of training points lying on the margin (called *support vectors* (SVs)).

The basic idea of nonlinear SVMs is to project the data into a high-dimensional *Reproducing Kernel Hilbert Space* (RKHS) F which is related to the input space by a nonlinear map $\Phi:\mathbf{R}^N\to F$ [2]. An important property of an RKHS is that the inner product of two points mapped by Φ can be evaluated using *kernel functions*

$$k(\mathbf{x},\mathbf{y}) = \Phi(\mathbf{x})\cdot\Phi(\mathbf{y}), \tag{4}$$

which allows us to compute the value of the inner product without having to carry out the map Φ explicitly. By replacing each occurrence of inner product in Eq. (3) with kernel function (4), the SVM can be compactly represented even in very high-dimensional (possibly infinite) spaces:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{l^*} y_i\alpha_i k(\mathbf{x}_i^*,\mathbf{x})+b\right). \quad [2] \tag{5}$$

To gain an insight into the performance of SVMs, an SVM with the polynomial kernel of degree 2 ($k(\mathbf{x},\mathbf{y})=(\mathbf{x}\cdot\mathbf{y})^2$) was applied to the LP detection problem [4]. The objective is to classify each pixel in an image into *plate* class or *non-plate* class based on their local color texture properties (RGB values of surrounding square image patch (11×11)). The SVM was trained on approximately 20,000 plate and non-plate patterns. The training set was initialized with 2,000 patterns sampled from a database of 100 vehicle images and is augmented by performing bootstrapping [5] on the same database. The trained SVM was composed of around 1,700 SVs and showed 2.7% training error rate. For testing, another set of 10,000 plate and non-plate patterns were collected from 350 vehicle images which are distinct from the images used in training. The SVM achieved 4.4% error rate with processing time of 1.7 sec. on 10,000 patterns (which corresponds to approximately 24 sec. for processing a 320x240-sized image). In terms of classification accuracy, the SVM was superior to several other existing methods (Sec. 4) and is suitable to LP detection application. However, the processing time of the SVM is far below the acceptable level. The main computational burden lies in evaluating the kernel map in Eq. (5) which is in proportional to the dimensionality of the input space and the number of SVs. In the case of

[1] This bound (called *radius margin bound*) holds in only linearly separable case. For the generalization of this bound for linearly non-separable case, readers are referred to Sec.4.3.3 of [11].

[2] The architecture of nonlinear SVM can be viewed from two different perspectives: one is the linear classifier lying in F and is parameterized by $(\mathbf{w},b)$ (cf. Eq. (2)). Another is the linear classifier lying in the space spanned by the *empirical kernel map* with respect to the training data $(k(\mathbf{x}_1,\cdot),\ldots,k(\mathbf{x}_l,\cdot))$ and is parametereized by $(\alpha_1,\ldots,\alpha_{l^*},b)$ (cf. Eq. 2). The second viewpoint characterizes the SVM as a two-layer ANN.

polynomial kernel, this implies 1,700 inner-product operations in processing a single pattern.

There are already several methods to reduce the runtime complexity of SVM classifiers, including reduced set method [2], cascade architecture of SVMs [6], etc. The preliminary experiment with reduce set method has shown that at least 400 SVs were required to retain a moderate reduction of generalization performance,[3] which is still not enough to realize a practical LP detection system. The application of cascade method is beyond the scope of this work. However, it should be noted that cascade architecture is independent of a specific classification method and can be directly applied in the method proposed in this paper.

3 Combining artificial neural networks with support vector machines

Recall that the basic idea of the nonlinear SVM is to cast the problem into a space where the problem is linearly separable and then use linear SVMs in that space. This idea is validated by Cover's theorem on the separability of patterns [7],[8]:

A complex pattern-classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space

The main advantage of using high-dimensional spaces to make the problem linearly separable is that it enables analysis to stay within the class of linear functions and accordingly leads to the convex optimization problem. On the other hand, as a disadvantage, it requires the solution to be represented in kernel expansion since one can hardly manipulate the solution directly in such spaces.

The basic intuition in using the ANN is to cast the problem into a moderately low-dimensional space where the problem is still (almost) linearly separable. In other words, we perform Φ explicitly and construct the linear SVM in terms of the direct solution (2) rather than the kernel representation (3) in the range of Φ . This reduces the computational cost when the dimensionality of the corresponding feature space is lower than the number of SVs. As demonstratd in many practical applicatins [8], ANNs has an ability to find a local minima in the empirical error surface. If it once achives an acceptably low error rate, it is guranteed that the problem in the space constructed by up to the last hidden layer is (almost) linearly separable since the output layer is simply a linear classifier (Fig. 1). The final classifier is then obtained by replacing the output layer of the ANN with a linear SVM.

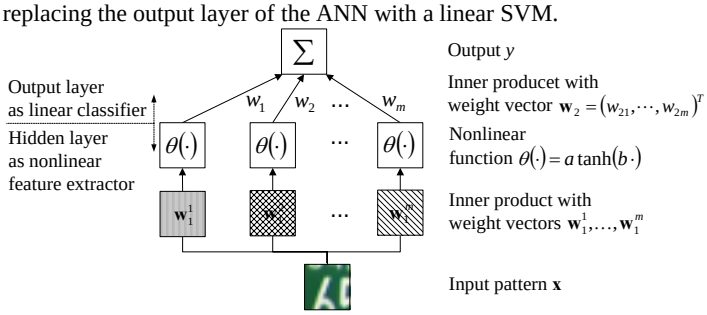


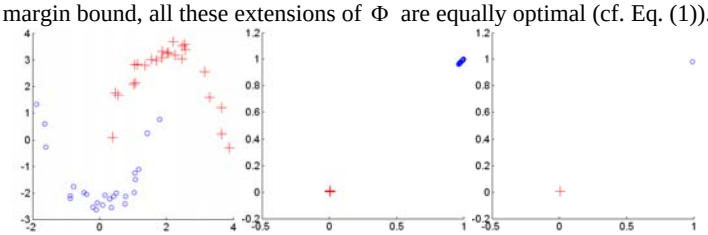
Fig.1. Example of two-layer ANN (single hidden layer).

However, this strategy alone does not work as examplified in Fig. 2: for a two dimensional toy problem, an ANN with two hidden layers of size 30 and 2 is trained: (a) plots the traing examples in the input space while (b) plots the activation of the last hidden layer corresponding to training examples in (a). The ANN feature extractor (up to the last hidden layer) did make the problem linearly separable. At the same time, it clustered training examples almost perfectly by mapping them (close) to two cluster centers according to class labels. Certainly, the trained ANN failed to generalize. This is an example of overftting occurred during the feature extraction. In this case, even the SVM with capicity control capability does not have any chance to generalize better than the ANN since all the information contained in the training set is already lost during the feature extraction.

From the model selection viewpoint, choosing the feature map Φ is an optimization problem where one control the kernel parameter (and equivalently the Φ) to minimize the cross validation error or a generalization error bound. Within this framework, one might try to train simultaneously both the feature extractor Φ and classifier based on the unified criterion of the generalization error bound. However, we argue that this method is not applicable to the ANN as Φ . Here we give two examples of error bounds which are commonly used in model selection.

In terms of the radius margin bound (Eq. (1)), the model selection problem is to minimize the capacity h by controlling both the margin Δ and the radius of the sphere R. Since the problem of estimating Δ from a fixed set of training examples is convex, both the Δ and R are solely determined by choosing Φ .

Let us define a map Φ which maps all the training examples into two points corresponding to their class labels, respectively (Fig. 2(c)). Certainly, there are infinitily many extensions of Φ for unseen data points, most of which may generalize poorly. However, for the radius margin bound, all these extensions of Φ are equally optimal (cf. Eq. (1)).



(a) (b) (c)

Fig. 2. Example of overfiting occurred during feature extraction: (a) input data, (b) activation of the last hidden layer corresponding to the input data, and (c) extremal case of overfitting map.

The *span bound* [9] provides an efficient estimation of the *leave-one-out* (LOO) error based on the geometrical analyis of the *span of SVs*. It is characterized by the cost of removing an SV and approximating the original solution based on the linear combination of remaining SVs. Again, the Φ in Fig. 2(c) is optimal in the sense that all training examples became duplicates of one of two SVs in F and accordingly the removal of one SV (one duplicate) does not affect the solution.[4]

Certainly, the example map in Fig. 2(c) is unrealistic and is even not optimal for all existng bounds. However it clarifies the inherent limitation in using error bounds for choosing the Φ : when the Φ is chosen based on a specific error bound from a large class of functions (containing very complex functions), there is no way to avoid overfitting. It should be noted that this is essentially the same problem to that of choosing the classifier in a fixed feature space F, whose solution suggests controlling the complexity[5] of Φ to avoid overfitting.[6]

There are already several methods developed for reducing the complexity of ANNs including early stopping of trainng, weight decaying, etc. We rely on the network pruning approach where one starts moderately large network with high complexity and prun the network based on the criteria of minimizling the *demage* of network. This type of algorithms does not have direct relation to any existing complexity control methods (eg., regularization). However, they are simple and have

[3] The reduced set method tries to find an approximate solution which is expanded in a small set of vectors (called reduced set). Finding the reduced set is a nonlinear optimization problem which in this paper, is solved using the fixed point iteration method. For details, readers are referred to [2]. The less number of SVs than 400 produced significant increase in the error rate.

[4] It should be noted that the map in Fig. 2(c) is optimal even in terms of the true LOO error.

[5] While there are various existing notion of complexity of function class including VC-dimension, smoothness, the number of parameters, etc., no specific definition of complexity is imposed on the class of Φ , since we will not directly minimize any of them. Accoridngly, the 'complexity' here should be understood in abstract sense and not be confused with VC-dimension.

[6] The reported excellent performances of model selection methods based on error bounds [3],[9] can be explained by the fact that the class of functions where the kernel is chosen was small (e.g., polynomial kernels or Gaussian kernels with one or few parameters).

advantage of providing the decomposition of chosing Φ into two sub-problems: 1) obtaining a linearly separable map and 2) reducing the complexity of the map while retaining the linear separability.

For the first objective, an ANN with one hidden layer of size 47 is trained based on the *back-propagation* algirithm. The use of only one hidden layer as the featuer extractor is supported by the architecture of the kernelized SVM where only one hidden layer (depending on the type of kernel function) is often enough to gurantee the linear separability. The number of hidden nodes is chosen based on trial and error: a fairly large network of 100 hidden nodes was initially constructed and is reduced by removing nodes by ones while retaining the training error similar to that of an SVM with degree 2 polynomial kernel.

The trained ANN is then prunned based on the OBS algorithm which evaluate the demage of the network based on the quadratic approximation of increase of training error and prune it to obtain the minimal demage. Here we give a brief review of OBS. For more detail, readeres are referred to [10],[8].

The basic idea of OBS is to use the second order approximation of error surfaces. Suppose for a given configuration of weights **w** (cf. Fig. 1), the cost function E in terms of empirical error is represented baed on *Taylor series* about **w**:

$$E(\mathbf{w}+\Delta\mathbf{w}) = E(\mathbf{w}) + \mathbf{g}^T(\mathbf{w})\Delta\mathbf{w} + \frac{1}{2}\Delta\mathbf{w}^T\mathbf{H}(\mathbf{w})\Delta\mathbf{w} + O\left(\|\Delta\mathbf{w}\|^3\right),$$

where $\Delta\mathbf{w}$ is a perturbation applied to the operating point **w** and $\mathbf{g}(\mathbf{w})$ and $\mathbf{H}(\mathbf{w})$ are the gradient vector and the Hessian matrix evaluated at the point **w**, respectively. Assuming that the **w** is located at the local optima and ignoring the third and all higher order terms. We get the approximation of increase in the error E based on $\Delta\mathbf{w}$ as follows

$$\begin{aligned}\Delta E &= E(\mathbf{w}+\Delta\mathbf{w}) - E(\mathbf{w}) \\ &\approx \frac{1}{2}\Delta\mathbf{w}^T\mathbf{H}(\mathbf{w})\Delta\mathbf{w}.\end{aligned} \tag{6}$$

The goal of OBS is to find the index i of a particular weight w_i which minimize ΔE when w_i is set to zero. The elimination of w_i is equivalent to the condition

$$\Delta w_i + w_i = 0\,. \tag{7}$$

To solve this constrained optimization problem, we construct the *Lagrangian*

$$S_i = \frac{1}{2}\Delta\mathbf{w}^T\mathbf{H}(\mathbf{w})\Delta\mathbf{w} - \lambda(\Delta w_i + w_i)$$

where λ is the *Lagrange multiplier*. Taking the derivative of S_i with respect to $\Delta\mathbf{w}$, applying the constraint of (7), and using matrix inversion, the optimum change in the weight vector **w** and resulting change in error (optimal value of S_i) are obtained as

$$\Delta\mathbf{w} = -\frac{w_i}{\left[\mathbf{H}^{-1}\right]_{i,i}}\mathbf{H}^{-1}\mathbf{1}_i \tag{8}$$

and

$$S_i^* = \frac{w_i^2}{2\left[\mathbf{H}^{-1}\right]_{i,i}}, \tag{9}$$

respectively, where $\left[\mathbf{H}^{-1}\right]_{i,i}$ is the (i,i)-th element of the inverse of the Hessian matrix $\mathbf{H}^{-1}$.

The OBS procedure for constrcuting the feature extractor Φ is summarized in Fig. 3.

1. Compute $\mathbf{H}^{-1}$
2. Find the index i that gives the smallest *sailency* value S_i^*. If the S_i^* is larger than a given threshold, then go to step 4.
3. Use the i from step 2 to update all weights according to (8). Go to step 2.
4. Retrain the network based on standard back propagation algorithm.

Fig. 3. OBS procedure for prunning ANN.

Although the OBS procedure does not have any direct correspondence to the regularization framework, the ANN obtained from the OBS will henceforth be refered to as the *regularized* ANN.

4. Experimental Results

The proposed method was tested using an LP image database of 450 images. For details of experimental setting, readers are referred to [4]. For training the ANN+SVM classifier, 20,000 training examples which were used in training the base SVM classifier (Sec. 2.1) were used: the ANN was firstly trained on random selection of 10,000 patterns. Then, the linear SVM was trained on the output of the ANN feature extractor on whole 20,000 patterns (including 10,000 patterns used to train the ANN). The size (number of weights) of the ANN feature extractor was initially 5,781 which were reduced to 843 after OBS procedure where the stopping criterion was 0.5% increase of training error rate. The testing environment was 2.2 GHz CPU with 1.2GB RAM. Table 1 summarizes the performances of various classifiers: the ANN and nonlinear SVM (with polynomial kernel of degree 2) have shown the best and worst error rates and processing times, respectively. Simply replacing the output layer of the ANN with an SVM did not provide any significant improvement as anticipated in Section 3, while the regularization of the ANN has already showed improved classification rate. The combination of the regularized ANN with SVM produced the second best error rate and the processing time which can be regarded as the best overall.

Table 1. Performance of different classification methods.

Classifier	Error rate (%)	Proc. time (10,000/ sec.)
ANN	7.31	0.14
ANN+SVM	6.87	0.14
Regularized ANN	5.1	0.06
Regularized ANN+SVM	4.48	0.08
Nonlinear SVM	4.43	1.74

5. Conclusions

The problem of high run-time complexity of SVMs was approached by utilizing a regularized ANN as the feature extractor. In comparison with the standard nonlinear SVM, classification performance of the proposed method was only slightly worse while the run-time is significantly better. Accordingly, it can provide a moderate alternative to the standard kernel SVMs in real-time applications.

Acknowledgement. The ideas presented in this paper have greatly profited from discussions with Matthias Hein, Arthur Gretton, Olivier Bousquet, and Jahwan Kim.

References

[1] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," *Proc. IEEE CVPR*, 1997, pp.130-136.
[2] B. Schölkopf and A. J. Smola, *Learning with Kernels*, MIT Press, 2002.
[3] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.
[4] K. I. Kim, K. Jung, and J. Kim, "Color texture-based object detection: an application to license plate localization," *in Proc International Workshop on Pattern Recognition with Support Vector Machines*, pp. 293-309, 2002.
[5] K. K. Sung, *Learning and example selection for object and pattern detection*, PhD thesis, MIT AI Lab, 1996.
[6] B. Heisele, T. Serre, S. Prentice, and T. Poggio, "Hierarchical classification and feature reduction for fast face detection with support vector machines," *Pattern Recognition*, vol. 36, pp. 2007-2017, 2003.
[7] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Trans. Electronic Computers*, vol. 14, pp. 326-334, 1965.
[8] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd Ed. Prentice Hall, 1998.
[9] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing kernel parameters for support vector machines," *Machine Learning*, pp. 131-159, 2000.
[10] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: optimal brain surgeon," in *Advances in Neural Information Processing Systems*, vol. 5, S. J. Hanson, J. D. Cowan, and C.L. Giles, eds., pp. 164-171, Morgan Kaufmann, 1993.
[11] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, 2000.