Journal of the Eastern Asia Society for Transportation Studies, Vol. 6, pp. 2589 - 2599, 2005

PERFORMANCE IMPROVEMENT IN TRAFFIC VISION SYSTEMS USING SVMS

Daehyon KIM
Assistant Professor
Transportation & Logistics System Eng.
Yosu National University
Dundeok-Dong, Yeosu-Si, Jeollanam-Do
Korea
Fax: +82-61-659-3349
E-mail: daehyon@yosu.ac.kr

Seongkil CHO
Assistant Professor
Department of Geoinfomatics
University of Seoul
Chunnong-dong, Dongdaemun-ku, Seoul
Korea
Fax: +82-2-2246-0186
E-mail: skcho@uos.ac.kr

Yongtaek LIM
Assistant Professor
Transportation & Logistics System Eng.
Yosu National University
Dundeok-Dong, Yeosu-Si, Jeollanam-Do
Korea
Fax: +82-61-659-3349
E-mail: limyt@yosu.ac.kr

Abstract: Computer vision system is one of important research topics in ITS(Intelligent Transport Systems). Moreover, Neural Networks have been increasingly and successfully applied to many problems for ITS. Even though there are currently many different types of neural network models, Backpropagation is the most popular neural network model. It is however known that the Support Vector Machines (SVMs) based on the statistical learning theory is currently another efficient approach for pattern recognition problem since their remarkable performance in terms of prediction accuracy. In this research, two different models, Backpropagation and SVMs have been studied to compare their performance in predictive accuracy through the experiment with real world image data of traffic scenes. Experimental results show that SVMs can provide higher performance in terms of prediction performance than any other models.

Key Words: Intelligent Transport Systems, Backpropagation, Support Vector Machines

1. INTRODUCTION

Computer vision extracting meaningful information from the image collected by the video camera is a key element for integrated automatic traffic surveillance and control systems, and it is one of main ITS(Intelligent Transport Systems) research subjects. Over the last few years, there has been increasing research on image processing for automatic traffic data collection - vehicle counting, vehicle speed detection, vehicle tracking, congestion and incident detection - because image processing is potentially more powerful and flexible than methods currently available, such as the buried induction loop detector, microwave, or infra-red beams. Even though there has been growing interest in the application of image processing in transportation engineering, detection algorithms are still unreliable in the complicated

circumstances, and more efficient algorithms should be developed.

Until recently, the Artificial Neural Networks, which hold considerable potential for recognizing and classifying spatial and temporal patterns, have been used as an efficient method for vehicle detection and classification. Especially, the multilayer feedforward using Backpropagation learning algorithm, which is one of the most popular neural networks, has been applied successfully to various problems in transportation engineering. However, Support Vector Machines (SVMs), which are generation learning systems based on advances in statistical learning theory, receive a great deal of attention recently with their remarkable performance. After the SVMs were introduced by Vapnik(Vapnik, 1995), they have been successfully applied to numerous pattern recognition problems, including object detection(Blanz et al., 1996), handwritten character recognition(Cortes and Vapnik, 1995; Schölkopf et al., 1995; Schölkopf et al., 1996), text categorization(Joachims, 1998), and face detection in images(Osuna et al., 1997).

In this research, SVMs have been used for vehicle detection in image processing-based traffic surveillance system. The prediction performance will be also compared with the multilayer feedforward network using Backpropagation using real world traffic images.

2. BACKPROPAGATION NEURAL NETWORK MODEL

A general model of the Backpropagation network is shown in Figure 1. The layers are organized as one input layer, L-1 hidden layers, and one output layer, with a one-directional flow of information from the input to the output. The symbol x denotes an input, y an output, w a connection weight.

During the feedforward calculations, two mathematical operations are performed by each neuron; the first is a linear component that computes the weighted sum of the node's input value ($\sum$), and the second is a non-linear component that transforms the weighted sum into an output value using an activation function, $f(\ )$.

While the computation for output of each neuron is working forward from the input to the output layer, the error at the output of every layer is propagated back to the previous layer, and the weights are changed so as to decrease that error. Mathematically, the feedforward processing can be written as follows;

Define input vector to the network $\mathbf{x}_p = (x_{p1}, x_{p2}, x_{p3}, ..., x_{pK_0})^T$ and output vector $\mathbf{y}_p = (y_{p1}, y_{p2}, y_{p3}, ..., y_{pK_L})^T$ for p =1,..., P, where the subscript p refers to the p^{th} training pattern, K_0 is the number of input layer nodes and K_L is the number of output layer nodes. Let each layer be denoted by l, i.e. input layer (l = 0), hidden layer(s) (l = 1,…, L-1), and output layer ($l = L$). There are K_l nodes in each layer. Inputs to the k^{th} neuron in the layer l are In_{pj}^{l} for j = 1,..., K_{l-1}. These inputs are the output values from previous layer, i.e. $In_{pj}^{l} = Out_{pj}^{l-1}$.

The net input for the k^{th} neuron in the layer l is

$$Net_{pk}^{l} = \sum_{j=1}^{K_{l-1}} w_{kj}^{l} In_{pj}^{l} + \theta_{k}^{l} \quad (1)$$

where w_{kj}^{l} is the weight on the connection between node j and k, and θ_{k}^{l} is the bias term. The output processed by k^{th} node in layer l is obtained by a transfer function, $f_{k}^{l}(\ )$ as follows

$$Out_{pk}^{l} = f_{k}^{l}(Net_{pk}^{l}) \quad (2)$$

The activation function, denoted by $f(\ )$, defines the output of a neuron in terms of the activity level at its input. The network training process is to determine the weight vectors that minimize an error function.

The training time of the standard Backpropagation is typically long when complex decision regions are required and when the network has many hidden neurons. In order to increase the speed of convergence, Rumelhart et al. (1986) also proposed an enhanced Backpropagation model, Backpropagation with Momentum, by adding a Momentum term to the delta rule. In the model, a fraction of previous change is added when calculating the weight change value. This additional term called momentum tends to keep the weight changes going in the same direction. The momentum parameter provides a kind of momentum in weight space that effectively filters out high-frequency variations of the error-surface in the weight space.

Fahlman (1988) has developed an algorithm called Quickprop to speed the learning of the Backpropagation model. One method to speed up the learning is to use information about the curvature of the error surface. This involves the computation of the second order derivatives of the error function. Quickprop assumes the error surface to be locally quadratic and attempts to jump in one step from the current position directly into the minimum of the parabola (Fahlman, 1988). The Quickprop weight update rule achieves impressive speedups in a number of instances, although it may also fail to converge in situations where Backpropagation would eventually reach an acceptable answer (Fausett, 1994). Quickprop has some more parameters, such as Maximum growth factor, Weight decay, and Prime-offset, that should be fine-tuned than standard Backpropagation.

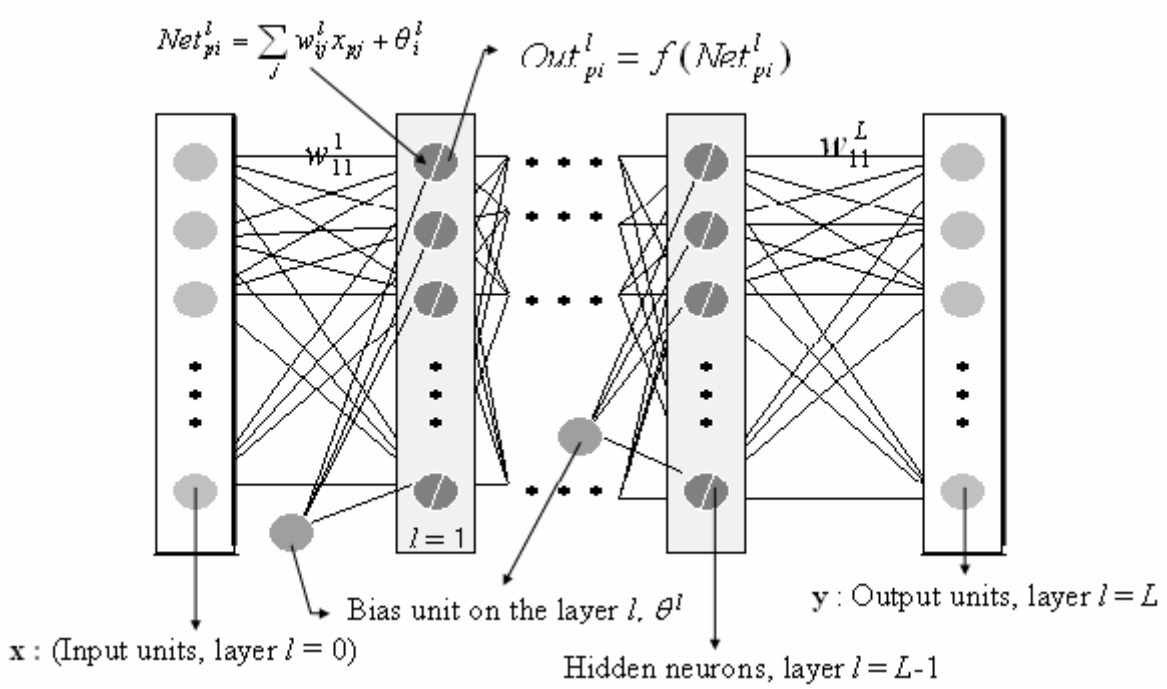


Figure 1. Network architecture of Backpropagation

During training of the standard Backpropagation and the Backpropagation with Momentum, output units may get stuck in a bad state for some training patterns because of the properties of the sigmoid function. This places severe limitations on the convergence speed of the Backpropagation method. This phenomenon is known as the flat-spots problem. One solution to the flat-spots problem is to ensure that the derivative function does not approach zero. Fahlman (1988) suggested the addition of a constant parameter, known as the Prime-offset, to the derivative of sigmoid function before using it. This keeps the weight changes for an output unit that is far from its target value from approaching zero. An alternative solution to the flat-spots problem has been explored in (Balakrishnan and Honavar, 1992), where they modified error function.

Even though the parameter of the Prime-offset has been suggested for the Quickprop by Fahlman (1988), it also can be applied to the standard Backpropagation or the Backpropagation with momentum model to avoid the flat-spots problem. It has been shown that the BPMP (Backpropagation with momentum & Prime-offset) which is added the Prime-offset parameter to the Backpropagation with Momentum was more efficient than other Backpropagation models in terms of prediction accuracy and computing cost for training (Kim, 2002). In this research, the BPMP model has been used for the experiments.

3. SUPPORT VECTOR MACHINES

3.1 Linear and Separable Classification Problems

Let the training data $(\mathbf{x}_i, y_i)$, $i = 1,\ldots,l$, $y_i \in \{\pm 1\}$ $\mathbf{x}_i \in \mathbf{R}^N$, then the support vector algorithm simply looks for the optimal hyperplane with largest margin. This can be formulated as follows:

$$\text{Min} \quad \tau(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{3}$$

$$\text{s. t.} \quad y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0, \quad i = 1,...,l \tag{4}$$

where **w** is a normal to the hyperplane and b is bias or offset. Using Lagrange multipliers, $\alpha_i \geq 0$, the primal form of the objective function can be:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i (y_i((\mathbf{x}_i \cdot \mathbf{w}) + b) - 1) \tag{5}$$

The Lagrangian L has to be minimized with respect to the primal variables **w** and b and maximized with respect to the dual variables α_i. From the Karush-Kunhn-Tucker(KKT) conditions, the derivative of L with respect to the primal variables must be vanish(Fletcher, 1987), subject to constraints $\alpha_i \geq 0$, i.e.,

$$\frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0, \quad \frac{\partial}{\partial b} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0 \tag{6}$$

Equation (6) leads to

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{7}$$

Equation (7) is equality constraints in the dual formulation, and following Equation (8) which is the Wolfe dual of the optimization problem is given by substituting it into Equation (5).

$$\max \quad W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{8}$$

$$\text{s.t.} \quad \alpha_i \geq 0, \text{ and } \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{9}$$

The hyperplane decision function can thus be given as

$$f(x) = \text{sgn}\big((\mathbf{x} \cdot \mathbf{w}) + b\big) = \text{sgn}\Big(\sum_{i=1}^{l} y_i \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + b\Big) \tag{10}$$

3.2 Soft Margin Hyperplane

In practice, real-world data sets are linearly inseparable in input space since a high noise level causes a large overlap of classes. This means that the constraints need to be relaxed somewhat to allow for the minimum amount of misclassification. The points that subsequently fall on the wrong side of the margin are considered to be errors, and slack variables have been introduced in order to deal with these errors(Cortes and Vapnik, 1995; Vapnik, 1995).

By adding slack variables $\xi_i \geq 0$ $(i = 1,...l)$ to Equation (1) and (2), the objective function and relaxed constraints are:

$$\min \quad \tau(\mathbf{w}, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l} \zeta_i \tag{11}$$

$$\text{s. t.} \quad y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i, \quad i = 1,...,l \tag{12}$$

where C is upper bound for the Lagrange multiplier, λ_i, i.e., $0 \leq \lambda_i \leq C$. It is the trade-off

between maximum margin and classification error and a higher C value will give a large penalty for classification error. The only difference from the separable problem is the upper bound C on the Lagrange multiplier (Schölkopf et al., 1999).

3.3 Nonlinear classification problems

To handle nonlinear problems, the optimal hyperplane algorithm needs to be augmented. The basic idea is to transform the data from the origin space into another dot product space called the *Feature Space*. This can be achieved by a mapping function Φ:

$$\Phi : \Re^N \to F \tag{13}$$

In this high dimension space, the data can be linearly separable, hence above linear algorithm can be applied for the problem. Then the training algorithm would only depend on the data through dot products in F, i.e., on functions of the form $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. However, if F is high-dimensional, the dot product will be very expensive to compute. The dot product in the high dimension space can be replaced by a kernel function:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \tag{14}$$

By the use of a kernel function, it is possible to compute the separating hyperplane without explicitly carrying out the map into the feature space. In this research, Equation (15) of Gaussian kernel, usually called *Radial Basis Function (*RBF) Kernel has been used because it is the most popular and has nonlinear and effective features.

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2) \tag{15}$$

3.4 Multi-class support vector machines (Multi-class classifiers)

Support vector machines (SVMs) are primarily designed for two-class classification problem. However, the two-class classifier described above can be expanded to the multi-class classifier. There are two main approaches for this, i.e., one-against-all and one-against-one (Weston and Watkins, 1998; Salomon, 2001; Lee et al. 2001; Hsu and Lin, 2002).

The one-against-all approach construct n SVM models where n is the number of classes. The i^{th} classifier is trained with all of the examples in the i^{th} class with positive labels, and all other examples with negative labels. However, the one-against-one approach, which is a popular and simple technique, creates a SVM for all possible combinations of classes. This approach simply constructs all possible two-class classifiers from a training set of k classes. Each classifier is trained on only two out of k classes. Thus, there will be $n(n - 1)/2$ classifiers. The one-against-one approach has been used in many researches (Friedman, 1996; Weston and Watkins, 1998; Platt et al., 2000) and shown that its performance was better than the one-against-all approach (Hsu and Lin, 2002). In this research, the one-against-one approach has also been used as many other researches.

4. EXPERIMENTS AND RESULTS

4.1 Data Sets for Training and Test

For the comparison of the prediction performance of two models, this research considered

following three different image patterns, Pattern A, Pattern B, and Pattern C (see Figure 2). The Pattern A images were obtained by extracting the ROI (Region of Interest) corresponding to the top of a vehicle, Pattern B images were obtained by extracting the rear part of a vehicle, and Pattern C images were obtained by extracting non-vehicle images in the ROI. These patterns are similar to those obtained from car park image by Kim and Bell (1997). There were a lot of irrelevant objects, such as shadow, noise and adjacent objects, which should be differentiated from a vehicle image in real-world images. From the traffic scene, 930 data sets in total have been obtained, 400 sets for Pattern A, 400 sets for Pattern B and 130 sets for Pattern C. The total data sets were split into two subsets, – one for training and the other for testing as shown in the Table 1.

Table 1. Data Sets for Training and Testing

	Training data	Test data	Total
Pattern A	100	300	400
Pattern B	100	300	400
Pattern C	30	100	130
Total	230	700	930

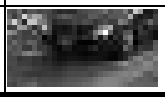
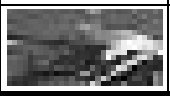
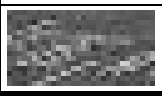
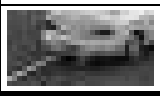
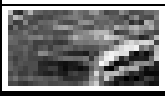
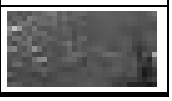
Input Images	Training data set			Test data set		
	Pattern A	*Pattern B*	*Pattern C*	*Pattern A*	*Pattern B*	*Pattern C*
256 gray, 15 by 30						

Figure 2. Three Patterns of Image Data for Training and Test

4.2 Normalization for Input Vectors

The normalization of input vectors is required for both SVMs and Neural Networks (Kim, 1999a; Kim, 2004). More importantly, the normalization method affects the prediction performance of the model. The normalization method used in this research is the following equation.

$$\tilde{a}_{pi} = \frac{a_{pi} - a_{\mathbf{p}\min}}{a_{\mathbf{p}\max} - a_{\mathbf{p}\min}} \tag{16}$$

where $a_{\mathbf{p}\max}$ and $a_{\mathbf{p}\min}$ are the maximum and minimum value of the input unit across all the input patterns, respectively, i.e.,

$a_{\mathbf{p}\max} = \max\{\mathbf{a}_1(a_1,...,a_m), \mathbf{a}_2(a_1,...,a_m),..., \mathbf{a}_P(a_1,...,a_m)\}$ and

$a_{\mathbf{p}\min} = \min\{\mathbf{a}_1(a_1,...,a_m), \mathbf{a}_2(a_1,...,a_m),..., \mathbf{a}_P(a_1,...,a_m)\}$

where $\mathbf{a}_1, \mathbf{a}_2,..., \mathbf{a}_P$ are the input patterns. The normalized value of the unit *i* of input vector, $\tilde{a}_{pi}$ are in the range [0,1].

4.3 Experimental Results

The inputs for two models, Backpropagation and SVMs, are normalized data with 256 gray

scale images of 15 by 30 pixels. For the neural network model, the *Backpropagation with Momentum & Prime-offset* (BPMP) model described in Section 2, which is one of advanced Backpropagation models, was used since it has been shown to possess better performance than other Backpropagation models such as the standard Backpropagation and Backpropagation with momentum (Kim, 2002). The implementation of this model was for batch mode learning with a learning rate of 0.1, a momentum of 0.95, and a prime-offset of 0.1. For the network topology, one-hidden-layer network has been used since the precious studies showed that one-hidden-layer network was enough for the pattern recognition problem as in this research (Lippmann, 1987; Wieland and Leighton, 1987; Arai, 1993; Kim, 1999b). The network topology used in this research is 450 - 225 - 3. For the output vectors, three units were used to recognize three different patterns; i.e. y1 = [1 0 0], y2 = [0 1 0], and y3 = [0 0 1]. All networks were fully interconnected, i.e. input layer to the hidden layer and the hidden layer to the output layer, and the sigmoid function was used for activation function. Training of the network will be stopped when 100% recognition accuracy is achieved on the training set.

For SVMs using RBF kernel, two parameters, C and γ, should be determined beforehand. The parameter of C which is a positive regularization parameter that controls the tradeoff between complexity of the machine and the allowed classification error, and γ is the parameter of the Gaussian kernel of Equation (15) determining the width of the kernel function. For the parameter C on the problem of this research, the prediction performance on the test sets was not different with various values between C = 1.0 to 1,000. This implies that the parameter C has not much influence on the performance of the SVM classifier. Accepting the results, all experiments have been done with C = 1.0 in this paper. For the parameter γ of the Gaussian RBF kernel, the results show the different performance according to the parameter value (see Table 2 and Figure 2). The Error Rate in Table 2 express the recognition error rate on the test sets (the number of error/the number of test sets). The best prediction performance was achieved when Gamma was a range from 0.0075 to 0.01.

One of the main issues in the application of SVMs is to determine the proper value of parameters before learning. A cross-validation could be one of efficient methods for estimating parameters. The parameters Gamma (γ) and C were set to 1 and 0.04 respectively using twofold cross-validation method in this paper. Optimal values for two parameters of Gamma (γ) and C, which were derived by using twofold cross-validation method, are 1 and 0.04 respectively. Table 3 shows the prediction performance on the test data set with two models, Backpropagation and SVMs. The prediction performance of Backpropagation model is sensitive to the initial weight configuration. With image data of this study, there were average 122.53/700 error rate -84/700 minimum error rate and 156/700 maximum error rate - during 30 trials with different initial weights. On the other hand, the SVMs give prediction error rate of 15/700. The results show that SVMs could give much better prediction performance than Backpropagation.

Table 2. Prediction Accuracy of SVM on Various Gamma Values

	Gamma(γ)								
	0.005	0.0075	0.01	0.02	0.03	0.04	0.05	0.06	0.07
Error Rate	171/700	149/700	52/700	52/700	43/700	29/700	24/700	16/700	15/700
Accuracy(%)	75.57	78.71	92.57	92.57	93.86	95.86	96.57	97.71	97.86

	Gamma(γ)								
	0.08	0.09	0.1	0.15	0.2	0.25	0.3	0.35	0.4
Error Rate	16/700	18/700	18/700	26/700	78/700	119/700	154/700	179/700	207/700
Accuracy(%)	97.71	97.43	97.43	96.29	88.86	83.00	78.00	74.43	70.43

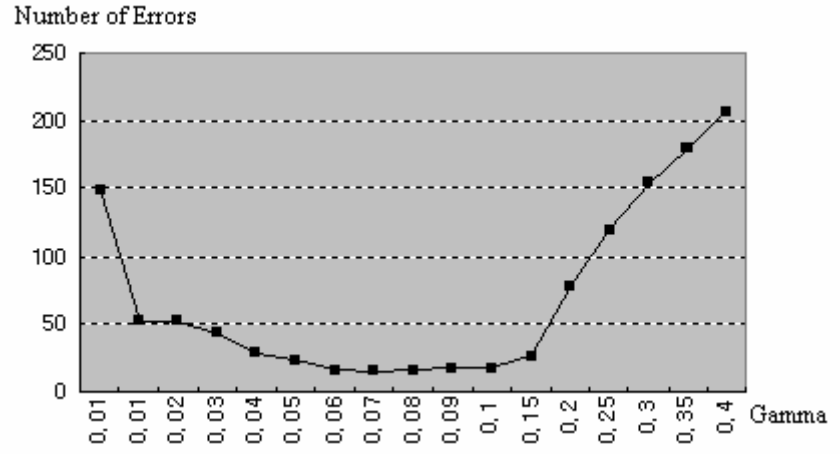


Figure 3. Prediction Errors on Various Gamma Values

Table 3. Prediction Performance of Backpropagation and SVMs on 256 Gray Images

	Backpropagation	SVMs
Average Prediction Errors Rate	122.53/700	15/700
Variance	342.40	-
Prediction accuracy (%)	82.50	97.86

5. CONCLUSION

This paper applied SVMs for traffic scene classification and the prediction performance of the model was compared with Backpropagation which is the most popular neural network model. The experimental results with real-world traffic images show that SVMs can provide higher performance in terms of prediction performance than the Backpropagation neural network model. The SVMs may be more efficient to classify complicate images with occlusion, shadow from other objects and noise problems which are inevitable in the real world image than conventional methods. Even though SVMs was used for vehicle detection in this paper, they also could be applicable to many other areas of transportation engineering and may no doubt provide much better performance than current approaches.

REFERENCES

Arai, M. (1993) Bounds on the number of hidden units in binary-valued three-layer neural networks, **Neural Networks**, **Vol. 6**, 855-860.

Balakrishnan, K. and Honavar, V. (1992) Improving convergence of backpropagation by handling flat-spots in the output layer. **Proceedings 2nd International Conference on Artificial Neural Networks**, 1003-1009.

Blanz,V., Schölkopf, B., Bülthoff, H., Burges, C., Vapnik, V. and Vetter, T.(1996) Comparison of view-based object recognition algorithms using realistic 3D models, In: von der Malsburg, C., von Seelen, W., Vorbrüggen, J. C. and Sendhoff, B. (eds.), **Artificial Neural Networks – ICANN'96**, Berlin, Springer Lecture Notes in Computer Science, **Vol. 1112**, 251-256

Cortes, C. and Vapnik, V. (1995) Support Vector Networks, **Machine Learning**, **Vol. 20**, 273-297.

Fahlman, S. E. (1988) An empirical study of learning speed in backpropagation networks. Technical Report CMU-CS-88-162, Canegie Mellon University.

Fausett, L. (1994) **Fundamentals of Neural Networks : architectures, algorithms, and applications**. Englewood Cliffs, NJ : Prentice-Hall.

Fletcher R. (1987) **Practical Methods of Optimization**. John Wiley and Sons

Friedman, J.(1996) Another approach to polychotomous classification, Technical report, Department of Statistics, Stanford University.

Hsu, C.-W. and Lin, C.-J. (2002) A comparison of methods for multi-class support vector machines, **IEEE Transactions on Neural Networks, Vol. 13**, 415-425.

Joachims, T. (1998) Text Categorization with Support Vector Machine: Learning with Many Relevant Features. **Proceedings of ECML-98, 10th European Conference on Machine Learning**, 137-142,.

Kim, D. (1999a) Normalization Methods for Input and Output Vectors in Backpropagation Neural Networks, **International Journal of Computer Mathematics**, **Vol. 71, No. 2**, 161-171.

Kim, D. (1999b) Problems Encountered During Implementation of the Backpropagation, **ITS Journal, Vol. 5, No. 1**, 71-86.

Kim, D. (2002) Standard and Advanced Backpropagation Models for Image Processing Application in Traffic Engineering, **ITS Journal, Vol. 7, No. (3-4)**, 199-211.

Kim, D. (2004) Prediction Performance of Support Vector Machines on Input Vector Normalization Methods, **International Journal of Computer Mathematics, Vol. 81, No. 5**, 547-554.

Kim, D. and Bell, M.G.H. (1997) Parking Vacancy Detection using Image Processing and Neural Networks, **Proceedings of 4th ITS world congress**, Berlin, Germany.

Lee, Y., Lin, Y. and Wahba, G.. (2001) Multicategory support vector machines, Technical report, University of Wisconsin.

Lippmann, R. P. (1987) An introduction to computing with neural nets, **IEEE Acostics, Speech and Signal Processing Magazine, Vol. 4, No. 2**, 4-22.

Osuna, E, Freund, R. and Girosi, F. (1997) Training Support Vector Machines: an Application to Face Detection, **IEEE Conference on Computer Vision and Pattern Recognition**, 130-136.

Platt, J., Cristianini, N. and Shawe-Taylor, J. (2000) Large margin DAGs for multiclass classification, **Advances in Neural Information Processing Systems, Vol. 12**, 547–553.

Rumelhart, D. E., McClelland, J. L. and the PDP Research Group (eds.) (1986) **Parallel Distributed Processing: Explorations in the Microstructure of Cognition** (Vol. 1 & 2), Cambridge, Massachusetts: MIT Press.

Salomon, J., (2001) Support vector machines for phoneme classification, M.Sc. Dissertation,

School of Artificial Intelligence, University of Edinburgh.
Schölkopf, B., Burges, C. and Vapnik, V. (1995) Extracting Support Data for a Given Task, In: U. M. Fayyad and R. Uthurusamy(eds.), **Proceedings of 1st International Conference on Knowledge Discovery & Data Mining**, Menlo Park, AAAI Press.
Schölkopf, B., Burges, C. and Vapnik, V. (1996) Incorporating Invariances in Support Vector Learning Machines", In: C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff (eds.), **Artificial Neural Networks –ICANN'96**, Berlin, Springer Lecture Notes in Computer Science, **Vol. 1112**, 47-52.
Schölkopf, B., Burges, C., and Smola, A. (1999) **Advances in Kernel Methods - Support Vector Learning**, Cambridge, MA, MIT Press.
Vapnik, V., (1995) **The Nature of Statistical Learning Theory**, Springer Verlag, New York.
Weston, J. and Watkins, C. (1998) Multi-class support vector machines, Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, UK.
Wieland, A. and Leighton, R. (1987) Geometric analysis of neural network Capabilities**. Proceedings 1st IEEE International Conference on Neural Networks, Vol. 3**, 385-392.