Two-stage License Plate Detection using Gentle Adaboost and SIFT-SVM

Wing Teng Ho, Hao Wooi Lim, and Yong Haur Tay

Computer Vision & Intelligent Systems (CVIS) group,
Universiti Tunku Abdul Rahman (UTAR).
9, Jalan Bersatu 13/4, 46200 Petaling Jaya, Selangor, Malaysia.
email: wingtengh@gmail.com, zybler+rnd@gmail.com, tayyh@utar.edu.my

Abstract

This paper presents a two-stage method to detect license plates in real world images. To do license plate detection (LPD), an initial set of possible license plate character regions are first obtained by the first stage classifier and then passed to the second stage classifier to reject non-character regions. 36 Adaboost classifiers (each trained with one alpha-numerical character, i.e. A..Z, 0..9) serve as the first stage classifier. In the second stage, a support vector machine (SVM) trained on scale-invariant feature transform (SIFT) descriptors obtained from training sub-windows were employed. A recall rate of 0.920792 and precision rate of 0.90185 was obtained.

Keywords: *License plate detection, SIFT, SVM, Adaboost.*

1. Introduction

Vehicle license plate detection (LPD) is an important step in any automated vehicle verification system. It is used to search for license plate portion on an image before verification of the license plate can be done. Failure is imminent if the license plate region is not detected or wrongly found.

There are many applications for vehicle verification system, such as stolen vehicles detection [10], driver navigation support [10], automated parking attendant [6], border crossing control [6], petrol station forecourt surveillance [6], personalized service via customer identification [6], automated toll ticketing [14] and etc. There are many constrains and difficulties in LPD when it comes to dealing with unconstrained environment such as scale, rotation, affine transformation, illumination, occlusion, translation; shearing, distortion and skew. In this paper, an approach that combined Adaboost and Support Vector Machine (SVM) in LPD is presented. Our framework will use Adaboost to do the character detection, and SVM is used to filter the false positive.

2. Background

Adaboost has been a successful approach for face detection that minimizes the computational time and obtaining high detection rate. Viola et al. [15] presented a framework for face detection that achieves high detection rate rapidly. Motivated by [5], they introduced an image representation known as the integral image that allows the features used in the detection to be calculated in many scales or locations in constant time. They applied Haar-like features [15] to classify the patterns for an image, which are reminiscent of Haar wavelet used by Papageorgiou and Poggio [4][5]. Then, they used AdaBoost learning algorithm to select a simple Haar-like features from the over-complete features set.

Chen and Yuille [20] demonstrated an algorithm for detecting text in natural images, also based on AdaBoost. They claimed that the set of features used for face detection by Viola and Jones [15] might not be suitable for detecting text. This is because there is less spatial similarity for the text compare to face; a face can be regarded as spatial similar object since it consists of facial features such as eyes, nose, and mouth that are approximately the same spatial position for any face.

Some of the algorithms are designed specifically for detecting character such as the adaptive algorithm for text detection from natural scenes by Gao and Yang [8]. They developed a prototype system that can recognize Chinese sign inputs. Shapiro et al. [18] have introduced image-based vehicle license plate recognition (CLPR) system. The system consists of few processes such as the license plate localization that is used to locate the license plate for an image. In [7], they proposed a LPD method that implements a global statistical feature in the first two layers while the rest of the layers are the local features selected by the Adaboost from the haar-like feature set. Their final classifier is invariant to the brightness, color, and size of license plates with 93.5% of detection rate and very low false positive rate [7].

In [19], they implemented Gentle Adaboost trained with Haar-like features on edge license plates. They achieved true positive rate of 0.76 and false alarm rate of 1.0, but it is not robust to extreme rotated license plate regions (about 30 degrees onwards). Proposed by [9], a process of line filtering and line projection is used to remove non-license plate pixels. Even though achieving 93% non-license plate lines removal, the method is only useable in constrained environment, because it assumes the region of character is within a limited range of size. Any attempt to make it more robust, the algorithm will not be able to remove non-license plate lines well.

3. System Architecture

Our license plate detector contains two stages, Adaboost combined classifier and SVM filter. Initially, we will pass in grayscale image to Adaboost Combined Classifier for character detection. After that, we obtained some detected windows from the character detection, and all the detected windows will be passed to SVM filter in second stage to remove all the non-character. Finally, we will have the final result after filtering the false images. The system architecture is shown in figure. 1.

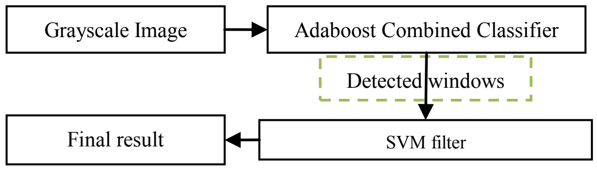

Figure 1. Figure shows out LPD system architecture.

4. Algorithms Applied

4.1. Haar-like Features

In [15], Viola et al. proposed to use simple Haar-like basis functions as features. These Haar-like basis functions are simple 2D wavelet constructs with at least two non-overlapping rectangular regions, represented as white or black. Feature can be calculated from the subtraction of pixels summation within the black region the white region, respectively. In our work, three types of features were used, these features were originally proposed by Viola et al. as shown in figure. 2 and these features are further discussed in [16]. These simple features consist of significant domain-knowledge information, and the detector operates much faster with features compare to pixels. In our character detection stage, we implemented these features and use Adaboost training algorithm to select the feasible features that can be used in character detection.

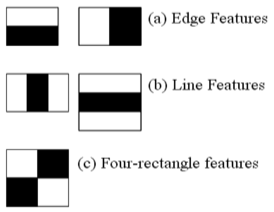

Figure 2. Figure shows Haar-like basis used as features in character detection. [15]

4.2 Adaboost Boosting Algorithm

Y. Freund et al. introduced a machine learning method called Adaboost, short for ***ada****ptive* ***boost****ing* [21]. This technique increases classification performance of a simple algorithm (for e.g. a simple perceptron) by combining weak classifiers to form one strong classifier. A weak classifier is any classifier that delivers performance slightly better than chance and preferably fast to compute. They discovered that a *committee* of weak classifiers when combined properly often outperforms strong classifiers such as Support Vector Machines (SVM) and Neural Networks [21]. Boosting algorithm comes with many variants such as Discrete Adaboost, Real Adaboost, and Gentle Adaboost [1].

In our character detection system, Gentle Adaboost is chosen because it outperforms other variants in an empirical analysis by A. Kuranov et al. [1]. Note that, boosting algorithms only differ in the procedure on how to re-weight training examples after the training iteration. To boost the performance of a strong classifier, Adaboost algorithm search over a pool of weak classifiers to find one with the lowest classification error for the subsequent combination. In each training iteration, training samples that were incorrectly classified were highlighted (re-weighted) so that in the next iteration more focus is on them. After training, we have a weighted combination of weak classifiers in the form of perceptrons and a simple binary threshold value determined automatically. Every weak classifier has an associated weight where better classifiers were assigned larger weight while poorer classifiers were given smaller weight. Figure. 3 shows the learning algorithm [15].

```
• Given example image $(x_1, y_1), \dots, (x_n, y_n)$, where
  $y_i = 0,1$ for negative and positive examples respectively.
• Initialise weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0,1$
  respectively, where $m$ and $l$ are the number of negative
  and positive respectively.
• For $t = ,\dots,T$ :
    1. For each feature, $j$, choose a classifier, $h_t$ by
       minimizing a weighted squared error
       $\varepsilon_j = \sum_i w_i |h_j(x_i) - y_i|$
    2. Update the classifier $F(x) \leftarrow (x) + {}_t(x)$
• Update weights by $w_i \leftarrow e^{-{}_i h_t(x_i)}$
• Final strong classifier is $F(x) = \mathrm{sgn}\left(\sum_{i=1}^{T} \alpha \bullet h_i(x) + b\right)$
```

Figure 3. Figure shows Gentle Adaboost procedure to construct a strong classifier. *T* is the number of weak classifiers. The final strong classifier is a weighted linear combination of the *T* weak classifiers $h_i(x)$ with biased offset *b* ([15], [16], [19]).

4.3 Character Detection with Adaboost

Our character detector runs detection window across the image at multiple scales and locations. Detection window is scaled at 1.2, indicates that window size increased at 20% rate between following scans, starting with minimum size of 20 ×28 (width x height). The features can be easily determined by scaling the base window features by current scale factor, this operation can be done at any scales with the same cost.

The output of the final detection contains multiple license plates characters detected around each license plate since LPD is insensitive to small translations and/or scale changes. For all the non-license plates' characters that are detected can be categorized as false positive. In our system, during each process of scanning the image, we obtain some false positives; In order to reduce such false positive, we implement the SVM to remove the false positive subsequently.

4.4 Scale-Invariant Feature Transform (SIFT) and SIFT Descriptor

SIFT is a location-histogram-based feature that is invariant to image scale, rotation, affine distortion, changes in 3D viewpoint, noise and changes of illumination. SIFT has proved itself to be excellent for image matching [22]. However, recent researches have shown that it is also good for object recognition [23]. Candidate keypoints are identified by detecting local extrema in the difference-of-Gaussian function convolved with the image in scale space. These keypoints usually contain distinctive information for matching. Being an extreme, its location is less susceptible to minor variations (rotation, affine transformation, etc) of its surrounding. Once a keypoint is found, local descriptor is extracted from its small neighbourhood. To do so, a local descriptor is divided into K smaller regions, with each being described by a histogram of sized Q of image gradients in the area. This local descriptor is invariant to small changes in the keypoint and brightness of its surrounding. In this paper, we choose K to be 4 and Q to be 8.

4.5 Support Vector Machine (SVM)

This paper uses SVM as a classifier ([3], [13]), which is provided by the library LIBSVM [2]. SVM is a linear perceptron that solves binary classification problem. Some supervised learning algorithm is usually applied to find an optimal N-dimensional hyperplane. The decision function of a SVM is defined as,

$$f(x) = sgn \sum_{i \in S} y_i \quad \propto_i \ K(x, s_i) + b \qquad (1)$$

where x is the input to be classified into y ∈{−1, +1}and S = $\{i|\alpha_i \in \mathbb{R}^+\}$ is a set of positive coefficients. Support vectors s_i, i ∈ S are a subset of the training vector extracted during the optimization process. K is a user-chosen kernel function.

In reality, problems are seldom linearly separable. Thus, a non-linear kernel function that meets the Mercer's theorem [17] is needed to map input space to higher dimensions so that one can solve a non-linear problem using linear classification. In this paper, radial basis kernel is used. It is defined as,

$$K(x, z) = \exp(-\gamma \|x - z\|^2) \qquad (2)$$

Given training vectors xi ∈ $\mathbb{R}^n$, i =1, ..., *l* such that y_i ∈ {−1, +1}, the following primal problem must be minimized [17],

$$Q(w, \xi_i, b, p) = \frac{1}{2} w^T w - vp + \frac{1}{l} \sum_{i=1}^{l} \xi_i \qquad (3)$$

subject to $y_i(w^T \varphi(xi)+b) \geq 1-\xi_i$, $\xi_i \geq 0$(i =1, ..., l), ρ ≥ 0
Its dual formulation is defined as,

$$Q(\propto) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{1} \propto_i \propto_j y_i y_j K(x_i x_j) \qquad (4)$$

subject to $0 \le \alpha_i \le 1 (i = 1, ..., l)$, $e^T \alpha \ge \nu l$, $y^T \alpha = 0$
$\nu \in (0, 1]$ is a user specified parameter to control the number of support vectors and training errors. It is an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors.

4.5 False Positive Removal with SVM

The purpose of using an SVM in the second stage is to remove regions that are wrongly deemed to be a character region by Adaboost classifier in the first stage. To use SIFT with SVM to remove false positives, a set of keypoints are obtained from the set of all negative samples, and SVM is trained on all on key points' 128 SIFT descriptors. During detection, SIFT features are obtained from a particular sub-window and each SIFT feature is passed to an SVM to obtained a probability of it being a character region. Final probability is calculated for the sub-window by summing the probability (for both classes) for each key point and the final probability divided by the number of key points in that sub-window so that probabilities from both classes summed to 1. If the probability of it being a non-character region is higher than the probability of it being a character region, it is rejected. A bias value is used to make it more likely to give a positive output or a negative output. A positive bias makes it harder to detect a character region while a negative bias makes it harder to detect a non-character region. The equation is shown below where *Prob(char)* represents the probability of it is a character region and Prob(nchar) represents the probability of it being a non-character region:

$$resultPos = \begin{cases} 1 & where\ Prob(char) - bias > Prob(nchar) + bias \\ 0 & otherwise \end{cases} \qquad (5)$$

5. Experiment Settings

For first stage classifier, training dataset consists of 880 positive samples and 800 negative samples. The positive samples are obtained from an artificial character generator. It consists of various capital letters and digits with different font.

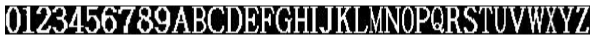

Figure 4. Positive sample used to train 36 characters classifiers.

Figure 5. Negative sample used to train 36 characters classifiers.

For second stage classifier, training dataset is a list of all 128 SIFT descriptors obtained from 46 regions. These regions were obtained by running the first stage classifier on a training real-world images (97 of them), taken with a 3 megapixel camera phone on close proximity the front/back view of vehicles. Each sample is labeled with location and size of one or more license plate character region(s) on the image. This data is represented by min_x, max_x, min_y and max_y. Each labeled license plate character region is considered detected if there is a smaller region (bigger than a certain threshold) resides within the labeled license plate bounding box with a tolerance of 22 pixels.

In this paper, we quantify the results using recall rate and precision rate. Recall rate measures how good the classifier able to classify a labeled positive license plate region as positive license plate. It is defined as,

$$\text{Recall rate} = \frac{\text{Number of correctly labeled positive regions}}{\text{Number of positive regions}} \qquad (6)$$

Precision rate measures how much percentage of the regions classified as positive license plate is labeled as a positive license plate. It is defined as,

$$\text{Precision rate} = \frac{\text{Number of correctly labeled positive regions}}{\text{Number of regions labeled as positive}} \qquad (7)$$

6. Results & Discussion

In this paper, we used radial basis function (RBF) as the kernel function in SVM. While using RBF kernel, we need to identify two parameters: C and gamma. In the first experiment, model selection is performed to search for the best C and gamma by doing grid-search using cross-validation [24]. The best result is observed when C is 32 and gamma is 0.5 where the cross-validation accuracy is 88% on training dataset as shown in figure 6.

In the second experiment, we tested on scaling the SVM's input range by normalizing feature values from SIFT. From the experiment with different scale range, we compare the hit rate of a SVM trained with the best C and gamma parameter, and a SVM trained with default values.

In the third experiment, different scale factor is tested and its effect on recall rate and precision rate is noted, as shown in figure. 7. This scale factor is used for resizing the detection windows during Adaboost Character detection.

The forth experiment is to find the optimum bias referring to formula (5), we tested on the bias of -0.5 till 0.4. The result is shown in figure. 8. From the figure, we can see that with the bias value of 0, we able to obtain optimal recall rate and precision rate.

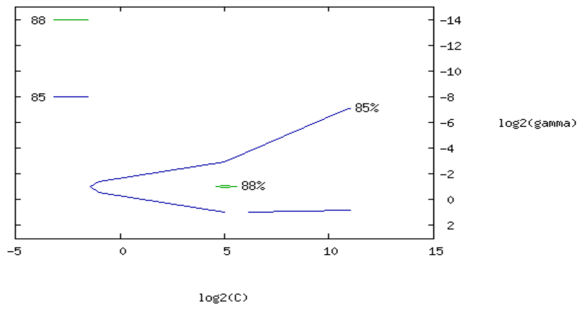

Figure 6. Graph showing the cross-validation accuracy on different C and gamma.

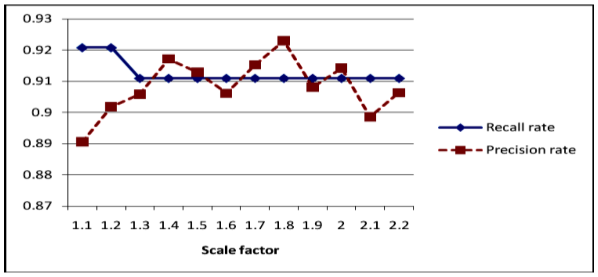

Figure 7. Figure shows the effect of varying scale factor on recall and precision rate.

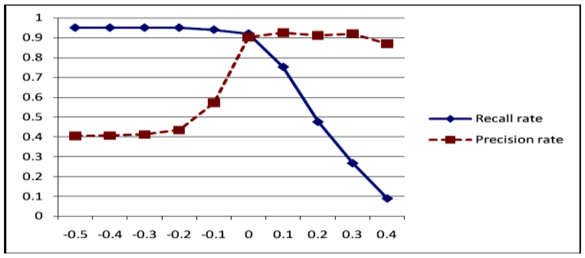

Figure 8. Figure shows the effect of varying bias on recall and precision rate.

From our observation, we found that characters smaller than 20x28 failed to be detected by our first stage detector because our training samples have a minimum resolution of 20x28. And, since 36 classifiers were used in the first stage detector, detection speed is not fast enough (about 5 seconds). Also, since the ratio of the scanning sub-window is fixed, there may be problems when dealing with characters that are too narrow or too wide. Finally, due to the nature of the windows scanning method, characters that are too rotated cannot be detected. For the second stage classifier, the scaling range for the SIFT descriptors turned out to be important. Scaling it to between 0 and 1 will cause serious information loss, decreasing the testing accuracy, as shown in Table 1.

Finally, table 2 shows the comparison between the Adaboost and two-stage classifier. The results show the system able to remove more false detection by using the two-stage classifier.

Table 1. Table shows results of varying scaling range of SVM's input on SVM trained with best parameters and SVM trained on default parameters.

Best SVM		Default SVM	
Scale range	Hit rate	Scale range	Hit rate
[0 1]	33.6549%	[0 1]	66.0235%
[-1 1]	66.3451%	[-1 1]	55.9670%
[-2 2]	66.3451%	[-2 2]	33.6549%

Table 2. Comparison Between Adaboost and 2-stage Adaboost+SVM.

	Adaboost	Adaboost+(SIFT+SVM)
Recall	0.950495	0.920792
Precision	0.350749	0.90185

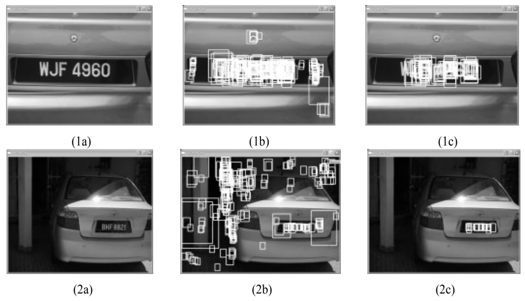

Figure 9 Figure shows (1a), (2a) shows the original image; (1b), (2b) shows result after character detection by first stage detector; (1c), (2c) shows result after filtering by second stage classifier.

7. Conclusion & Future Works

In this paper, we have proposed a way to do LPD using a character detector trained on license plate characters using Adaboost and an SVM trained SIFT descriptors to remove falsely detected sub-windows from the previous stage. Experiments have shown that the Adaboost is able to detect almost all license plate characters but false positive is high. These turns out to be easily filtered by an SVM trained on SIFT.

Since the grouping of detected license plate characters to form a license plate is beyond the scope of this paper (we locate all license plate characters, using 2 different viewpoints), perhaps our next step is to build a clustering mechanism that groups detected license plate characters to form a license plate, likely based on domain knowledge as prior.

For other future work, perhaps a single Adaboost classifier that train on all 36 alphanumeric alphabets instead of requiring 36 separate Adaboost classifier can be employed to decrease run time. To do so, better features than Haar-like feature may be required. Also, instead of training on raw SIFT features, some kind of features such as statistical information may be extracted from the few SIFT key point descriptors of a given sub-window for classification.

8. Acknowledgements

This research is partly funded by Malaysian MOSTI ScienceFund SF-01-02-11-SF0019.

9. References

[1] A. Kuranov, R. Lienhart, et al., "Empirical analysis of detection cascades of boosted classifiers for rapid object detection", *Lecture Notes in Computer Science*, Springer-Verlag, Heidelberg, 2003, pp. 294–304.

[2] C.C. Chang, C.J Lin, "LIBSVM: a library for Support Vector Machines", 2001, Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm

[3] C.J. Burges, "A tutorial on Support Vector Machines for Pattern Recognition", *Data Mining and Knowledge Discovery*, Vol. 2, no. 2, 1998, pp. 121–167.

[4] C.P. Papageorgiou, T. Poggio, "A Trainable Object Detection System: Car Detection in Static Image", *A.I. Memo*, No 1673, C.B.C.L Paper No 180. Massachusetts Institute of Technology, 1999.

[5] C.P. Papageorgiou, T. Poggio, "A Trainable System for Object Detection", *International Journal of Computer Vision 38(0)*, 2000, pp. 15-33.

[6] D.G. Bailey, D. Irecki, B.K. Lim, L. Yang, "Test bed for number plate recognition applications", *1th IEEE International Workshop on Electronic Design, Test and Applications*, 2002, pp. 501–503.

[7] H. Zhang, W. Jia, X. He, Q. Wu, "Learning-Based License Plate Detection Using Global and Local Features", *Proc. the 18th International Conference on Pattern Recognition (ICPR'06)*, Hong Kong, 2006, pp. 909–912.

[8] J. Gao, J. Yang, "An Adaptive Algorithm for Text Detection from Natural Scenes", *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001.

[9] J.C.M. Lee, "Automatic Character Recognition for Moving and Stationary Vehicles and Containers in Real-life images", *International Joint Conference on Neural Networks*, Vol. 4, 1999, pp. 2824–2828.

[10] J. Matas, K. Zimmermann, "Unconstrained Licence Plate and Text Localization and Recognition", *IEEE Intelligent transportation Systems*. 2005, pp. 225–230.

[11] J.R. Parker, *Algorithms for Image Processing and Computer Vision*, Wiley Computer Publishing, John Wiley & Sons, Inc, Professional, Reference and Trade Group, United States of America, 1997.

[12] M. Nixon, A. Aguado, *Feature Extraction & Image Processing (1st Ed.)*, British Library Cataloguing in Publishing Data, 2002.

[13] N. Cristianini, J.S. Taylor, *Support Vector Machines*, Cambridge University Press, Cambridge, 2002.

[14] P. Castello, C. Coelho, E. Del Ninno, E. Ottaviani, M. Zanini, "Traffic monitoring in motorways by real-time number platerecognition", *International Conference on Image Analysis and Processing* 1999, pp. 1128–1131.

[15] P. Viola, M. Jones, "Robust Real-Time Face Detection. International Journal of Computer Vision", 57(2), 2004, 137-154.

[16] R. Lienhart, A. Kuranov, V. Pisarevsky, *Empirical Analysis of Detection Cascade of Boosted Classifiers for Rapid Objects Detection*, MRL Technical Report, Intel Corporation, Santa Clara, USA, 2002.

[17] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, 1995.

[18] V. Shapiro, G. Gluhchev, D. Dimov, "Towards a Multinational Car License Plate Recognition System", *Machine Vision and Application*, 17, 2006, 173-183.

[19] W.T. Ho, W.H. Yap, Y.H. Tay, "Learning-based License Plate Detection on Edge Features", *Proceedings of the 10th MMU International Symposium on Information and Communications Technologies*, 2007.

[20] X.R. Chen, A.L. Yuille, "Detecting and Reading Text in Natural Scenes", *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

[21] Y. Freund, R.E. Schapire, "A decision-theoretic generalization of on-line learning and application to boosting", *Computational Learning Theory: Eurocolt'95*, Springer-Verlag, 1995, pp. 23–37.

[22] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", *International Journal of Computer Vision*, Vol. 20, 2004, pp. 91-110.

[23] P. Moreno, M.J. Marin-Jimenez, A. Bernadino, J. Santos-Victor, N.P. Blanca, "A comparative study of local descriptors for object category recognition: SIFT vs HMAX", *Proceedings of Pattern Recognition and Image Analysis, Third Iberian Conference, IbPRIA, Girona, Spain*, 2007, pp. 515-522.

[24] C. Hsu, C. Chang, C. Lin, *A Practical Guide to Support Vector Machine*, Department of Computer Science and Information Engineering National Taiwan University, Taipei 106, Taiwan, 2008.