International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems
Vol. 0, No. 0 (1993) 000—000
© World Scientific Publishing Company

# SUPPORT VECTOR MACHINES FOR THAI PHONEME RECOGNITION

NUTTAKORN THUBTHONG, and BOONSERM KIJSIRIKUL
*Machine Intelligence & Knowledge Discovery Laboratory*
*Department of Computer Engineering, Chulalongkorn University*
*Bangkok, 10330, Thailand*

Received March 2001
Revised October 2001

The Support Vector Machine (SVM) has recently been introduced as a new pattern classification technique. It learns the boundary regions between samples belonging to two classes by mapping the input samples into a high dimensional space, and seeking a separating hyperplane in this space. This paper describes an application of SVMs to two phoneme recognition problems: 5 Thai tones, and 12 Thai vowels spoken in isolation. The best results on tone recognition are 96.09% and 90.57% for the inside test and outside test, respectively, and on vowel recognition are 95.51% and 87.08% for the inside test and outside test, respectively.

*Keywords*: Support vector machine; Vowel recognition; Tone recognition; Thai phoneme.

## 1. Introduction

The Support Vector Machine (SVM) is a new promising pattern classification technique[1] which is based on the principle of the structural risk minimization. Unlike traditional methods which minimize the empirical training error, the SVM aims to minimize the upper bound of the generalization error through maximizing the margin between the separating hyperplane and data. SVMs learn the boundary regions between samples belonging to two classes by mapping the input samples into a high dimensional space, and seeking a separating hyperplane in this space[2]. The separating hyperplane is chosen in such a way that its distance is maximized from the closest training samples.

In recent years, SVMs have been used in many applications from the vision problem to text classification[3,4]. They have been shown to provide higher performance than traditional techniques, such as neural networks. However, their application to speech recognition problems has been very limited (for example, see[2,5,6,7]).

In this paper, we investigate the application of SVMs for two problems of phoneme recognition, i.e. Thai tone recognition and Thai vowel recognition. We run experiments to compare a number of techniques for *multi-class* SVMs as well as multi-layer perceptron (MLP).

This paper is organized as follows. Section 2 briefly introduces the concept of SVMs. In Section 3, we run experiments using SVMs and MLP for Thai tone recognition and Thai vowel recognition. The conclusion is given in Section 4.

## 2. Support Vector Machines

This section will introduce the basic idea of SVMs and a number of techniques for constructing multi-class SVMs.

### 2.1. *Linear support vector machines*

Suppose we have a data set $D$ of $l$ samples in an $n$-dimensional space belonging to two different classes ($+1$ and $-1$):

$$D = \{(\mathbf{x}_k, y_k) \mid k \in \{1, \ldots, l\}, \mathbf{x}_k \in \Re^n, y_k \in \{+1, -1\}\}. \tag{1}$$

The hyperplane in the $n$ dimensional space is determined by the pair $(\mathbf{w}, b)$ where $\mathbf{w}$ is an $n$-dimensional vector orthogonal to the hyperplane and $b$ is the offset constant. The hyperplane $(\mathbf{w} \cdot x) + b$ separates the data if and only if

$$\begin{array}{ll} (\mathbf{w} \cdot \mathbf{x}_i) + b > 0 & \text{if } y_i = +1 \\ (\mathbf{w} \cdot \mathbf{x}_i) + b < 0 & \text{if } y_i = -1. \end{array} \tag{2}$$

If we additionally require that $\mathbf{w}$ and $b$ be such that the point closest to the hyperplane has a distance of $1/|\mathbf{w}|$, then we have

$$\begin{array}{ll} (\mathbf{w} \cdot \mathbf{x}_i) + b \geq +1 & \text{if } y_i = +1 \\ (\mathbf{w} \cdot \mathbf{x}_i) + b \leq -1 & \text{if } y_i = -1 \end{array} \tag{3}$$

which is equivalent to

$$y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1, \forall i. \tag{4}$$

To find the optimal separating hyperplane, we have to find the hyperplane that maximizes the minimum distance between the hyperplane and any sample of training data. The distance between two closest samples from different classes is

$$d(\mathbf{w}, b) = \min_{\{\mathbf{x}_i \mid y_i = 1\}} \frac{(\mathbf{w} \cdot \mathbf{x}_i) + b}{\mid \mathbf{w} \mid} - \max_{\{\mathbf{x}_i \mid y_i = -1\}} \frac{(\mathbf{w} \cdot \mathbf{x}_i) + b}{\mid \mathbf{w} \mid}. \tag{5}$$

From (3), we can see that the appropriate minimum and maximum values are $\pm 1$. Therefore, we need to maximize

$$d(\mathbf{w}, b) = \frac{1}{\mid \mathbf{w} \mid} - \frac{-1}{\mid \mathbf{w} \mid} = \frac{2}{\mid \mathbf{w} \mid}. \tag{6}$$

Therefore, the problem is equivalent to:

- minimize $\frac{|\mathbf{w}|}{2}^2$
- subject to the constrains:

(1) $y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1, \forall i.$

For non-separable case, the training data cannot be separated by a hyperplane without error. The previous constraints then must be modified. A penalty term consisting of the sum of deviations $\xi_i$ from the boundary is added to the minimization problem. Now, the problem is to

- minimize $\frac{|\mathbf{w}|}{2}^2 + C\sum_{i=1}^{l} \xi_i$
- subject to the constraints:
  (1) $y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1 - \xi_i,$
  (2) $\xi_i \geq 0, \forall i.$

The penalty term for misclassifying training samples is weighted by a constant $C$. Selecting a large value of $C$ puts a high price on deviations and increases computation by effecting a more exhaustive search for ways to minimize the number of misclassified samples.

By forming the Lagrangian and solving the dual problem, this problem can be translated into:

- minimize

$$L(\mathbf{w}, b, \alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j). \tag{7}$$

- subject to the constraints:
  (1) $0 \leq \alpha_i \leq C, \forall i.$
  (2) $\sum_{i=1}^{l} \alpha_i y_i = 0$

where $\alpha_i$ are called Lagrange multipliers. There is one Lagrange multiplier for each training sample. In the solution, those samples for which $\alpha_i > 0$ are called *support vectors*, and are ones such that the equality in (4) holds. All other training samples having $\alpha_i = 0$ could be removed from the training set without affecting the final hyperplane.

Let $\alpha^0$, an $l$-dimensional vector denote the minimum of $L(\mathbf{w}, b, \alpha)$. If $\alpha_i^0 > 0$, then $\mathbf{x}_i$ is a *support vector*. The optimal separating hyperplane $(\mathbf{w}^0, b^0)$ can be written in terms of $\alpha^0$ and the training data, specifically in terms of the support vectors:

$$\mathbf{w}^0 = \sum_{i=1}^{l} \alpha_i^0 y_i \mathbf{x}_i = \sum_{\texttt{support vector}} \alpha_i^0 y_i \mathbf{x}_i. \tag{8}$$

$$b^0 = 1 - \mathbf{w} \cdot \mathbf{x}_i \text{ for } \mathbf{x}_i \text{ with } y_i = 1 \text{ and } 0 < \alpha_i < C. \tag{9}$$

The optimal separating hyperplane classifies points according to the sign of $f(x)$,

$$f(\mathbf{x}) = \texttt{sign}(\mathbf{w}^0 \cdot \mathbf{x} + b^0) = \texttt{sign}[\sum_{\texttt{support vector}} \alpha_i^0 y_i (\mathbf{x}_i \cdot \mathbf{x}) + b^0]. \tag{10}$$

### 2.2. *Non-linear support vector machines*

The above algorithm is limited to linear separating hyperplanes. SVMs get around this problem by mapping the sample points into a higher dimensional space using a non-linear mapping chosen in advance. This is, we choose a map $\Phi : \Re^n \mapsto H$ where the dimensionality of $H$ is greater than $n$. We then seek a separating hyperplane in the higher dimensional space; this is equivalent to a non-linear separating surface in $\Re^n$.

The data only ever appears in our training problem (7) in the form of dot products, so in the higher dimensional space we are only dealing with the data in the form $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. If the dimensionality of $H$ is very large, then this could be difficult, or very computationally expensive. However, if we have a kernel function such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, then we can use this in place of $\mathbf{x}_i \cdot \mathbf{x}_j$ everywhere in the optimization problem, and never need to know explicitly what $\Phi$ is. Some widely used kernels are:
Polynomial degree $d$: $K(\mathbf{x}, \mathbf{y}) = \mid \mathbf{x} \cdot \mathbf{y} + 1 \mid^d$
Radial basis function (RBF): $K(\mathbf{x}, \mathbf{y}) = e^{-|\mathbf{x}-\mathbf{y}|^2/c}$

For more information on this see[1,4].

### 2.3. *Multi-class classifiers*

Now, we discuss the multi-class ($k$-class) problem solving by considering the problem as a collection of binary classification problems. In this paper, two approaches are considered, i.e. one-against-the-rest and one-against-one approaches.

#### 2.3.1. *The one-against-the-rest approach*[1]

This approach works by constructing a set of $k$ binary classifiers. 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. The final output is the class that corresponds to the classifier with the highest output value. We refer to this approach as *one-vs-all.*

#### 2.3.2. *The one-against-one approach*

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 $k(k-1)/2$ classifiers. This approach is referred to as *one-vs-one.* To evaluate the final output, we employ two algorithms.

(i) Max Wins algorithm[8]
A test example is classified by all of classifiers. Each classifier provides one vote for its preferred class and the majority vote is used to make the final output. However, if there are more than one class giving the highest score, a class will be randomly selected as the final output.

(ii) Decision Directed Acyclic Graph (DDAG)

The Decision Directed Acyclic Graph[9] (DDAG) is implemented using a rooted binary DAG* with $k$ leaves labelled by the classes where each of the $k(k-1)/2$ internal nodes is labelled with an element of boolean function. The nodes are arranged in a triangle with the single root node at the top, two nodes in the second layer and so on until the final layer of $k$ leaves. The $i^{th}$ node in layer $j < k$ is connected to the $i^{th}$ and $(i+1)^{st}$ nodes in the $(j+1)^{st}$ layer.

To evaluate a DDAG, starting at the root node, the binary function at a node is evaluated. The node is then exited via the left edge, if the binary function is -1; or the right edge, if the binary function is 1. The next node's binary function is then evaluated. The value of the decision function is the value associated with the final leaf node (see Fig. 1). Only $k-1$ decision nodes will be evaluated in order to derive an answer.

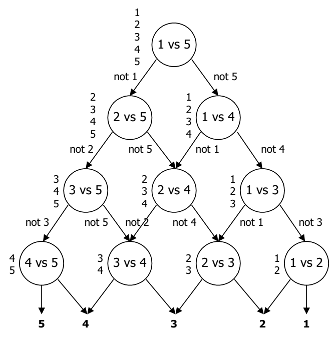

Fig. 1. The decision DAG for finding the best class out of five classes.

## 3. Experiments

In this section, we first describe the speech data used in the experiments. We then show two experiments, i.e. Thai tone recognition, and Thai vowel recognition, and report the results comparing SVMs using one-vs-one, Max Wins and DDAG.

### 3.1. *Speech Data*

We built a speech corpus, called the *Thai Syllable Speech Corpus (TSSC)*. The corpus is designed to cover the *coarticulatory effect*† of each phoneme in each syllable. It consists of 387 different syllables created from the combination of 38 initial consonants (33 Thai consonants + 5 borrowed consonants, i.e. /bl, br, dr, fl, fr/), 12 long vowels, 8 final consonants and 5 tones. The corpus was collected from 24 native

*The Directed Acyclic Graph (DAG) is a decision graph whose edges have an orientation and no cycles. A rooted DAG has a unique node such that it is the only node which has no arcs pointing into it. A rooted binary DAG has nodes which have either 0 or 2 arcs leaving them. The rooted binary DAG is used in order to define a class of functions to be used in classification task.[9]

†The Coarticulatory effect refers to the phenomenon whereby a given speech sound is altered in its phonetic manifestation depending upon influences from adjacent sounds.[10]

Thai speakers (6 male and 18 female speakers), ranging in age from 19 to 37 years (mean=24.30 and SD=5.16). The first two male speakers (M1-M2) and the first six female speakers (F1-F6) read all syllables for five trials, while the other speakers read all syllables for one trial. The speech signals were digitized by a 16-bit A/D converter of 11 kHz. These were then manually segmented to determine phoneme boundaries using audio-visual cues from a wave form display.

## 3.2. *Experimental setting*

To compare the recognition robustness against speaker variation, two tests were evaluated. The first test, the inside test, uses identical speech data in both training and testing while the second test, the outside test, uses the data from different speakers for training and testing. In the first test, we use all five trials from speakers: F1 to F6 and M1 to M2. The data is divided into 5 groups based on each trial. The training set is comprised of all utterances from four groups (totally 12384 tokens) and test set is comprised of all utterances from the other group (totally 3096 tokens). In the second test, the training set is comprised of the data from F7 to F18 and M3 to M6 (totally 6192 tokens) and the test set is the same as in the first test.

## 3.3. *Thai tone recognition*

Our first set of experiments focus on Thai tone recognition. This is comparatively a simple task, since the data is low dimensional; there are only five classes to choose between.

In Thai, there are five different lexical tones as follows: mid, low, falling, high, and rising. The following examples show the effect of tones on the meaning of an utterance[11] : /khāa/ ("a kind of grass"); /khàa/ ("galangale"); /khâa/ ("to kill"); /kháa/ ("to trade"); and /khǎa/ ("a leg"). The tone information is superimposed on the voiced portion of a syllable. The identification of a Thai tone relies on the shape of the fundamental frequency ($F_0$) contour. Fig.2 shows the average of $F_0$ contours of five different tones when syllables are spoken in isolation. To capture the characteristic of the $F_0$ contour, we designed tone features as follows.

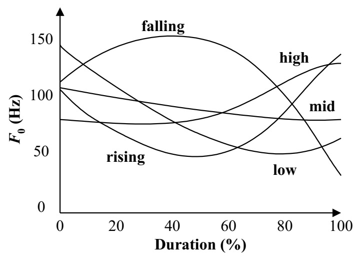

Fig. 2. $F_0$ contour patterns of the five Thai tones.

We first located the voiced portion before extracting an $F_0$. The voiced portion is detected using energy and zero crossing[12]. Then, the Average Magnitude Different

Function (AMDF) algorithm[13] is applied for $F_0$ extraction with 30 ms frame size and 5 ms frame shift. A searching method[14] is applied to smooth an $F_0$ contour. An $F_0$ is basically a physiologically determined characteristic and is regarded as being speaker dependent[12]. Therefore, for independent-speaker tone recognition that uses the relative $F_0$ of each syllable as the main discriminative feature, a normalization procedure is needed to align the range of the $F_0$ height for different speakers. In this paper, a raw $F_0$ is normalized by transforming the Hertz values to a z-score[15]. The precomputed mean and standard deviation are computed from raw $F_0$ values of all syllables for each speaker. Since not all syllables are of equal duration, $F_0$ contours of each syllable are equalized for duration on a percentage scale[10]. We obtain $F_0$'s at 11 different time points between 0 to 100% of voiced portion with the equal step size of 10%. Each $F_0$ is interpolated by Lagrange's interpolating polynomial based on four points around its position. Thus the $F_0$ profile of each utterance has the same dimension of 11. The $F_0$ profile is denoted as $\{F_0(0), F_0(1), F_0(2), \ldots, F_0(10)\}$.

The initial $F_0$, the final $F_0$ and four delta $F_0$'s at 20, 40, 60, and 80% of voiced portion are used as the tone features (for more information, see[16]). The delta $F_0$ ($dF_0$), initial $F_0$ ($F_{0I}$) and final $F_0$ ($F_{0F}$ ) are defined as follows:

$$dF_0 = F_0(n+1) - F_0(n-1) \tag{11}$$

$$F_{0I} = \frac{F_0(0) + F_0(1)}{2} \quad \text{and} \quad F_{0F} = \frac{F_0(9) + F_0(10)}{2} \tag{12}$$

Before training and evaluating, all features are also normalized to lie between -1.0 and 1.0 to avoid convergence problems with the quadratic optimizer.

We put particular emphasis on comparing different kernel functions, i.e. polynomial (from degree 1 through 10) and RBF ($c = 0.1, 0.2, 0.3, 0.4$ and $0.5$), and different multi-class classifiers using one-vs-all and one-vs-one (i.e. Max Wins and DDAG) approaches. The best results for each kernel function are shown in Table 1. It can be seen that the results of the RBF kernel often outperform those of the polynomial kernel. The DDAG provides the best recognition rates for most experiments. The results of the outside test are less than those of the inside test about 5%. The best recognition rates are 96.09% and 90.57% for the inside test and outside test, respectively. Both results are for DDAG.

The number of support vectors is a good indication of evaluation time. For one-vs-all and Max Wins, this number is the total number of unique support vectors for all SVMs, while, for DDAG, this number is the number of unique support vectors averaged over the evaluation paths through the DDAG. The results show that Max Wins is faster than one-vs-all and the DDAG has the fastest evaluation. In addition, polynomial kernels give a smaller number of support vectors than RBF kernels.

In order to provide a comparison, we conducted experiments using multi-layer perceptrons (MLPs). The MLP has an input layer of 6 units, a hidden layer of 10 units, and an output layer of 5 units corresponding to 5 Thai tones. The MLP was trained by the error back-propagation algorithm for a maximum of 1000 epochs.

We obtained the recognition rates of 95.48%, and 87.21% for the inside test and outside test, respectively. Comparing these results with the results of SVMs, we found that SVMs provide higher recognition rates than MLPs for all experiments.

Table 1. Recognition rates of tone recognition. The normalization factors of 6 are tailored to the dimensionality of the data. Therefore, the polynomial and RBF kernels used in the experiments are $K(\mathbf{x},\mathbf{y}) = \mid (\mathbf{x}\cdot\mathbf{y}+1)/6 \mid^d$ and $K(\mathbf{x},\mathbf{y}) = e^{-|\mathbf{x}-\mathbf{y}|^2/6c}$, respectively.

| | Polynomial | | | RBF | | |
|---|---|---|---|---|---|---|
| | $d$ | accuracy | no. of SVs | $c$ | accuracy | no. of SVs |
| The inside test | | | | | | |
| one-vs-all | 10 | 95.77 | 3197 | 0.1 | 96.09 | 5393 |
| Max Wins | 9 | 95.86 | 2818 | 0.1 | 96.09 | 4147 |
| DDAG | 9 | 95.93 | 1377 | 0.1 | 96.09 | 1774 |
| MLP | | 95.48 | | | | |
| The outside test | | | | | | |
| one-vs-all | 6 | 89.40 | 779 | 0.4 | 89.24 | 965 |
| Max Wins | 4 | 90.08 | 522 | 0.5 | 90.37 | 699 |
| DDAG | 4 | 90.57 | 229 | 0.5 | 90.53 | 300 |
| MLP | | 87.21 | | | | |

## 3.4. *Thai vowel recognition*

Our second set of experiments focus on a little more complicate task of Thai vowel recognition. Thai has 18 monophthongs, nine long and nine short. A pair of short and long phonemes are quantitatively different but qualitatively quite similar. In this paper, we concern only 12 long vowels.

For stationary vowels, a spectral shape is the only cue for phoneme discrimination and their spectra are adopted as reference spectra in the phonetic feature mapping[17]. On the contrary, nonstationary vowels contain two or three segments of reference vowels and transitions between these vowels. Therefore, we propose a novel vowel feature set such that each vowel segment is separated into three regions. For each region, 12-order RASTA[18] and their time derivatives were computed at the center of each region within 25 ms Hamming window. The vowel feature has the same dimension of 72.

Like the previous ones, these experiments used different kernel functions and different multi-class classifiers. The best results for each kernel functions are shown in Table 2. It show that the results of the RBF kernels are slightly better than those of the polynomial kernels for most experiments, except for one-vs-all in the inside test. For the inside test, one-vs-all and Max Wins provide the best recognition rates for polynomial and RBF kernels, respectively. For the outside test, Max Wins and one-vs-all yield the best recognition rates for polynomial and RBF kernels, respectively. The results of the inside test are less than those of the outside test about 8%. The best results are 95.51% (by Max Wins) and 87.08% (by one-vs-all) for the inside test and outside test, respectively.

Considering the number of support vectors, we found that one-vs-all is faster than Max Wins and the DDAG has the fastest evaluation. Moreover, polynomial kernels provide faster evaluation time than RBF kernels.

To evaluate the performance of SVMs, we also conducted experiments using multi-layer perceptron (MLP). The MLP has an input layer of 72 units, a hidden layer of 100 units, and an output layer of 12 units corresponding to 12 Thai vowels. The MLP was trained by the error back-propagation algorithm for a maximum of 2000 epochs. We obtained the recognition rates of 92.28% and 82.72% for the inside test and outside test, respectively. These results are also lower than those for SVMs.

Table 2. Recognition rates of vowel recognition. The normalization factors of 72 are tailored to the dimensionality of the data. Therefore, the polynomial and RBF kernels used in the experiments are $K(\mathbf{x},\mathbf{y}) = \mid (\mathbf{x}\cdot\mathbf{y}+1)/72 \mid^d$ and $K(\mathbf{x},\mathbf{y}) = e^{-|\mathbf{x}-\mathbf{y}|^2/72c}$, respectively.

| | Polynomial | | | RBF | | |
|---|---|---|---|---|---|---|
| | $d$ | accuracy | no. of SVs | $c$ | accuracy | no. of SVs |
| The inside test | | | | | | |
| one-vs-all | 7 | 94.99 | 14628 | 0.3 | 94.67 | 21188 |
| Max Wins | 7 | 94.48 | 19218 | 0.2 | 95.51 | 49596 |
| DDAG | 7 | 94.51 | 3242 | 0.3 | 94.64 | 6205 |
| MLP | | 92.28 | | | | |
| The outside test | | | | | | |
| one-vs-all | 6 | 86.18 | 9302 | 0.3 | 87.08 | 12814 |
| Max Wins | 7 | 86.28 | 13053 | 0.4 | 86.92 | 18341 |
| DDAG | 6 | 86.05 | 2096 | 0.4 | 86.75 | 3214 |
| MLP | | 82.72 | | | | |

From the above experiments, we may make a number of qualitative conclusions based on the quantitative results:

(i) The results depend on kernel functions, capacity control in the chosen type of structure, and multi-class classifier algorithms.

(ii) Although RBF kernels provide better recognition rates than polynomial kernels, the evaluation time of the RBF kernels is considerably longer than that of the polynomial kernels.

(iii) The DDAG provides the faster evaluation but it does not guarantee the recognition rate.

(iv) The recognition rates of the outside test are less than those of the inside test about 5% and 8% for tone and vowel recognition, respectively. This interval can be decreased by increasing the number of speakers in the training set or using features that are less sensitive to speaker variation.

(v) The recognition rates of SVMs are better than those of MLPs. Moreover the training times of SVMs are shorter than those of MLPs. This means that SVMs outperform MLPs in our tasks.

## 4. Conclusions

This paper presents a preliminary step for applying SVMs to Thai speech recognition. We have demonstrated that SVMs can be applied to two phoneme speech recognition problems: 5 Thai tones, and 12 Thai vowels spoken in isolation. The results show that multi-class SVMs based on one-against-one approach perform best and all techniques of multi-class SVMs surpass MLPs. In the future, we plan to extend SVMs for solving more difficult problems.

**Acknowledgement**

This work was supported in part by the Thailand-Japan Technology Transfer Project.

## References

1. V. Vapnik, *Statistical Learning Theory* (Wiley, 1998).
2. P. Clarkson and P. J. Moreno, "On the use of support vector machines for phonetic classification", in *Proc. Int. Conf. Acoust., Speech, Signal Processing.* **2** (1999) 585–588.
3. B. Schölkopf, "Support vector learning", Ph.D. Thesis, R.Oldenbourg Verlag Publications, Munich, Germany, 1997.
4. C. Burges, "A tutorial on support vector machines", *J. Data Mining and Knowledge Discovery.* **2** (1998).
5. M. Schmidt and H. Gish, "Speaker identification via support vector classifiers", in *Proc. Int Conf. Acoust., Speech, Signal Processing.* **12** (1996) 105–108.
6. A. Ganapathiraju, J. Hamaker and J. Picone, "Support vector machines for speech recognition", in *Proc. Int. Conf. Spoken Language Processing.* **12** (1998) 410–413.
7. P. Niyogi, C. Burges and P. Ramesh, "Distinctive feature detection using support vector machines", in *Proc. Int. Conf. Acoust., Speech, Signal Processing.* **1** (1999) 425–428.
8. J. H. Friedman, "Another approach to polychotomous classification", Technical report, Department of Statistics, Stanford, 1996.
9. J. C. Platt and N. Cristianini, "Large margin DAGs for multiclass classification", in S. A. Solla, T. K. Leen and K.-R. Müller, *Advance in Neural Inforamtion Processing System 12.* (1999) 547–553.
10. J. Gandour, S. Potisuk and S. Dechnongkit, "Tonal coarticulation in Thai", *J. Phonetics*, **22** (1994) 477–492.
11. S. Luksaneeyanawin, "Intonation in Thai", in D. Hirst and A. D. Cristo, *Intonation Systems A Survey of Twenty Language.* (1998) 376-394.
12. T. Lee, P. C. Ching, L. W. Chan, Y. H. Cheng and B. Mak, "Tone recognition of isolated cantonese syllables", *IEEE Trans. Speech Audio Processing* **3** (1995) 204–209.
13. M. J. Ross, H. L. Shaffer, A. Cohen, R. Freudberg and H. J. Manley, "Average magnitude difference function pitch extractor", *IEEE Trans. Acoust., Speech, Signal Processing*, **ASSP-22** (1974) 353–362.
14. L. Jun, X. Zhu and Y. Luo, "An approach to smooth fundamental frequencies in tone recognition", in *Proc. Int. Conf. Communication Technology*, **1** (Beijing, China 1998) S16-10-1–S16-10-5.
15. J. Gandour, A. Tumtavitikul and N. Satthamnuwong, "Effects of speaking rate on Thai tones", *Phonetica*, **56** 1999 123–134.
16. N. Thubthong, A. Pusittrakul and B. Kijsirikul, "An efficient method for isolated Thai tone recognition using combination of neural networks", in *Proc. the* $4^{th}$ *Symposium on Natural Language Processing* (2000) 224–242.
17. L. Bu and T. D. Chiueh, "Perceptual speech processing and phonetic feature mapping for robust vowel recognition", *IEEE Trans. Speech Audio Processing*, **8** (2000) 105–114.
18. H. Hermansky and N. Morgan, "RASTA processing of speech", *IEEE Trans. Speech Audio Processing*, **2** (1994) 578–589.