A Support Vector/Hidden Markov Model Approach to Phoneme Recognition

Steven E. Golowich and Don X. Sun

Bell Labs, Lucent Technologies
Murray Hill, NJ 07974

October 14, 1998

Abstract

A novel method for classifying frames of speech waveforms to a given set of phoneme classes is proposed. The method involves combining an approximation to multiple smoothing spline logistic regression (known as the "Support Vector Machine" in the machine learning literature) with hidden Markov models (HMMs). The method is compared with the standard technique in the speech recognition literature, that of HMMs with Gaussian mixture models. Both models were trained and tested using data drawn from the publicly available TIMIT database. Our results show that the two types of models are competitive for this data, but have very different structures. Such differences can be used to improve recognition rates by combining the two types of classifiers.

Key Words: Support Vector, hidden Markov models, phoneme recognition, classification.

1 Introduction

The lowest level of most speech recognition systems involves classifying short (e.g. 10 ms) windows, or *frames*, of speech waveforms to a given set of phoneme classes. The sequence of frames is then segmented and classified as a sequence of phonemes, which are further segmented and classified as a sequence of words. In this paper, we present a novel method for accomplishing the subtask of classifying a segment of frames, whose boundaries are specified in advance, to a set of phonemes.

Our method combines Support Vector classifiers (SVCs) with hidden Markov models (HMMs). SVCs have recently attracted a great deal of attention in the machine learning literature due to their strong performance on classification and regression problems [7, 5]. A characteristic feature of many such problems is that the input data consist of very high-dimensional vectors. Speech data is also high-dimensional, especially if context dependence is taken into account, which appears to make SVCs a natural choice for speech recognition tasks. However, there are several problems which arise in applying SVCs to speech data.

The most glaring difficulty is that SVCs do not explicitly take the temporal structure of the data into account. We resolve this by combining SVCs with hidden Markov models (HMMs). In this scheme, SVCs replace the Gaussian mixture models traditionally used in speech recognition.

Unfortunately, this replacement introduces a second problem. The scores produced by SVCs are distances in a metric space which has no simple interpretation, whereas HMMs call for conditional class probabilities. We avoid this difficulty by interpreting multiple SV classification as an approximation to multiple logistic smoothing spline regression. This allows us to obtain approximate conditional class probabilities from the SV results, which naturally fit into the HMM framework.

We present results of experiments that compare our method with the standard technique in the speech recognition literature, that of HMMs with Gaussian mixture models. We train and test both models using data drawn from the publicly available TIMIT database.

2 Support Vector Polychotomous Classifiers

In this section we describe the SVCs underlying our phoneme classification method. SVCs were introduced in the machine learning literature, where they are viewed as a generalization of maximum-margin classifiers with their attendant theoretical justification in terms of VC-bounds and structural risk minimization [7]. SVCs can also, however, be viewed as an approximation to smoothing spline logistic regression [9]. We will take the latter perspective, because it allows us to interpret the scores from the SVCs as conditional class probabilities, which is crucial for integrating them into HMMs. Accordingly, we will derive the optimization problem by which we train our classifiers purely in the language of smoothing splines.

We denote our training data set as

$$\{(y_i, x_i)\}_{i=1}^{\ell} \qquad y \in \{1...K\} \qquad x \in \mathbf{R}^n$$

which we assume is drawn iid from some density $p(x, y)$. The Bayes classification rule tells us to classify a new testing vector x' to the class given by $\arg\max_y p(y|x')$. We must estimate, from the training data, the K functions $p(y|x)$, which satisfy the constraint

$$\sum_{y=1}^{\ell} p(y|x) = 1. \tag{1}$$

For each conditional class probability $p(y|x)$, we introduce the log-odds ratio

$$h_y(x) = \log \frac{p(y|x)}{1 \Leftrightarrow p(y|x)}. \tag{2}$$

The log-likelihood of our training data is then given by

$$l = \sum_{i=1}^{\ell} \log p(y_i|x_i) = \Leftrightarrow \sum_{i=1}^{\ell} \log\left(1 + e^{-h_{y_i}(x_i)}\right).$$

One may obtain a smoothing spline estimate of these densities by adding a penalty term to the likelihood and maximizing the result. One way to do this is to write one of the odds ratios, say h_K, in terms of the others, which we may do implicitly with

$$p(K|x) = 1 \Leftrightarrow \sum_{y=1}^{K-1} p(y|x). \tag{3}$$

We then determine $h_1...h_{K-1}$ by maximizing

$$l \Leftrightarrow \lambda \sum_{y=1}^{K-1} \|h_y\|_\kappa, \tag{4}$$

where the λ is a free parameter that controls the strength of the penalties imposed by the norms $\|\cdot\|_\kappa$. The norms, in turn, are reproducing kernel Hilbert space (RKHS) norms defined by

$$\|f\|_\kappa = \int dx\, dy f(x) \kappa^{-1}(x, y) f(y)$$

where κ is a positive definite kernel, and κ^{-1} is interpreted as an operator inverse. In the smoothing spline literature, the penalties are often chosen as Sobolev space norms which penalize derivatives. Below, we will use different kernels drawn from the SVC's origins in the machine learning literature.

A standard argument [8] shows that the solution to the optimization problem (4) can always be written in the form

$$h_y(x) = \sum_{i=1}^{\ell} c_{y,i} \kappa(x_i, x) \tag{5}$$

with some constants $c_{y,i}$. By inserting (5) back into (4), the minimization over an infinite-dimensional space of functions is reduced to the finite-dimensional convex optimization problem of maximizing

$$\Leftrightarrow \sum_{i=1}^{\ell} \log\left(1 + e^{-h_{y_i}(x_i)}\right) \Leftrightarrow \sum_{y=1}^{K-1} \lambda(c_y, \tilde{\kappa}\, c_y), \tag{6}$$

where $\tilde{\kappa}_{i,j} = \kappa(x_i, x_j)$ and $(\cdot, \cdot)$ denotes the ℓ_2 inner product.

The problem (6) is hopelessly large to solve computationally. Two reasons are that the various odds ratios h_y are coupled to one another through (3), and that each one depends on ℓ free parameters. For speech data, ℓ can easily be on the order of 10^6. Clearly some simplifications are required.

We will make two approximations which will render (4) equivalent to K independent Support Vector classifiers (SVCs), which are computationally much more tractable to train. The first is to break the coupling between the various odds ratios by dropping the constraint (3), and treating h_K on the same footing as the other odds ratios. This gives us K independent two-class optimization problems: maximize

$$\Leftrightarrow \sum_{i:y_i=Y} \log\left(1 + e^{-h_Y(x_i)}\right) \Leftrightarrow \lambda \|h_Y\|_\kappa.$$

In order to approximate the effect of the constraint, to each two-class objective function we add the term

$$\Leftrightarrow \sum_{i:y_i \neq Y} \log\left(1 + e^{h_Y(x_i)}\right)$$

which has the effect of penalizing misclassifications of training data points not in the class Y.

As above, each solution can be represented as

$$h_Y(x) = \sum_{i=1}^{\ell} c_{Y,i} \kappa(x_i, x)$$

which yields the K independent optimization problems: maximize

$$\Leftrightarrow \sum_{i=1}^{\ell} \log\left(1 + \exp\left(\Leftrightarrow \delta_{Y,i} \sum_{j=1}^{\ell} c_Y\, \tilde{\kappa}_{i,j}\right)\right) \Leftrightarrow \lambda(c_Y, \tilde{\kappa}\, c_Y) \tag{7}$$

where we have defined

$$\delta_{Y,i} = \begin{cases} +1 & \text{if } y_i = Y \\ \Leftrightarrow 1 & \text{if } y_i \neq Y. \end{cases}$$

The problem (7) consists of K independent convex programming problems with smooth objective functions. This is considerably easier than (6) but is still computationally infeasible for more than a moderate number of training data points. Therefor, following [9], we make a further approximation which greatly reduces the size of each problem and brings us to the definition of SVCs.

The key observation is that

$$\log(1+e^{\tau}) \simeq (\tau)_{+}, \tag{8}$$

where $(x)_{+} = x$ if $x > 0$, 0 otherwise. In order to agree with the original definition of the SV approach [7], we replace the RHS of (8) with $(1+\tau)_{+}$. Each optimization problem then becomes

$$\text{minimize } \sum_{i=1}^{\ell} \left(1 \Leftrightarrow \delta_{Y,i} \sum_{j=1}^{\ell} c_{Y,j}\, \tilde{\kappa}_{i,j} \right)_{+} + \lambda (c_Y, \tilde{\kappa}\, c_Y). \tag{9}$$

We next recast (9) into the SV form [7], which is computationally very attractive. We introduce slack variables ξ_i and write the problem as

$$\text{minimize } \sum_{i=1}^{\ell} \xi_i + \lambda(c_Y, \tilde{\kappa}\, c_Y). \tag{10}$$

subject to the inequality constraints

$$\begin{aligned} 1 \Leftrightarrow \delta_{Y,i} \sum_{j=1}^{\ell} c_{Y,j} \kappa(x_j, x_i) &\leq \xi_i \\ \xi_i &\geq 0. \end{aligned}$$

The Lagrangian for (10) is

$$\begin{aligned} L = &\sum_{i=1}^{\ell} \xi_i + \lambda(c_Y, \tilde{\kappa}\, c_Y) \qquad (11) \\ &+ \sum_{i=1}^{\ell} \alpha_i \left(1 \Leftrightarrow \delta_{Y,i} \sum_{j=1}^{\ell} c_{Y,j}\, \tilde{\kappa}_{i,j} \Leftrightarrow \xi_i \right) \Leftrightarrow \sum_{i=1}^{\ell} r_i \xi_i, \end{aligned}$$

which must be minimized w.r.t. c_i, ξ_i and maximized w.r.t. α_i, r_i. Performing the (unconstrained) minimizations, we find that the optimal solution to (9) is

$$h_Y(x) = \sum_{i=1}^{\ell} \alpha_i\, \delta_{Y,i}\, \kappa(x_i, x) \tag{12}$$

where the α_i are the solutions to

$$\text{maximize } \sum_{i=1}^{\ell} \alpha_i \Leftrightarrow \sum_{i,j=1}^{\ell} \alpha_i\, \delta_{Y,i}\, \tilde{\kappa}_{i,j}\, \alpha_j\, \delta_{Y,j} \tag{13}$$

subject to

$$0 \leq \alpha_i \leq \lambda^{-1}.$$

The optimization problem is now a quadratic programming problem with simple box constraints.

We gain some insight into the nature of the solutions to (13) by noting the Kuhn-Tucker conditions

$$\begin{aligned} \alpha_i > 0 \quad &\Leftrightarrow \quad \delta_{Y,i} h_Y(x_i) \leq 1 \\ \alpha_i = 0 \quad &\Leftrightarrow \quad \delta_{Y,i} h_Y(x_i) > 1 \end{aligned}$$

We see that those training data points with associated Lagrange multiplier $\alpha_i = 0$ are correctly classified. Conversely, those points for which $\alpha_i > 0$ are incorrectly classified, or close to being incorrectly classified. The solution to the SV optimization problem is based solely on misclassified points and points close to the classification boundary. Points that are correctly classified fall in the insensitive zone in $(\cdot)_{+}$ in (9), which explains the reason for the insensitive zone.

The points with nonzero α_i are called the support vectors (SVs). The computational advantage in the SV approximation to smoothing spline logistic regression is that, for many problems, the SVs may comprise only a small fraction of the training data points. The expansion (12) in SVs is thus quite sparse, and consequently easy to find computationally.

Once we have trained the K odds ratios $h_Y(x)$, we may recover the estimated conditional class probabilities from (2). Because we approximated the constraint (1) the $p(Y|x)$ are not guaranteed to sum to unity, which we may rectify by renormalizing them.

Below we will describe the use of HMMs in phoneme recognition. The Viterbi algorithm for finding the optimal state sequence in an HMM calls for estimates of $p(x|y)$ or, equivalently, $p(x|y)/p(x)$. We may recover these from the $p(y|x)$ via

$$\frac{p(x|y)}{p(x)} = \frac{p(y|x)}{p(y)}$$

where we may estimate the class probabilities $p(y)$ as the frequencies of the classes in the training data.

3 Hidden Markov Models for Phoneme Recognition

A complete speech recognition system produces a sequence of transcribed words given a raw speech waveform. In this paper, we will concentrate on the particular sub-task of phoneme recognition, namely classifying segments of speech feature vectors into a set of phoneme labels.

A standard method for solving the segment recognition task in the speech recognition literature is the use of hidden Markov models (HMMs). HMMs have been widely used in automatic speech recognition systems for modeling non-stationary time-varying signals [3, 4, 6]. In this context, HMMs are used as probability models for phonemes.

To describe the standard HMM, let s_t denote the state of the system at time t. The Markov chain structure is specified by a state transition probability matrix $\mathbf{A} = [a_{ij}]$, where $a_{ij} = P(s_t = j | s_{t-1} = i), \ \ 1 \le i, j \le N$. The probability of the observation vector $\mathbf{O}_t$ in a given state i is denoted by $b_i(\mathbf{O}_t, \boldsymbol{\theta}_i)$. Assuming that all the observation vectors are independent, for a given state sequence $\mathbf{S} = (s_1, \cdots, s_T)$, the probability of the entire observation sequence $\mathbf{O} = (\mathbf{O}_1, \cdots, \mathbf{O}_T)$ is

$$P(\mathbf{O}|\mathbf{S};\boldsymbol{\theta}) = \prod_{t=1}^{T} b_{s_t}(\mathbf{O}_t).$$

When the state sequence is unknown, the likelihood function of the observation sequence $\mathbf{O}$ can be expressed as

$$P(\mathbf{O}|\boldsymbol{\theta}) = \sum_{\text{all } \mathbf{S}} P(\mathbf{O}, \mathbf{S}|\boldsymbol{\theta}) = \sum_{\text{all } \mathbf{S}} \prod_{t=1}^{T} [a_{s_{t-1}, s_t} \cdot b_{s_t}(\mathbf{O}_t)]$$

where the summation is over all possible state sequences. Therefore, an HMM can be specified by the number of states N and the parameter vector $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_N; \mathbf{A})$. During speech recognition, the probability of an incoming utterance $\mathbf{O}_{\text{new}}$ (i.e., the new observation sequence) is evaluated using all the HMMs in the inventory, and the one with the largest likelihood is recognized as the speech sound or the sequence of sounds that generates $\mathbf{O}_{\text{new}}$.

In principle, all aspects of an HMM (the number of states N, the transition matrix a_{ij}, and the output densities b_i) may be estimated from training data. The novelty of our method is in the use of a SVCs for modeling the densities b_i, as opposed to the standard Gaussian mixture models. Therefore, we will fix the other parameters of the HMMs to be those estimated by the standard method.

4 Experiments

We choose the standard acoustic-phonetic TIMIT database for the evaluation experiments in this paper.

A speech signal is originally recorded as a sequence of acoustic waveform. It is usually sampled at rate 8-20 kHz (1 kHz = 1000 times/second) with 16-bit amplitude quantization (i.e., each waveform sample is represented by one of 2^{16}=65,536 values). Since most of the phonetic features of speech signal are based on the spectral properties, we convert the time domain signal into frequency domain for speech signal modeling. Since the data obtained from discrete Fourier transformation has very high dimensionality (typically 256-dimension), it is not directly suitable to speech modeling. The common approach is to transform the high dimensional FFT spectrum to a low dimensional feature space using the so called mel-frequency cepstral coefficients (MFCC). More details on computing MFCC can be found in [1, 6].

Two important questions that must be answered before using SVCs are how to choose the kernel and the regularization parameter (box constraint). Also, we may vary the window around a time step when building feature vectors. In figure 1 we present results of binary SVCs separating pairs of phonemes. We have selected five sets of phonemes that range from easily distinguished to easily confused. For each pair, we trained SVCs at five different regularizations and with three different window widths. All of these SVCs were trained using a polynomial kernel of degree 2. Though there are exceptions, the results indicate that a high degree of regularization (corresponding to a small box constraint) is generally beneficial, and that window widths of 3 and 5 frames perform similarly, with both substantially outperforming 1 frame. The fact

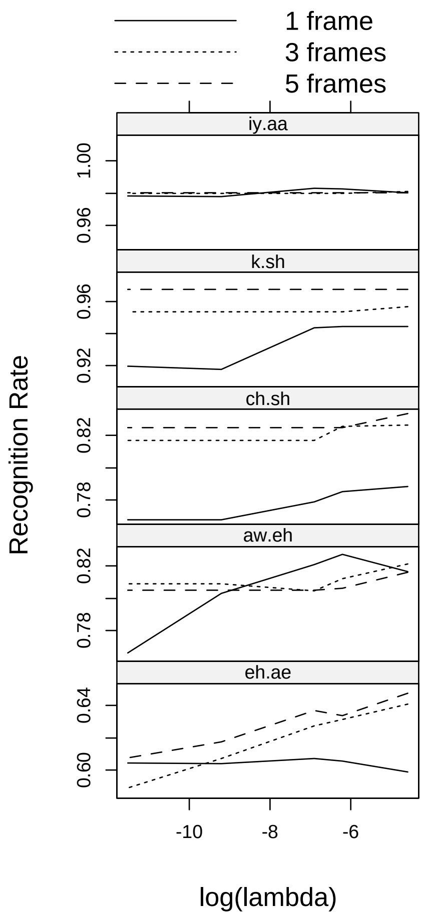

Figure 1: Comparison of binary SV classifiers on five sets of pairs of phonemes, at varying values of the regularization parameter λ.

that high regularization is necessary indicates that using higher order polynomial kernels would not be helpful. We ran some additional tests that corroborated this.

We next trained three full sets of phoneme classifiers, at three different regularizations, on the 16 speaker TIMIT training data set, with a window width of three frames and a degree two polynomial kernel. We incorporated these classifiers into an HMM and tested on the 24 speaker TIMIT test set. The best overall score of 54.9% correct was obtained with the strongest regularization, which compares with the best score from the GMM/HMM of 53.8%. In figure 2 we compare the results of the best SV/HMM model with those of the GMM/HMM at the phoneme level. Each row of the plot displays the fraction of a particular phoneme in the test set that were correctly identified by each of the two models. The overall scores of the two models are very similar. Of note, however, is that the models achieve the common overall score in very different ways. We observe large differences from phoneme to phoneme in figure 2. To take advantage of the different characteristics of these two classifiers, we develop a measure of prediction confidence based on likelihood ratios between the three best predictions. We first perform classification based on the SV/HMM model, and compute the confidence measure for each sample. Once the confidence measure drops below a predefined threshold, we switch to the prediction given by the GMM/HMM model. Figure 3 gives the results on combining SV/HMM with GMM/HMM using different thresholds. With a properly chosen threshold, the recognition rate can be improved from around 54% to around 59%, which is about 10% reduction in error rate.

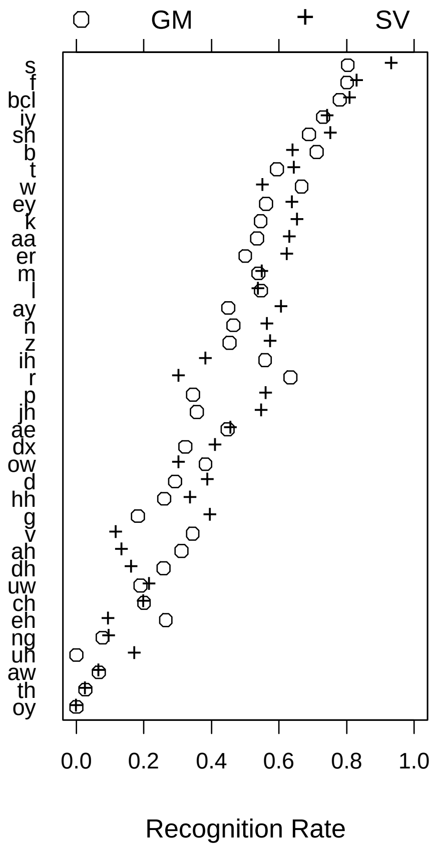

Figure 2: Fraction of phonemes in the testing data that were correctly identified by the two models

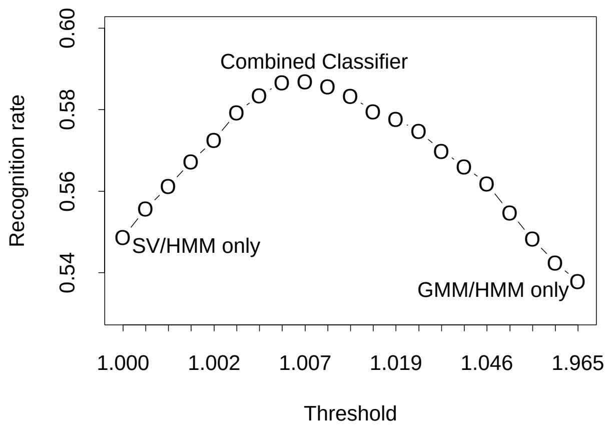

Figure 3: Combining SV/HMM with GMM/HMM using likelihood ratio based confidence measure

5 Discussion

Support Vector classifiers were introduced as binary classifiers in the statistical learning literature. Their original interpretation is theoretically attractive in certain ways, but presents problems in combining the results of several SVCs to solve polychotomous classification problems. The root of the problem is that the scores returned by binary SVCs are distances in a metric space which has no obvious interpretation. The most common way of doing K-fold multiple classification via SVCs has been by comparing scores of K SVCs, each trained to separate one class from the others. Though this method has been empirically the most successful, it it not obvious how to incorporate SVCs as components in higher level models such as HMMs due to the lack of interpretation of the scores.

By using the relationship between SVCs and smoothing splines, we have shown how to recover approximate conditional class probabilities from the scores of K-fold multiple SV classifiers. There are several advantages to this interpretation of SV scores. For our purposes, the most important is that it allows us to recover approximate conditional class probabilities which are required as inputs to an HMM. The ability to combine SVCs with HMMs opens up new areas of application to this attractive new classification method. Many types of data which vary dynamically with time are suitable for using HMMs.

We applied the combined SV/HMM classifier to the problem of phoneme recognition, which is a component in most current speech recognition systems. We found that SVCs are competitive with the Gaussian mixture models which are the standard in the field. In addition, we observed that the error structure of the SVC predictions is substantially different than that of Gaussian mixtures, which leads to significant gains in recognition rates by combining the two types of classifiers.

In certain types of problems, especially when the optimal decision boundaries are complicated, we believe that SVCs will have an advantage over Gaussian mixture models. In such cases, a large number of Gaussian mixture components will be required, which would make training impractical. SVCs, on the other hand, are designed to concentrate computational effort near the decision boundary, which enables them to train more complicated boundaries than Gaussian mixture models. We have observed that SVCs generally require significant regularization for the speech data we studied, meaning that the resulting decision boundaries are quite smooth. We expect that this is the reason for the competitive performance of Gaussian mixtures to SVCs in our experiments.

We end with some comments on promising directions for future work. An important issue is to assess the quality of the approximations in section 2. At low levels of regularization the approximation (8) can not be expected to hold. Also, dropping the constraint (1) may pose problems when one allows the regularization constants λ to vary across classifiers. An attractive alternative is to retain the constraints, at least partially. How much of a computational burden this would impose is an open question.

Our present results are based on a training data set that is much smaller than those found in the speech recognition literature. We believe a large amount of the difference can be made up by improvements in the quadratic optimization algorithm. Such algorithms can potentially converge very quickly when the solution contains many variables pinned at the boundaries [2], a fact our code does not fully exploit. Further improvements in training speed can be obtained by a preliminary clustering of the training data, though experiments will be necessary to assess the resulting loss in accuracy.

A potential advantage of SVCs over Gaussian mixtures is the former's seeming ability to cure the "curse of dimensionality" which arises in classifying high-dimensional feature vectors. For example, in our two-class experiments, we observed that adding additional frames to the feature vectors generally lowered the misclassification rates. This ability makes SVCs an attractive choice for combining new types of features with the standard cepstral coefficients.

References

[1] S. B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 28(4):357–365, 1980.

[2] L. Kaufman. Solving the quadratic programming problem arising in support vector classification. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*. MIT Press, Cambridge, USA, 1998.

[3] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77:257–285, 1989.

[4] L. R. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., New Jersey, 1993.

[5] B. Schölkopf, C. Burges, and A. Smola, editors. *Advances in Kernel Methods – Support Vector Learning*. MIT Press, Cambridge, USA, 1998.

[6] D. X. Sun, L. Deng, and C. F. J. Wu. Topics on hidden markov models and their applications in speech recognition. *American Statistical Association 1994 Proceedings of the Statistical Computing Section*, pages 90–99, 1994.

[7] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[8] G. Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, 1990.

[9] G. Wahba. Support vector machines, reproducing kernel hilbert spaces and the randomized gacv. Technical Report 984, University of Wisconsin Statistics Dept., 1997.