*Gini*Support Vector Machines for Segmental Minimum Bayes Risk Decoding of Continuous Speech

Veera Venkataramani * Shantanu Chakrabartty [2] William Byrne

Center for Language and Speech Processing
The Johns Hopkins University, Baltimore, MD 21218, USA

Abstract

We describe the use of Support Vector Machines (SVMs) for continuous speech recognition by incorporating them in Segmental Minimum Bayes Risk decoding. Lattice cutting is used to convert the Automatic Speech Recognition search space into sequences of smaller recognition problems. SVMs are then trained as discriminative models over each of these problems and used in a rescoring framework. We pose the estimation of a posterior distribution over hypothesis in these regions of acoustic confusion as a logistic regression problem. We also show that *Gini*SVMs can be used as an approximation technique to estimate the parameters of the logistic regression problem. On a small vocabulary recognition task we show that the use of *Gini*SVMs can improve the performance of a well trained Hidden Markov Model system trained under the Maximum Mutual Information criterion. We also find that it is possible to derive reliable confidence scores over the *Gini*SVM hypotheses and that these can be used to good effect in hypothesis combination. We discuss the problems that we expect to encounter in extending this approach to Large Vocabulary Continuous Speech Recognition and describe initial investigation of constrained estimation techniques to derive feature spaces for SVMs.

Key words: Support Vector Machines, Segmental Minimum Bayes Risk decoding, discriminative training, continuous speech recognition

* Address for Correspondence: 320 Barton Hall, 3400 N. Charles St., Baltimore, MD 21218. USA. Tel.: +1-410-516-5409; Fax: +1-410-516-5050.

Email addresses: veera@jhu.edu (Veera Venkataramani), shantanu@jhu.edu (Shantanu Chakrabartty), byrne@jhu.edu (William Byrne).

[1] This work was supported by the NSF (U.S.A) under the Information Technology Research (ITR) program, NSF IIS Award No. 0122466.

[2] Shantanu Chakrabartty was supported by a grant from the Catalyst Foundation, New York.

1 Introduction

Current state of the art Large Vocabulary Continuous Speech Recognition (LVCSR) systems are based on the Hidden Markov Model (HMM). The HMM naturally models variable length observations that are typical of speech. Each class (or word) can be modeled by a single HMM and can be readily incorporated into a larger classification task. Powerful algorithms (*e.g.,* Viterbi, A*) have also been developed for using HMMs in a sequential decision problem as complex as Automatic Speech Recognition (ASR). However, the HMM makes fundamental conditional independence assumptions about the processes that it models. Such assumptions do not hold for speech in general. Thus the HMM may be sub-optimal for use in ASR. We would like to develop an alternate framework, but not lose the benefits and performance of HMM based recognizers.

In their basic formulation, Support Vector Machines (Vapnik, 1995) are binary pattern classifiers. Given a data sample to be classified, the SVM will assign it as belonging to one of two classes. In training an SVM each labeled data point is represented as a real valued vector of fixed high dimension. The SVM is defined by a hyperplane in this feature space that is constructed so as to maximize a measure of the "margin" between two classes. A new data sample is classified by the SVM according to the decision boundary defined by the hyperplane. The location of the hyperplane is usually determined by a small number of the training samples which are ideally those near the boundaries of the two classes. SVMs are often observed to generalize well in cases when training data is limited. It is also possible to improve classification performance by transforming the raw data into a higher dimensional feature space so that the two classes can be more easily separated by a linear classifier. The original SVM has since been developed into a multi-class pattern classifier (Weston and Watkins, 1998). Due to these and other beneficial properties, SVMs have been successfully used in many pattern recognition tasks (Burges and Schölkopf, 1997; Drucker et al., 1997).

In speech recognition we would like to classify a variable length sequence of fixed dimension patterns typically vectors of acoustic spectral energy measurements as a sequence of words. These raw observation sequences cannot be assumed to have fixed dimension. Only the simplest of word or phrase recognition tasks can be described as classification of fixed-duration sequences. Also, since the number of possible sequences of words is countably infinite, representing each sequence of words as a single SVM is not feasible. Finally, if we were to model sub-sentence units with an SVM, there is no known framework that scales up to the classification problems of the size we typically address in ASR. If SVMs are to be employed in continuous ASR, their formulation as isolated-pattern classifiers of fixed dimension observations will have to be overcome or circumvented.

Smith et al. (2001) have developed *Score-Spaces* (Jaakkola and Haussler, 1998) to represent a variable length sequence of acoustic vectors via fixed dimensional vectors. This is done by using HMMs to find the likelihood of each sequence to be classified and then computing the gradient of the likelihood with respect to the HMM parameters. Since the HMMs have a fixed number of parameters, this yields a fixed-dimension feature to which the SVMs can be applied. It has the added benefit that the features provided to the SVM can be derived from a well-trained HMM recognizer. Smith and Gales (2002) provide an elegant explanation for why the SVMs trained on these Score-Spaces improve performance even though the scores are generated by the HMMs themselves. However, the SVM is still essentially an isolated pattern classifier, so that this approach is still limited to the classification of variable length sequences as isolated classes.

To apply SVMs beyond the isolated pattern classification problem we employ an approach to continuous speech recognition in which the recognition task is transformed into sequential, independent (and thus isolated) classification tasks. Each of these sub-tasks will be an independent recognition problem in which the goal is to decide which of several words were spoken. This yields a large but manageable sequence of decision problems and SVMs will be trained and applied to each. This is fundamentally an ASR rescoring approach. HMMs are used to generate recognition lattices in the usual way, and these lattices are post-processed to identify regions of acoustic confusion in which the first-pass HMMs were unable to distinguish between competing word hypotheses. The goal of this work is to apply SVMs to resolve the uncertainty remaining after the first-pass of the HMM-based recognizer. We will build on previous work in which this two-pass recognition approach was used to develop specialized discriminative training procedures for HMMs (Doumpiotis et al., 2003a,b). For clarity of presentation, we will also focus only on binary classification problems.

We refer to this divide-and-conquer recognition strategy as *acoustic code-breaking* (Jelinek, 1996). The idea is first to perform an initial recognition pass with the best possible system available, which we take as HMM-based; then isolate and characterize regions of acoustic confusion encountered in the first-pass; and finally apply models to each region that are specially trained for these confusion problems. This provides a framework for incorporating models that might not otherwise be appropriate for continuous speech recognition. We observe in passing that since the first-pass HMM system provides a proper posterior distribution over sequences, this approach may be less affected by the label-bias problem that can be encountered when discriminative classifiers are applied in sequential classification (Lafferty et al., 2001).

In addition to selecting a hypothesis from each region of acoustic confusion, we use the SVM to provide a posterior distribution over all the hypotheses

in each confusion set. This will allow us to associate a measure of confidence with each hypothesis. This has value in itself and is also useful for hypothesis combination. In particular we propose a voting scheme between the baseline HMM system and the SVMs that improves over the individual systems.

We set as the SVM training criterion the maximization of the posterior distribution over confusion sets found in the training set; in other words, we construct the SVM to lower the probability of error in training. We will employ the *Gini*SVM (Chakrabartty and Cauwenberghs, 2002) which is an SVM variant that can be directly constructed to provide a posterior distribution over competing hypotheses with the goal of minimizing classification error.

To place our work in context, there have been previous applications of SVMs to speech recognition. Ganapathiraju et al. (2003) obtain a fixed dimension problem by using a heuristic method to normalize the durations of each variable length sequence. The distances to the decision boundary in feature space are then transformed into phone posteriors using sigmoidal non-linearities. Smith et al. (2001) use score-spaces to train SVMs followed by a majority voting scheme among binary SVMs to recognize isolated letters. Golowich and Sun (1998) interpret multi-class SVM classifiers as an approximation to multiple logistic smoothing spline regression and use the resulting SVMs to obtain state emission densities of HMMs. Forward Decoding Kernel Machines (Chakrabartty and Cauwenberghs, 2002) perform maximum a posteriori forward sequence decoding, where transition probabilities are regressed as a kernel expansion of acoustic features and trained by maximizing a lower bound on a regularized form of cross-entropy. Salomon et al. (2002) uses a frame-by-frame classification approach and explores the use of the Kernel Fisher Discriminant for the application of SVMs for ASR.

The rest of the paper is organized as follows: we first give a brief introduction to ASR and formulate it as a sequential classification problem. Next we discuss the application of SVMs for variable length observations and use the *Gini*SVMs to approximate a posterior distribution over hypotheses via logistic regression. We will then list out the steps involved in implementing the new framework; this framework is evaluated in the experiments section. Following this we explore approaches to extend our work to large vocabulary tasks and conclude with final remarks.

2 Continuous Speech Recognition as a Sequence of Independent Classification Problems

The goal of the speech recognizer is to determine what word string W was spoken given an input acoustic signal O. The acoustic signal is represented

as a T-length string of spectral measurements $O = o_1, o_2, \cdots, o_T$ and W by a string of N words given by $W = w_1, w_2, \cdots, w_N$.

The *maximum a posteriori* (MAP) recognizer can then be formulated as follows: choose the most likely word string ($\hat{W}$) given the acoustic data:

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}} P(W|O), \tag{1}$$

where $\mathcal{W}$ represents all possible word strings. Since the search in Eq. (1) is independent of O, it follows the recognizer can simply pick according to the rule

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}} P(O|W)P(W). \tag{2}$$

To compute $P(O|W)$, we employ an *acoustic model*, usually an HMM. An HMM is defined by a finite state space $\{1, 2, \cdots, S\}$; an output space $\mathcal{O}$, usually R^D; transition probabilities between states $P(s_t = s'|s_{t-1} = s)$; and output distributions for states $P(o|s)$. For continuous output spaces, the output distribution of each state is modeled as a multiple mixture Gaussian mixture model

$$P(\mathbf{o}_t = o|\mathbf{s}_t = s) = \sum_{i=1}^{K} \frac{w_{i,s}}{(2\pi)^{D/2}|\Sigma_{i,s}|^{1/2}} \exp\left\{(o - \mu_{i,s})^\top \Sigma_{i,s}^{-1}(o - \mu_{i,s})\right\}, \tag{3}$$

where K is the number of Gaussian components, $w_{i,s}$, $\mu_{i,s}$ and $\Sigma_{i,s}$ are the mixture weight, mean and co-variance matrix of the ith component of the observation distribution of state s respectively. In this work, the effect of the language model ($P(W)$) is not studied; it is modeled by a uniform distribution.

In addition to producing the MAP hypothesis $\hat{W}$, the speech recognizer can also produce a set of most likely hypotheses that can be compactly represented by a lattice (see Fig. 1, a). Each link in the lattice represents a word hypothesis. Associated with each link are also the start and end times of the word hypothesis and the posterior probability of that word hypothesis relative to all the hypothesis in the lattice (Wessel et al., 1998). The N most likely hypotheses can also be generated from a lattice; such a list is called a N-best list.

2.1 The Sequential Problem Formulation

The MAP decoder as stated in Eq. (2) assumes all word strings are of equal importance. The Minimum Bayes Risk (MBR) decoder (Goel and Byrne, 2000)

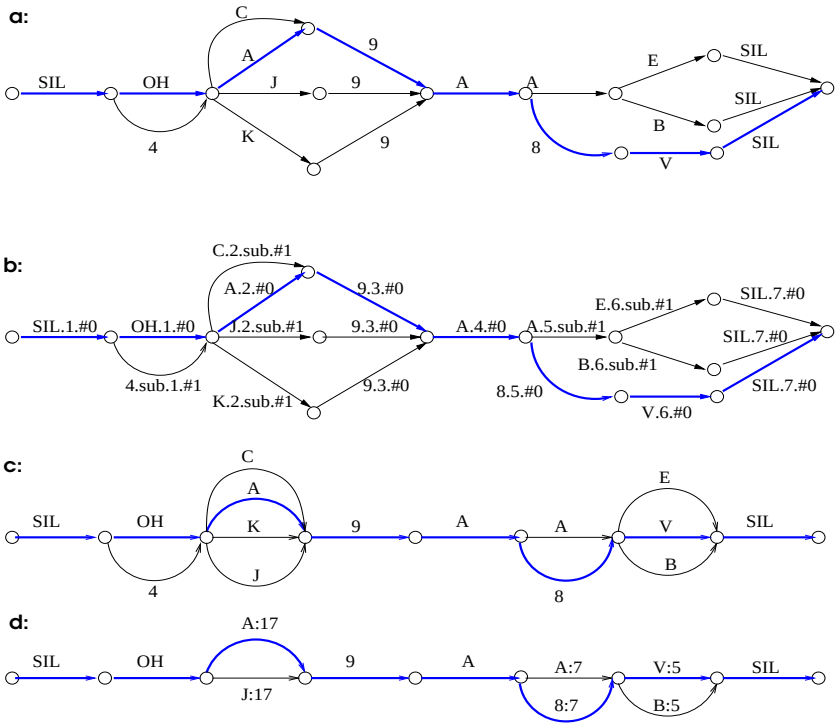

Fig. 1. Lattices and their segmentation. *a*: First-pass lattice of likely sentence hypotheses with a reference path in bold; *b*: Alignment of lattice paths to reference path; the link labels indicate word hypothesis, the segment index, the edit operation and its cost; *c*: Single word alternatives to reference word string; *d*: Search space $\hat{\mathcal{W}}_i$ consisting of binary segment sets selected for training of specialized models and rescoring. The specialized models have been tagged to distinguish them from the original models.

attempts to address this issue by associating an empirical risk $E(W)$ with each candidate hypothesis W. Given a loss function $l(W, W')$ between two word strings W and W', *e.g.,* the string-edit distance, $E(W)$ can be found as

$$E(W) = \sum_{W' \in \mathcal{W}} l(W, W') P(W'|O). \tag{4}$$

The goal of the MBR decoder is then to find the hypothesis with the minimum empirical risk as,

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmin}}\, E(W). \tag{5}$$

It is not feasible to consider all possible hypotheses while computing $E(W)$. A possible solution is to approximate $\mathcal{W}$ by an N-best list. However for coverage and computational reasons we use lattices as our hypothesis space. Thus we

find

$$E(W) = \sum_{W' \in \mathcal{L}} l(W, W') P(W'|O), \tag{6}$$

where $\mathcal{L}$ is a lattice for the utterance under consideration.

Given a string W, computation of Eq. (6) requires the alignment of every path in the lattice against W. Given the vast number of paths in a lattice, this cannot be done by enumeration. However, we have an efficient algorithm (Goel and Byrne, 2003; Kumar and Byrne, 2002) that transforms the original lattice into a form (Fig. 1, b) that contains the information needed to find the best alignment of every word string to the reference string W.

Using the alignment we can then transform the original lattice into a form in which all paths in the lattice are represented as alternatives to the words in the reference string W. This alignment identifies high confidence regions corresponding to the reference hypothesis as well as low confidence regions within which the lattice contains many alternatives. At this point we note that no paths have been removed; any path that was in the original lattice remains in the aligned lattice. Therefore we can use these segmented or *pinched* lattices for rescoring. We first discard alternatives that contain more than one word in succession; this gives groups of single word hypothesis (Fig. 1, c). We then apply likelihood based pruning to reduce the number of alternatives to produce pairs of confusable words (Fig. 1, d). Each of these remaining word pairs is called a confusion pair, G. $G(1)$ and $G(2)$ refer to the words in each confusion pair. Associated with each instance of these pairs in the lattices are the acoustic segments that caused these confusions; these are the acoustic observations and their start and end times. This pruning does reduce the search space; however alternatives to the reference hypothesis are available so that improvement is still possible.

2.2 MBR over segmented lattices

Let the original lattice be segmented into N sub-lattices, $\mathcal{W}_1, \mathcal{W}_2, \cdots, \mathcal{W}_N$. We can perform MBR decoding using the loss induced (L_I) by lattice cutting,

$$\hat{W} = \underset{W' \in \mathcal{L}}{\operatorname{argmin}} \sum_{W \in \mathcal{L}} L_I(W, W') P(W|O), \tag{7}$$

which reduces (Goel et al., 2001; Goel and Byrne, 2003) to

$$\hat{W}_i = \underset{W' \epsilon \mathcal{W}_i}{\operatorname{argmin}} \sum_{W \epsilon \mathcal{W}_i} l(W, W') P_i(W|O) \tag{8}$$

where $\hat{W}_i$ is the minimum risk path in the ith sub-lattice and $\mathcal{W}_i$ represents all possible strings in the ith sub-lattice. The sentence-level MBR hypothesis is obtained as $\hat{W} = \hat{W}_1 \cdot \hat{W}_2 \cdots \hat{W}_M$ (Goel et al., 2003). Note that this formulation allows for the use of specially trained probability models $P_i(W|O)$ for each sub-lattice $\mathcal{W}_i$. We emphasize while the hypothesis space $\mathcal{L}$ has been segmented, the observed acoustics $\mathbf{O}$ remain unsegmented. In the case of binary decision problems, each $\mathcal{W}_i$ that contains alternatives is reduced to a confusion pair $G_i = \{w_1, w_2\}$, where the subscripts indicate their classes. If $l(\cdot, \cdot)$ is taken to be the string-edit distance, Eq. (8) reduces to

$$\hat{W}_i = \underset{W \epsilon G_i}{\operatorname{argmin}} \{P_i(w_1|O)\delta(W, w_2),\ P_i(w_2|O)\delta(W, w_1)\} \tag{9}$$

$$= \underset{W \epsilon G_i}{\operatorname{argmax}}\, P_i(W|O) \tag{10}$$

i.e., the sub-lattice $\mathcal{W}_i$ specific decoder chooses the word with the higher posterior probability. Note that in Eq. (9) the loss associated with an hypothesis is the posterior probability of its alternative. As can be seen in Fig. 1, d it often happens that in many cases the $\mathcal{W}_i$ contain only a single word. In these cases the word from the reference string is selected as the segment hypothesis.

In summary, lattice cutting converts ASR into a sequence of smaller, independent regions of acoustic confusion. Specialized decoders can then be trained for these decision problems and their individual outputs can be concatenated to obtain a new system output. We will next discuss Support Vector Machines and a formulation which allows them to applied in this way.

3 Support Vector Machines for Variable Length Observations

We now briefly review the basic SVM (Vapnik, 1995). Let $\{\mathbf{x}^i\}_{i=1}^l$ be the training data and $\{y^i\}_{i=1}^l$ be the corresponding labels, where $\mathbf{x}^i \in \mathbf{R}^d$ and $y^i \in \{-1, +1\}$. Training an SVM involves maximizing a measure of margin between the two classes, or equivalently, minimizing the following cost function

$$\frac{1}{2}\|\phi\|^2 - C\left[\sum_i 1 - y^i(\phi \cdot \zeta(\mathbf{x}^i) + \mathbf{b})\right]_+ \tag{11}$$

where $\|\phi\|^{-1}$ is the margin, C is the SVM trade-off parameter that determines how well the SVM fits the training data, ζ is the mapping from the input space ($\mathbf{R}^d$) to a higher dimensional feature space, $\mathbf{b}$ is the bias of the hyperplane separating the two classes and $[\cdot]_+$ gives the positive part of the

argument. This minimization is carried out using the technique of Lagrangian multipliers (Boser et al., 1992) which results in minimizing

$$\frac{1}{2}\sum_{i,j}\alpha_i \mathbf{K}(\mathbf{x}^i, \mathbf{x}^j)\alpha_j - \sum_i \alpha_i \tag{12}$$

subject to

$$\sum_i y^i \alpha_i = 0, \quad \text{and} \quad 0 \leq \alpha_i \leq C, \tag{13}$$

where α_i are the Lagrange multipliers and $\mathbf{K}(\cdot,\cdot)$ is the kernel function that computes an inner product in the higher dimensional feature space $\zeta(\cdot)$ (Cortes and Vapnik, 1995). New observations $\mathbf{x}$ are classified using the decision rule

$$\hat{y} = \text{sgn}\left(\sum_i y^i \alpha_i \mathbf{K}(\mathbf{x}, \mathbf{x}^i) + \mathbf{b}\right). \tag{14}$$

3.1 Feature Spaces

SVMs are static classifiers; a data sample to be classified must belong to the input space ($\mathbf{R}^d$). However, speech utterances vary in length. To be able to use SVMs for speech recognition we need some method to transform variable length sequences into vectors of fixed dimension. Towards this end, we would also like to use the HMMs that we have trained so that some of the advantages of the generative models can be used along with the discriminatively trained models.

Fisher scores (Jaakkola and Haussler, 1998) have been suggested as a means to map variable length observation sequences into fixed dimension vectors and the use of Fisher scores has been investigated for ASR (Smith et al., 2001). Each component of the Fisher score is defined as the sensitivity of the likelihood of the observed sequence to each parameter of an HMM. Since the HMMs have a fixed number of parameters, this yields a fixed-dimension feature even for variable length observations. Smith et al. (2001) have extended Fisher scores to score-spaces in the case when there are two competing HMMs. This formulation has the added benefit that the features provided to the SVM can be derived from a well-trained HMM recognizer. For a complete treatment of Score-Spaces, see Smith and Gales (2002).

For discriminative binary classification problems, the log likelihood-ratio Score-Space has been found to perform best among a variety of possible score-spaces. If we have two HMMs with parameters θ_1 and θ_2 and corresponding likelihoods

$p_1(\mathbf{O};\theta_1)$ and $p_2(\mathbf{O};\theta_2)$, the projection of an observation sequence ($\mathbf{O}$) into the log likelihood-ratio score-space is given by

$$\varphi(\mathbf{O};\theta) \;=\; \begin{bmatrix} \varphi_0(\mathbf{O};\theta) \\ \varphi_1(\mathbf{O};\theta_1) \\ -\varphi_2(\mathbf{O};\theta_2) \end{bmatrix} = \begin{bmatrix} \ln \frac{p_1(\mathbf{O};\theta_1)}{p_2(\mathbf{O};\theta_2)} \\ \nabla_{\theta_1} \ln p_1(\mathbf{O};\theta_1) \\ -\nabla_{\theta_2} \ln p_2(\mathbf{O};\theta_2) \end{bmatrix} \tag{15}$$

where $\theta = [\theta_1 \;\; \theta_2]$.

In our experiments we derive the score space solely from the means of the multiple-mixture Gaussian HMM state observation distributions, denoted via the shorthand $\theta_i[s,j,k] = \mu_{i,s,j}[k]$, where k denotes a component of a vector; the decision to focus only on the Gaussian means will be discussed in Section 6. We first define the parameters of the j^{th} Gaussian observation distribution associated with state s in HMM i as $(\mu_{i,s,j}, \Sigma_{i,s,j})$. The gradient with respect to these parameters (Smith et al., 2001) is

$$\nabla_{\mu_{i,s,j}} \ln P(\mathbf{O};\theta_i) = \sum_{t=1}^{T} \gamma_{i,s,j}(t) \Big[(o_t - \mu_{i,s,j})^\top \Sigma_{i,s,j}^{-1} \Big]^\top, \tag{16}$$

where $\gamma_{i,s,j}$ is the posterior for mixture component j, state s under the i^{th} HMM found via the Forward-Backward procedure; and T is the number of frames in the observation sequence. Scores have to be normalized for the sequence length T, as they are accumulators over the individual observations.

3.2 Posterior Distributions Over Segment Sets by Logistic Regression

SMBR decoding over binary classes requires estimation of the posterior distribution $P(W|\mathbf{O})$ (Eq. (10)) over binary segment sets $\{w_1, w_2\}$. To apply SVMs to classification within the segment sets, we will first recast this posterior calculation as a problem in logistic regression. Our approach follows the general approach of Jaakkola and Haussler (1998).

If we have binary problems with HMMs as described in the previous section, the posterior can be found by first computing the quantities $p_1(\mathbf{O};\theta_1)$ and $p_2(\mathbf{O};\theta_2)$ so that

$$P(w_j|\mathbf{O};\theta) = \frac{p_j(\mathbf{O};\theta_j)P(w_j)}{p_1(\mathbf{O};\theta_1)P(w_1) + p_2(\mathbf{O};\theta_2)P(w_2)} \quad j = 1, 2\,. \tag{17}$$

This distribution over the binary hypotheses can be rewritten as

$$P(w|\mathbf{O};\theta) = \frac{1}{1+\exp[1+k(w)\log\frac{p_1(\mathbf{O};\theta_1)}{p_2(\mathbf{O};\theta_2)}+k(w)\log\frac{P(w_1)}{P(w_2)}\,]} \tag{18}$$

$$\text{where} \quad k(w) = \begin{cases} +1 & w = w_1 \\ -1 & w = w_2 \end{cases} .$$

If a set of HMM parameters $\bar{\theta}$ is available, the posterior distribution can be found by first evaluating the likelihood ratio $\log\frac{p_1(\mathbf{O};\bar{\theta}_1)}{p_2(\mathbf{O};\bar{\theta}_2)}$ and inserting the result into Eq. (18). If a new set of parameter values becomes available, the same approach could be used to reestimate the posterior. Alternatively, the likelihood ratio could be considered simply as a continuous function in θ whose value could be found by a Taylor Series expansion around $\bar{\theta}$

$$\log\frac{p_1(\mathbf{O};\theta_1)}{p_2(\mathbf{O};\theta_2)} = \log\frac{p_1(\mathbf{O};\bar{\theta}_1)}{p_2(\mathbf{O};\bar{\theta}_2)} + (\theta-\bar{\theta})\,\nabla_\theta \log\frac{p_1(\mathbf{O};\bar{\theta}_1)}{p_2(\mathbf{O};\bar{\theta}_2)} + \cdots \tag{19}$$

which of course is only valid for $\theta \approx \bar{\theta}$.

If we ignore the higher order terms in this expansion and gather the statistics into a vector

$$\Psi(\mathbf{O};\bar{\theta}) = \begin{bmatrix} \varphi_0(\mathbf{O};\bar{\theta}) \\ \varphi_1(\mathbf{O};\bar{\theta}_1) \\ \varphi_2(\mathbf{O};\bar{\theta}_2) \\ 1 \end{bmatrix} \tag{20}$$

we obtain the following approximation for the posterior at θ

$$P(w|\mathbf{O};\theta) \approx \frac{1}{1+\exp[\; k(w)\;[1 \quad (\theta-\bar{\theta}) \quad \log\frac{P(w_{+1})}{P(w_{-1})}]\;\; \Psi(\mathbf{O};\bar{\theta})\;]} \;\; . \tag{21}$$

We will realize this quantity by the logistic regression function

$$P_a(w|\mathbf{O};\phi) = \frac{1}{1+\exp[\; k(w)\;\phi^\top\;\Psi(\mathbf{O};\bar{\theta})\;]} \tag{22}$$

and Eq. (21) is realized exactly if we set

$$\phi = \begin{bmatrix} \phi_0 \\ \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} = \begin{bmatrix} 1 \\ \theta_1 - \bar{\theta}_1 \\ \theta_2 - \bar{\theta}_2 \\ \log \frac{P(w_1)}{P(w_2)} \end{bmatrix} \quad . \tag{23}$$

Our goal is to use estimation procedures developed for large margin classifiers to estimate the parameters of Eq. (22) and in this we will allow ϕ to vary freely. This has various implications for our modeling assumptions. If we allow ϕ_3 to vary, this is equivalent to computing P_a under a different prior distribution than initially specified. If ϕ_1 or ϕ_2 vary, we allow the parameters of the HMMs to vary from their nominal values $\bar{\theta}_1$ and $\bar{\theta}_2$. This might produce parameter values that lead to invalid models, although we restrict ourselves here to the means of the Gaussian observation distributions which can be varied freely. Variations in ϕ_0 are harder to interpret in terms of the original posterior distribution derived from the HMMs; despite that, we still allow this parameter to vary.

3.3 GiniSVMs

Taking the form of Eq. (22), we assume that we have a labeled training set $\{\bar{\mathbf{O}}^j, \bar{w}^j\}_j$ and that we wish to refine the distribution P_a over the data according to the following objective function

$$\min_{\phi} \quad \frac{1}{2}\|\phi\|^2 - C \sum_j \log P_a(\bar{w}^j | \bar{\mathbf{O}}^j; \phi) \quad , \tag{24}$$

where C is a trade-off parameter that determines how well P_a fits the training data. The role of the regularization term $\|\phi\|^2$ penalizes HMM parameter estimates that vary too far from their initial values $\bar{\theta}$. Similarly, it allows reestimation of the prior over the hypotheses, but prefers estimates that assign comparable likelihood to hypotheses.

If we define a binary valued indicator function over the training data

$$y^j = \begin{cases} +1 & w^j = w_1 \\ -1 & w^j = w_2 \end{cases}$$

we can use the approximation techniques of Chakrabartty and Cauwenberghs (2002) to minimize Eq. (24) where the dual is given by

$$\frac{1}{2}\sum_{i,j}\alpha_i\ [\ \mathbf{K}(\ \Psi(\mathbf{O}^i;\bar{\theta}),\ \Psi(\mathbf{O}^j;\bar{\theta})\ ) + \frac{2\gamma}{C}\delta_{ij}\ ]\ \alpha_j - 2\gamma\sum_i \alpha_i \tag{25}$$

subject to

$$\sum_i y^i\alpha_i = 0, \quad 0 \leq \alpha_i \leq C, \tag{26}$$

where γ is the rate distortion factor chosen as $2\log 2$ in the case of binary classes and δ_{ij} is the Kronecker delta function. The optimization can be carried out using the *Gini*SVM Toolkit which is available online (Chakrabartty, 2003).

After the optimal parameters α are found, the posterior distribution of an observation is found as

$$P_a(w|\mathbf{O};\phi) = \frac{1}{1+\exp[\ k(w)\ \phi^\top\ \zeta(\Psi(\mathbf{O};\bar{\theta}))]} \tag{27}$$

$$= \frac{1}{1+\exp[\ k(w)\sum_i y^i\ \alpha_i\ \mathbf{K}(\ \Psi(\mathbf{O}^i;\bar{\theta}),\ \Psi(\mathbf{O};\bar{\theta})\ )]}\ , \tag{28}$$

and ϕ can be written as $\phi = \sum_i \alpha_i\ y^i\ \zeta(\Psi(\mathbf{O}^i;\bar{\theta}))$.

Using *Gini*SVM in this way allows us to estimate the posterior distribution under penalized likelihood criterion of Eq. (24). The distribution that results can be used directly in the classification of new observations with the added benefit that the form of the distribution in Eq. (28) makes it easy to assign 'confidence scores' to hypotheses. This will be useful in the weighted hypothesis combination rescoring procedures that will be described subsequently.

4 Modeling Issues

4.1 Estimation of sufficient statistics

There are two algorithms one can use for computing the mixture-level posteriors in Eq. (16), the Viterbi and the Baum-Welch algorithm. If time segmentations of the utterance at the word level are available, we can simply normalize the scores with the length of the word. Otherwise, the sum of the state occupancy over the entire utterance is more appropriate (Smith and Gales, 2002), *i.e.*,$\sum_{t=1}^{T}\gamma_s(t)$, where s is the state index.

We want to apply SVMS to word hypotheses in continuous speech recognition. In these cases, the start and end times of the hypotheses are uncertain. One possibility is to take the timing information from the first pass ASR output. Another alternative can be seen from the example in Fig. 1, d. Consider the confusion pair A:17 *vs.* J:17. We can compute the statistics by performing two Forward-Backward calculations with respect to the transcriptions

SIL OH A:17 NINE A EIGHT B SIL

SIL OH J:17 NINE A EIGHT V SIL

where A:17 and J:17 are cloned versions of models A and J respectively. In this case, since word boundaries are not fixed, duration is unknown and cannot be used for normalization. The sum of the state occupancies as mentioned earlier can be used in this case.

When we perform Forward-Backward calculations over the entire utterance, it is possible to also consider the alternatives paths in the neighboring confusion segments. For the confusion pair B:5 *vs.* V:5 in Fig. 1, d, this would imply considering the following four hypotheses:

SIL OH A NINE A EIGHT B:5 SIL

SIL OH A NINE A EIGHT V:5 SIL

SIL OH A NINE A A B:5 SIL

SIL OH A NINE A A V:5 SIL

4.2 Normalization

While a linear classifier can subsume a bias in the training, the parameter search (α_i in Eq. 25) can be made more effective by ensuring that the training data is normalized. We first adjust the scores for each acoustic segment via mean and variance normalization. The normalized scores are given by

$$\varphi^N(\mathbf{O}) = \hat{\Sigma}_{sc}^{-1/2}[\varphi(\mathbf{O}) - \hat{\mu}_{sc}], \tag{29}$$

where $\hat{\mu}_{sc}$ and $\hat{\Sigma}_{sc}$ are estimates of the mean and variances of the scores as computed over the training data of the SVM. Ideally, the SVM training will subsume the $\hat{\mu}_{sc}$ bias and the variance normalization would be performed by the scaling matrix $\hat{\Sigma}_{sc}$ as

$$\varphi^N(\mathbf{O}) = \hat{\Sigma}_{sc}^{-1/2}\varphi(\mathbf{O}) \tag{30}$$

where $\hat{\Sigma}_{sc} = \int \varphi(\mathbf{O})'\varphi(\mathbf{O})P(\mathbf{O}|\theta)d\mathbf{O}$. For implementation purposes, the scaling matrix is approximated over the training data as

$$\hat{\Sigma}_{sc} = \frac{1}{N-1}\sum(\varphi(\mathbf{O}) - \hat{\mu}_{sc})^{\top}(\varphi(\mathbf{O}) - \hat{\mu}_{sc}) \tag{31}$$

where $\hat{\mu}_{sc} = \frac{1}{N}\sum\varphi(\mathbf{O})$, and N is the number of training samples for the SVM. However we used a diagonal approximation for Σ_{sc} since the inversion of the full matrix $\hat{\Sigma}_{sc}$ is problematic. Prior to the mean and variance normalization, the scores for each segment are normalized by the segment length T.

4.3 Dimensionality Reduction

For efficiency and modeling robustness there may be value in reducing the dimensionality of the score-space. There has been research (Blum and Langley, 1997; Smith and Gales, 2002) to estimate the information content of each dimension so that non-informative dimensions can be discarded. Assuming independence between dimensions, the goodness of a dimension can be found based on Fisher discriminant scores as (Smith and Gales, 2002)

$$g[d] = \frac{|\hat{\mu}_{sc[1]}[d] - \hat{\mu}_{sc[2]}[d]|}{\hat{\Sigma}_{sc[1]}[d] + \hat{\Sigma}_{sc[2]}[d]} \tag{32}$$

where $\hat{\mu}_{sc[i]}(d)$ is the dth dimension of the mean of the scores of the training data with label i and $\hat{\Sigma}_{sc[i]}[d]$ are the corresponding diagonal variances. SVMs can then be trained only in the most informative dimensions by applying a pruning threshold to $g[d]$.

4.4 GiniSVM and its Kernels

*Gini*SVMs have the advantage that, unlike regular SVMs, they can employ non positive-definite kernels. For ASR, the linear kernel ($\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i{}' \cdot \mathbf{x}_j$), has previously been found to perform best among a variety of positive-definite kernels (Smith and Gales, 2002). We found that while the linear kernel does provide some discrimination, it was not sufficient for satisfactory performance. This observation can be illustrated using kernel maps. A kernel map is a matrix plot that displays kernel values between pairs of observations drawn from two classes, $G(1)$ and $G(2)$. Ideally if $\mathbf{x}, \mathbf{y} \in G(1)$ and $\mathbf{z} \in G(2)$, then $\mathbf{K}(\mathbf{x}, \mathbf{y}) \gg \mathbf{K}(\mathbf{x}, \mathbf{z})$. and the kernel map would be block diagonal. In Figs. 2 and 3, we draw 100 samples each from two classes to compare the linear kernel map to the tanh kernel ($\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \tanh(d * \mathbf{x}_i{}' \cdot \mathbf{x}_j)$) map. Visual

inspection shows that the map of the tanh kernel is closer to block diagonal. We have found in our experiments with *Gini*SVM that the tanh kernel far outperformed the linear kernel; we therefore focus on tanh kernels for the rest of the paper.

We also found that the *Gini*SVM classification performance was sensitive to the SVM trade-off parameter C; this is in contrast to earlier work (Smith et al., 2001). Unless mentioned otherwise, a value of $C = 1.0$ was chosen for all the experiments in this paper to balance between over-fitting and the time required for training.

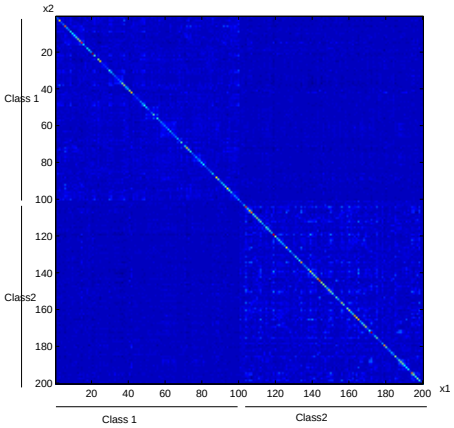

Fig. 2. Kernel Map $\mathbf{K}(\ \Psi(\mathbf{O}^i;\bar{\theta}),\ \Psi(\mathbf{O}^j;\bar{\theta})\ )$ for the linear kernel over two class data.

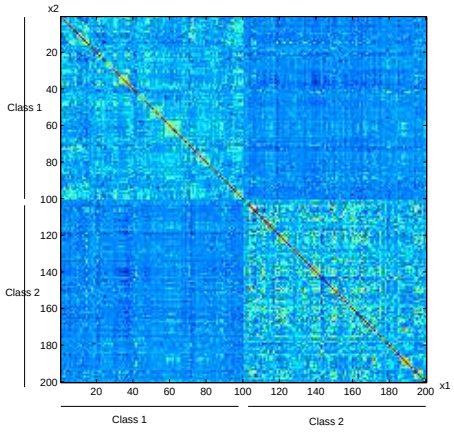

Fig. 3. Kernel Map $\mathbf{K}(\ \Psi(\mathbf{O}^i;\bar{\theta}),\ \Psi(\mathbf{O}^j;\bar{\theta})\ )$ for tanh kernel over two class data.

5 The SMBR-SVM framework

We now describe the steps we performed to incorporate SVMs in the SMBR framework.

5.1 Identifying confidence sets in the training set

Initial lattices are generated using the baseline HMM system to decode the speech in the training set. The lattices produced are then aligned against the reference transcriptions (Goel et al., 2003). Period-1 lattice cutting is performed and each sub-lattice is pruned (by the word posterior) to contain two competing words. This process identifies regions of confusion in the training set. The most frequently occurring confusion pairs (confusable words) are kept, and their associated acoustic segments are identified, retaining time boundaries and the true identity of the word spoken.

5.2 Training SVMS for each confusion pair

For each acoustic segment in every sub-lattice, likelihood-ratio scores as given by Eq. (15) are generated. The dimension of these scores is equal to the sum of the number of parameters of the two competing HMMs plus one. If necessary, the dimension of the score-space is reduced using the goodness criterion (Eq. (32)) with appropriate thresholds. SVMs for each confusion pair are then trained in our normalized score-space using the appropriate acoustic segments identified as above.

5.3 SMBR decoding with SVMs

Initial test set lattices are generated using the baseline HMM system. The MAP hypothesis is obtained from this decoding pass and the lattice is aligned against it. Period-1 lattice pinching is performed on the test set lattices. Instances of confusion pairs for which SVMs were trained are identified and retained; other confusion pairs are pruned back to the MAP word hypothesis. The appropriate SVM is applied to the acoustic segment associated with each confusion pair in the lattice. The HMM outputs in the regions of high confidence are concatenated with the outputs of the SVMs in the regions of low confidence. This is the final hypothesis of the SMBR-SVM system.

5.4 Posterior-based System Combination

We now have the HMM and the SMBR-SVM system hypotheses along with their posterior estimates. If these posterior estimates serve as reliable confidence measures, we can combine the system hypotheses to yield better performance. Several voting schemes have been proposed on how to choose between

the outputs of two or more systems. We use either

$$\hat{p}_{+}(i) = \frac{p_h(i) + p_s(i)}{2}. \tag{33}$$

or

$$\hat{p}_{\times}(i) = \frac{p_h(i)p_s(i)}{p_h(1)p_s(1) + p_h(2)p_s(2)}. \tag{34}$$

where $p_h(1)$ and $p_h(2)$ are the posterior estimates of the two competing words in a segment as estimated by the HMM system and $p_s(1)$ and $p_s(2)$ of the SMBR-SVM system. Both these schemes then pick the word with the new higher estimate.

5.5 Rationale

The most ambitious formulation of acoustic code-breaking is to first identify all acoustic confusion in the test set, and then return to the training set to find any data that can be used to train models to remove the confusion. To present these techniques and show that they can be effective, we have chosen for simplicity, to focus on modeling the most frequent errors found in training. Earlier work (Doumpiotis et al., 2003a) has verified that training set errors found in this way are good predictors of errors that will be encountered in unseen data.

6 Initial Experiments

We evaluate our proposed method on the OGI-Alphadigits corpus (Noel, 1997). This is a small vocabulary task that is fairly challenging. The baseline Word Error Rates (WERs) for ML models are around 10%; this ensures that there are enough number of errors to allow for analysis. The corpus has a vocabulary of 36 words: 26 letters and 10 digits. The corpus has 46,730 training and 3,112 test utterances. We first describe the training procedure for the various baseline models. A more detailed description can be found in Doumpiotis et al. (2003b).

Word based HMMs were trained for each of the 36 words. The word models were left-to-right with approximately 20 states each, and had 12 mixtures per state. The data are parametrized as 13 dimensional MFCC vectors with first and second order differences. The baseline ML models were trained following

System	HMM Training Criterion	Segmented Data	HMMs cloned	HMM	SMBR-SVM	Voting
A	ML	N	N	10.70	-	-
B	MMI	N	N	9.07	8.10	7.76
C	ML	Y	N	9.67	7.94	-
D	ML	Y	Y	9.67	7.86	-
E	PLMMI	N	N	7.98	8.01	7.54

Table 1
WERs of various HMM and SMBR-SVM systems. A: Baseline HMMs trained under the ML criterion; B: The HMMs from A are trained using MMI; C: The HMMs from A were trained using Forward-Backward on only the confusable segments; D: The HMMs from A were cloned and tagged as illustrated in Fig. 1, d and were trained using Forward-Backward on only the confusable segments; E: The HMMs from B were MMI trained on the pinched lattices

the HTK-book (Young et al., 2000). The AT&T decoder (Mohri et al., 2001) was used to generate lattices on both the training and the test set. Since the corpus has no language model (each utterance is a random six word string), an unweighted free loop grammar was used during decoding. The ML baseline WER is 10.70%. MMI training was then performed (Normandin, 2002; Woodland and Povey, 2000) at the word level using word time boundaries taken from the lattices. A new set of lattices for both the training and the test sets was then generated using the MMI models. The WER was 9.07% and the Lattice Oracle Error Rate for these lattices was 1.27%. Period-1 lattice cutting was then performed on these lattices; the number of confusable words in each segment was further restricted to two. This increased the Lattice Oracle Error Rate to 3.11%. At this point there are two sets of confusion pairs from the pinched lattices: one set comes from the training data, and the other from the test data. We keep the 50 confusion pairs that are observed most frequently in the test data. All other confusion pairs in training and test data are pruned back to the truth and the MAP hypothesis respectively. We emphasize that this is a fair process; the truth is not used in identifying confusion in the test data.

Doumpiotis et al. (2003b) have found performing further MMI training of the baseline MMI models on the pinched lattices yields improvements. The performance of this Pinched Lattice MMI (PLMMI) system is listed in Table 1 as System E. We see a reduction in WER over the MMI models from 9.07% to 7.98%.

6.1 SMBR-SVM systems

SVMs were trained for the 50 dominant confusion pairs using the *Gini*SVM Toolkit (Chakrabartty, 2003) based on the lattices generated by the MMI system. The word time boundaries of the training samples were extracted from the lattices. The statistics needed for the SVM computation were found using the Forward-Backward procedure over these segments; in particular the mixture posteriors of the HMM observation distributions were found in this way. Log-likelihood ratio scores were generated from the 12 mixture MMI models and normalized by the segment length as described in Section 4.1.

We initially investigated score spaces constructed from both Gaussian mean and variance parameters. However training SVMs in this complete score space is impractical since the dimension of the score space is prohibitively large; the complete dimension is approximately 40,000. Filtering these dimensions based on Eq. (32) made training feasible, however performance was not much improved. We hypothesize that there is significant dependence between the model means and variances so that the underlying assumptions of the goodness criterion are violated.

We then used only the filtered mean sub-space scores for training SVMs (training on the unfiltered mean sub-space is still impractical because of the prohibitively high number of dimensions). The best performing SVMs used around 2,000 of the most informative dimensions, which is approximately 10% of the complete mean space. As shown in Table 1, applying SVMs to the MMI system (Table 1, System B) yields a significant 9.5% relative reduction in WER from 9.07% to 8.10%. This demonstrates that the SMBR-SVM system can be used to improve performance of MMI trained HMM continuous speech recognition systems.

6.2 Voting

In comparing the MMI and SMBR-SVM hypotheses to each other, we observed that they differ by more than 4%; this has been observed in some but not all previous work (Fine et al., 2001; Golowich and Sun, 1998; Smith et al., 2001). This suggests that hypothesis selection can produce an output better than each of the individual outputs. Ideally the voting schemes will be based on posterior estimates provided by each system. Transforming HMM acoustic likelihoods into posteriors is well established (Wessel et al., 1998). However we need to validate the posterior estimates of the SVM hypothesis as confidence scores. The quality of a confidence score can be measured by the Normalized Cross-Entropy (NCE) as used by Evermann and Woodland (2000).

$$NCE = \frac{H_{max} + \sum_{correct\ w} \log_2(\hat{p}(w)) + \sum_{incorrect\ w} \log_2(1 - \hat{p}(w))}{H_{max}} \quad (35)$$

where $H_{max} = -n \log_2(p_c) - (N - n) \log_2(1 - p_c)$

$n =$ # of correctly hypothesized words

$N =$ # of hypothesized words

$p_c =$ average probability that an output word is correct$(\frac{n}{N})$

$\hat{p}(w) =$ the confidence measure as a function of the output word w.

NCE is such that the higher the score the better the relative reliability of a system's confidence estimates. The NCE estimates for the *Gini*SVM output were encouraging in that they appeared as good as the HMM system NCE estimates.

We then performed system combination as described in Section 5.4. This combined system ('Voting' in Table 1) yielded improvements over the MMI system (7.76% *vs.*9.08%) and the result is even better than that of the PLMMI systems (7.76% *vs.*7.98). It was interesting to note that both the sum and the product scheme yielded the same output even to the level of individual word hypotheses.

6.3 Training Set Refinements for Code-Breaking

Table 1 presents the results for the baseline HMM systems. We now investigate the effect of training set refinement in acoustic Code-Breaking. We propose a technique that first identifies errors, then identifies training data associated with each error type, and finally applies models trained to fix those errors. We have shown that the use of SVMs improves over recognition with HMMs; however some of the improvement maybe due to training on these selected subsets.

We investigated the effect of retraining on the confusable data in the training set. Specifically, we performed supervised Forward-Backward re-estimation over the time bounded segments of the training data associated with all the error classes. We note that we take the confusion sets and their time boundaries from the MMI system for both training and test data; therefore these results are not directly comparable to the ML baseline (Table 1, System A). Simply by refining the training set in this way we found a reduction in WER from 10.70 to 9.67 (Table 1, System C). We then considered ML training a set of HMMs for each of the error classes; since there are 50 binary error classes, we added 100 models to the baseline model set. This is the most basic approach to Code-Breaking: we clone the ML-baseline models and retrain them over the

time bounded segments of the training data associated with each error class. The results of rescoring with these models are given in Table 1, System D. We see a reduction in WER from the 10.70% baseline to 9.67%. This is the same performance as System C, which was trained in the same way but without cloning. We conclude tentatively that some gains can be obtained simply by retraining the ML system on the confusable data selected from the training set.

6.4 Systems trained from PLMMI models

SVMs were also trained on the filtered mean only sub-space of the 12 mixture PLMMI models. The best performing SVMs in this case also used 10% of the most informative dimensions. While the performance was comparable to the PLMMI HMM system, we still do not improve upon it (8.01% *vs.* 7.98%). However, the same system combination scheme outlined above does produce significant gains over the PLMMI HMM system (7.54% *vs.* 7.98%).

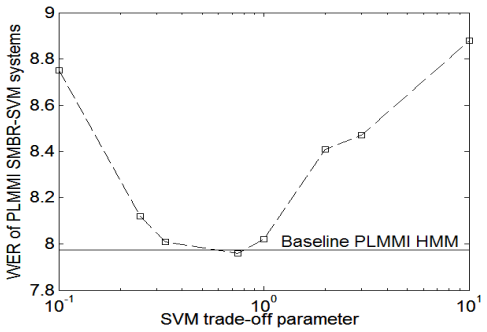

Fig. 4. WERs for different PLMMI SMBR-SVM systems as the global SVM trade-off parameter (C) is varied.

Finally, the effect of the SVM trade-off parameter (C in Eq. (26)) was studied. Fig. 4 presents the WER results from training the SVMs for the confusion pairs at different values of C. We find some sensitivity to C, however optimal performance was found over a fairly broad range of values (0.3 to 1.0).

All experiments reported thus far employ a global trade-off parameter value for the SVMs trained for the confusion pairs. We now investigate tuning the trade-off parameter for each SVM. The results in Table 2 show that further gains can be obtained by finding the optimal value of this parameter for each SVM. The oracle result is obtained by 'cheating' and choosing the parameter for each SVM that yields the lowest class error rate. An alternative systematic rule for choosing the parameter based on the number of training examples is presented in Table 3 where C decreases with the amount of training data. WER results using SVMs trained with the trade-off parameter set by this rule

	HMM	SMBR-SVM
PLMMI	7.98	8.01
Oracle	-	7.77
Piecewise C	-	7.88

Table 2
WERs for SMBR-SVM systems with trade-off parameter tuning.

N	$N > 10{,}000$	$N < 10{,}000$ $N > 5{,}000$	$N < 5{,}000$ $N > 500$	$N < 500$
C	0.33	0.75	1.0	2.0

Table 3
Piecewise Rule for choosing the trade-off parameter (C) through the number of training observations (N).

are presented in Table 2. By this tuning we find that the SVMS have the potential to improve over the PLMMI HMMs.

7 Extensions to Large Vocabulary Speech Recognition

We have studied a simple task so that we could develop the SMBR-SVM modeling framework and describe it without complications. Our ultimate goal is to apply this framework to large vocabulary speech recognition. Large vocabulary systems typically consist of sub-word models that are shared across words. We could apply the approach we have described thus far in a brute force manner by cloning the models in the original large vocabulary HMM system and retraining them over confusion sets. From systems B & C in Table 1 we saw there is value in retraining. However there are some drawbacks to such an approach.

7.1 Deriving SVM Score-Spaces through Constrained Parameter Estimation

Apart from the unwieldy size of a cloned system, the main problem would be data sparsity in calculating statistics for SVM training. This situation suggests the use of models obtained via constrained estimation. We can use Linear Transforms (LT) such as Maximum Likelihood Linear Regression (MLLR) (Legetter and Woodland, 1995) to estimate model parameters. Following the approach we have developed, these transforms are estimated over segments in the acoustic training set that were confused by the baseline system. We emphasize that the LTs are not used as a method of adaptation to test set data.

Consider the case of distinguishing between two words in a large vocabulary system. We need to construct models θ_1 and θ_2 from which we will produce the statistics needed to train an SVM. We identify all instances of this confusion pair G in the training set and estimate two transforms L_1 and L_2 relative to the baseline HMM system. These are trained via supervised adaptation *e.g.,* MLLR. One approach is to derive our Score-Space from the LT Score-Space is

$$\varphi(\mathbf{O}) = \begin{bmatrix} 1 \\ \nabla_{L_{G(1)}} \\ \nabla_{L_{G(2)}} \end{bmatrix} \ln \left(\frac{p(\mathbf{O}|L_{G(1)} \cdot \theta_{G(1)})}{p(\mathbf{O}|L_{G(2)} \cdot \theta_{G(2)})} \right). \tag{36}$$

The LT Score-Space has attractive qualities. By using regression classes we can control the dimensionality of the Score-Space. This will also allow us to address data sparsity problems by clustering together similar error patterns into regression classes. However, the Score-Space as found in Eq. (36) was unsuitable for classification. We hypothesize that since the LT scores can give no more than a direction in the HMM parameter manifold, the SMBR-SVM system cannot build effective decision boundaries in the LT Score-Space. When we inspected the kernel maps, we saw no evidence of the block diagonal structure which would indicate features useful for pattern classification.

An alternative is to create a constrained Score-Space by applying MLLR transforms to original models to derive a new set of models. Our Score-Space is the original mean Score-Space while the HMM parameters are modified by a LT estimated as described above. If $\theta'_{G(i)} = L_{G(i)} \cdot \theta_{G(i)}$ and $\theta' = [\theta'_{G(1)} \quad \theta'_{G(2)}]$ then,

$$\varphi(\mathbf{O}) = \begin{bmatrix} 1 \\ \nabla_{\theta'} \end{bmatrix} \ln \left(\frac{p(\mathbf{O}|\theta'_{G(1)})}{p(\mathbf{O}|\theta'_{G(2)})} \right). \tag{37}$$

Although intended for LVCSR, we investigated the feasibility of the approach in our small vocabulary experiments. The results are tabulated in Table 4. We estimate MLLR transforms with respect to the MMI models over the confusion sets. A single transform was estimated for each word hypothesis in each confidence set. We then apply the transforms to the MMI models and estimate statistics as described in Eq. (37). The performance is shown in Table 4, System B. We see a reduction in WER with respect to the MMI baseline from 9.07% to 8.00%. We conclude that the severely constrained estimation is able to generate Score-Spaces that perform at similar WERs to those of unconstrained estimation. For completeness, we rescored the confusions sets using the transformed MMI models. As can be expected performance degrades

System	HMMs Used	HMM	SMBR-SVM
A	MMI	9.07	8.10
B	MMI+MLLR	9.35	8.00

Table 4
WERs of HMM systems with and without MLLR transforms; SMBR-SVM systems were trained in the Score-Space of the transformed models

slightly from 9.07% to 9.35% suggesting that performing ML estimation subsequent to MMI estimation undoes the discriminative training (Normandin, 1995).

8 Conclusions

We have developed a Code-Breaking framework that applies Support Vector Machines in continuous speech recognition. We use available baseline HMM models for the identification of confusable regions, train error specific SVMs, and attempt to resolve the remaining confusion in the test data using the error specific models.

Our framework uses lattice cutting techniques to convert the continuous ASR problem into a sequence of independent but coupled classification problems. We used the previously proposed technique of Score-Spaces to convert the variable length acoustic sequences associated with the problems into fixed dimensional vectors which can then be classified by SVMs.

We posed the estimation of a posterior distribution over hypothesis in the confusable regions as a logistic regression problem. We showed that *Gini*SVMs can be used as an approximation technique to estimate the parameters of the logistic regression problem. We also found significant improvements by using tanh kernels over other kernels that have been studied for ASR. We conjecture that this is due to the ability of *Gini*SVMs to incorporate non-positive-definite kernels in its training.

We investigated several methods to compute the sufficient statistics required to generate scores. While the approaches performed similarly on the problems we study, we noted different aspects of their implementation that may make them more appropriate choices for LVCSR. We see considerable improvement in the performance of SVMs through selection of the most informative score-space dimensions, as has been noted (Smith and Gales, 2002). We suspect this to be an artifact of the approximation to the scaling matrix. If improved normalization of the Score-Space is found either through better numerical

methods or an improved modeling formulation, the SMBR-SVM formulation should be expected to yield further improvements.

We find that confidence measures over hypotheses can be robustly produced by *Gini*SVMs. This allows for hypothesis selection from the baseline and the SVM system using a weighted voting scheme. We further found that SMBR-SVM rescoring performed significantly better than MMI and using the voting schemes we obtained significant improvements over another form of discriminative training, namely PLMMI.

We have identified two components to the gains that we find in this use of SVMs. The first contribution comes from the refinement of the training data. The baseline models themselves can be improved by training over confusable data identified by lattice cutting. The second contribution comes from the use of SVMs themselves.

Our ultimate goal is to apply our new framework to LVCSR. We have discussed some of the problems we expect to encounter and have proposed and investigated constrained estimation techniques that will allow us to derive features for SVMs when training data is scarce.

We have introduced a new framework that incorporates the benefits of HMMs and improves upon their performance. The promise of this framework is that it allows us to explore the application of new modeling techniques to continuous speech recognition without having to address all aspects of that large and complex problem.

Acknowledgments We would like to thank Gert Cauwenberghs for helpful suggestions. Baseline MMI and PLMMI models were trained by Vlasios Doumpiotis and the ML models were trained by Teresa M. Kamm. We thank Mehryar Mohri for use of the AT&T large vocabulary decoder. Veera Venkataramani thanks Shankar Kumar, Peng Xu and Yonggang Deng for helpful discussions.

References

Blum, A., Langley, P., 1997. Selection of relevant features and examples in machine learning. Artificial Intelligence 97 (1-2), 245–271.

Boser, B., Guyon, I., Vapnik, V., 1992. A training algorithm for optimal margin classifier. In: Proc. 16th Conf. Computational Learning Theory. pp. 144–152.

Burges, C. J., Schölkopf, B., 1997. Improving the accuracy and speed of support vector learning machines. In: Mozer, M., Jordan, M., Petsche, T. (Eds.), Advances in Neural Information Processing Systems 9. Cambridge: MIT Press, pp. 375–381.

Chakrabartty, S., 2003. The giniSVM toolkit, Version 1.2. Available: http://bach.ece.jhu.edu/svm/ginisvm/.

Chakrabartty, S., Cauwenberghs, G., 2002. Forward decoding kernel machines: A hybrid HMM/SVM approach to sequence recognition. In: Proc. SVM'2002, Lecture Notes in Computer Science. Vol. 2388. Cambridge: MIT Press, pp. 278–292.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Machine Learning 20 (3), 273–297.

Doumpiotis, V., Tsakalidis, S., Byrne, W., 2003a. Discriminative training for segmental minimum Bayes risk decoding. In: Proc. ICASSP. Hong Kong.

Doumpiotis, V., Tsakalidis, S., Byrne, W., 2003b. Lattice segmentation and minimum Bayes risk discriminative training. In: Proc. Eurospeech. Geneva.

Drucker, H., Burges, C. J., Kaufman, L., Smola, A., Vapnik, V., 1997. Support vector regression machines. In: Advances in neural Information Processing Systems 9. Cambridge: MIT Press, pp. 155–161.

Evermann, G., Woodland, P. C., 2000. Large vocabulary decoding and confidence estimation using word posterior probabilities. In: Proc., ICASSP.

Fine, S., Navrátil, J., Gopinath, R., 2001. A hybrid GMM/SVM approach to speaker identification. In: ICASSP. Utah, USA.

Ganapathiraju, A., Hamaker, J., Picone, J., 2003. Advances in hybrid SVM/HMM speech recognition. In: GSPx / International Signal Processing Conference. Dallas, Texas, USA.

Goel, V., Byrne, W., 2000. Minimum Bayes-risk automatic speech recognition. In: Computer Speech & Language. Vol. 14(2).

Goel, V., Byrne, W., 2003. Minimum Bayes-risk automatic speech recognition. In: Chou, W., Juang, B.-H. (Eds.), Pattern Recognition in Speech and Language Processing. CRC Press.

Goel, V., Kumar, S., Byrne, W., 2001. Confidence based lattice segmentation and minimum Bayes-risk decoding. In: Proc. Eurospeech. Aalborg, Denmark.

Goel, V., Kumar, S., Byrne, W., 2003. Segmental minimum Bayes-risk decoding for automatic speech recognition. IEEE Transactions on Speech and Audio Processing *to appear*.

Golowich, S. E., Sun, D. X., 1998. A support vector/hidden Markov model approach to phoneme recognition. In: ASA Proceedings of the Statistical Computing Section. pp. 125–130.

Jaakkola, T., Haussler, D., 1998. Exploiting generative models in discriminative classifiers. In: M. S. Kearns, S. A. S., Cohn, D. A. (Eds.), Advances in Neural Information Processing System. MIT Press.

Jelinek, F., February 1996. Speech recognition as code-breaking. Tech. Rep. Tech Report No. 5, CLSP, JHU.

Kumar, S., Byrne, W., 2002. Risk based lattice cutting for segmental minimum Bayes-risk decoding. In: ICSLP. Denver, Colorado, USA.

Lafferty, J., McCallum, A., Pereira, F., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML.

Legetter, C. J., Woodland, P. C., 1995. Maximum likelihood linear regression for speaker adaptation of continuous density hmms. In: Computer, Speech and Language. Vol. 9. pp. 171–186.

Mohri, M., Pereira, F., Riley, M., 2001. AT&T General-purpose Finite-State Machine Software Tools. Available: http://www.research.att.com/sw/tools/fsm/.

Noel, M., 1997. Alphadigits. CSLU, OGI, Available: http://www.cse.ogi.edu/CSLU/corpora/alphadigit.

Normandin, Y., 1995. Optimal splitting of HMM gaussian mixture components with mmie training. In: Proc. ICASSP. Vol. 15. pp. 449–452.

Normandin, Y., 2002. Maximum mutual information estimation of hidden Markov models. In: M. S. Kearns, S. A. S., Cohn, D. A. (Eds.), Automatic Speech and Speaker Recognition. Vol. 15. Cambridge: MIT Press.

Salomon, J., King, S., Osburne, M., 2002. Framewise phone classification using support vector machines. In: Proc. ICSLP.

Smith, N. D., Gales, M. J. F., April 2002. Using SVMs to classify variable length speech patterns. Tech. Rep. CUED/F-INFENG/TR412, Cambridge University Eng. Dept.

Smith, N. D., Gales, M. J. F., Niranjan, M., April 2001. Data-dependent kernels in SVM classification of speech patterns. Tech. Rep. CUED/F-INFENG/TR387, Cambridge University Eng. Dept.

Vapnik, V., 1995. The Nature of Statistical Learning Theory. Springer-Verlag, New York, Inc., Ch. 5.

Wessel, F., Macherey, K., Schlueter, R., 1998. Using word probabilities as confidence measures. In: Proc. ICASSP. Seattle, WA, USA, pp. 225–228.

Weston, J., Watkins, C., May 1998. Multi-class support vector machines. Tech. Rep. CSD-TR-9800-04, Department of Computer Science, University of London.

Woodland, P. C., Povey, D., 2000. Large scale discriminative training for speech recognition. In: Proc. ITW ASR, ISCA.

Young, S., Evermann, G., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P., July 2000. The HTK Book, Version 3.0.