Modeling Prior Belief for Speaker Verification SVM Systems

Luciana Ferrer[1,2]

[1]Department of Electrical Engineering, Stanford University, Stanford, CA, USA
[2]Speech Technology and Research Laboratory, SRI International, Menlo Park, CA, USA
lferrer@stanford.edu

Abstract

Support vector machines (SVMs) can be interpreted as a maximum a posteriori (MAP) estimation of a model's parameters, for an appropriately chosen likelihood function. In the standard formulation for SVM classification and regression problems, the prior distribution on the weight vector is implicitly assumed to be a multidimensional Gaussian with zero mean and identity covariance matrix. In this paper we propose to relax the assumption that the covariance matrix is the identity matrix, allowing it to be a more general block diagonal matrix. In speaker verification, this covariance matrix can be estimated from held-out speakers. We show results on two speaker verification systems: a Maximum Likelihood Linear Regression (MLLR)-based system and a prosodic system. In both cases, the proposed prior model leads to more than 10% improvement in equal error rate (EER) with respect to results obtained using the standard prior assumptions.

Index Terms: Support Vector Machines, Kernels, Speaker Recognition, Speaker Verification.

1. Introduction

SVMs can be interpreted as a MAP estimation of a model's parameters given certain assumptions on the form of the likelihood function of the data and the prior probability distribution of the parameters [1]. In particular, the prior on the SVM weight vector is implicitly assumed to be given by $\mathcal{N}(0, I)$, that is, a normal distribution with zero mean and identity covariance matrix. We can easily imagine cases in which this assumption is not a very reasonable one. For example, when input features are correlated with each other it would be more natural to assume that the resulting weights will also be correlated.

In [2], we proposed a method that effectively modifies the assumptions about the prior distribution of the weight vector. We consider the case in which the input features to the SVM are spatially related in some underlying metric space. A measure of smoothness of the weight vector in this space is used to further regularize the parameter search. This is equivalent to defining a new prior covariance matrix for the weight vector, using the knowledge of the spatial structure of the underlying features. In [2] we showed significant gains from the application of this method to a prosodic speaker verification system. Unfortunately, the method cannot be applied to all SVM speaker verification systems, since it relies on the assumption that the features are spatially related.

The method presented in this paper is another attempt at relaxing the assumptions on the prior distribution of the weight vector. In this case, no assumptions are made on the structure of the feature vector. Instead, the covariance matrix for the prior distribution of the weight vector is estimated from a set of held-out models trained with samples from several distinct speakers. As we will see, in order for the method to be successful held-out models have to be trained using several target samples. That is, we require a held-out database similar to that used for session variability compensation purposes, where many samples are available for each target speaker. As long as such a database is available, the method is quite general and may apply to any speaker verification system in which SVMs are used as target models.

We present results on a speaker verification task for two systems that use MLLR and prosodic features as input. We show that the proposed method gives around 10% relative EER improvement for these systems.

2. Support Vector Machines

Consider a training set with m samples, $S = \{(x_i, y_i) \in \mathcal{R}^d \times \{-1, +1\}; i = 1, ..., m\}$, where x_i are the features and y_i is the class corresponding to sample i. Our goal is to find a function $f(x) = w^T x + b$, such that sign$(f(x))$ is the predicted class for feature vector x. The SVM formulation for classification is given by (see, e.g., [3]):

$$\begin{array}{lll} \text{minimize} & J(w, \epsilon) = \frac{1}{2} w^T w + C \sum_{i=1}^{m} \epsilon_i & \\ \text{subject to} & y_i(w^T x_i + b) \geq 1 - \epsilon_i & i = 0, ..., m \\ & \epsilon_i \geq 0 & i = 0, ..., m \end{array} \quad (1)$$

Minimizing the norm of the weight vector is equivalent to maximizing the margin between the samples and the hyperplane. The *slack* variables ϵ_i allow for some samples to be at a distance smaller than the margin from the separating hyperplane or even on the wrong side. The parameter C controls the trade-off between the size of the margin and the total amount of error. By deriving the dual form of the optimization problem above we find that input vectors appear only as inner products with each other. Hence, each inner product between input features can be replaced with a function $K(x_i, x_j) = \phi(x_i)^t \phi(x_j)$ called the kernel function, where $\phi(x)$ is a transform of the input features.

The SVM problem can also be expressed as a maximization of

$$l(w, b) = -\frac{1}{2} w^T w - C \sum_{i=1}^{m} h(y_i(w^T x_i + b)) \quad (2)$$

where $h(z)$ is the *hinge loss*, given by $(1-z)u(1-z)$ with $u(z)$ the Heaviside step function. This expression can be interpreted as the log-posterior probability for the parameters w and b given the data. To see this, we write the posterior probability as

$$P(w, b|S) \propto P(w, b) \prod_{i=1}^{m} \mathcal{P}(y_i|x_i, w, b) \quad (3)$$

where we have assumed that class labels are independently drawn from a distribution $P(y_i|x_i, w, b)$ and that features x_i are independent of parameters w and b. If we assume that w's prior distribution is $\mathcal{N}(0, I)$ and b's is uniform and that w and b are independent, then $\log(P(w, b))$ reduces to $-\frac{1}{2}w^T w + K_1$, where K_1 is a constant. If we appropriately define $P(w, b)$ (see [1] for more details on this) we get $l(w, b) = log(P(w, b|S)) + K_2$, where K_2 is constant with respect to w and b. Hence, the SVM problem (1) is equivalent to the maximization of (3).

In summary, when training an SVM for classification we are implicitly assuming a Gaussian prior on the weight vector, with zero mean and identity covariance matrix. That is, weights are, a priori, assumed to have equal variance and be independent of each other.

The above setup corresponds to a classification problem. The regression problem can also be posed as a convex optimization problem by choosing an appropriate distance measure [3, 4] with the objective function given by the sum of the square norm of the weight vector and an error term, as in the classification case. The dual of this problem again takes a form in which features appear only in inner products with other features. Furthermore, the interpretation of the SVM as a MAP estimation still holds given an appropriate choice of likelihood function [1]. Hence, even though the development in Section 3 will be done considering a classification problem for simplicity, the method described and the interpretations given are equally applicable to SVM regression problems.

3. Changing the Prior Assumption

Assume now that we have knowledge of the covariance matrix of the weight vector w and that it is not the identity matrix. In this case the SVM problem can be changed to include this prior knowledge to

$$\begin{array}{lll} \text{minimize} & J'(w, \epsilon) = \frac{1}{2} w^T \Sigma^{-1} w + C \sum_{i=1}^{m} \epsilon_i & \\ \text{subject to} & y_i(w^T x_i + b) \geq 1 - \epsilon_i & i = 0, ..., m \\ & \epsilon_i \geq 0 & i = 0, ..., m \end{array} \tag{4}$$

which assumes a prior on w given by $\mathcal{N}(0, \Sigma)$. We further assume that Σ is positive definite. As we describe in Section 3.1, in order to estimate Σ robustly we use a regularization procedure that effectively makes Σ positive definite.

The new objective function $J'(w, \epsilon)$ is still convex since the matrix Σ^{-1}, being the inverse of a positive definite matrix, is also positive definite. This new problem can be cast as a standard linear SVM problem by taking the change of variable $\tilde{w} = Bw$, with $B^T B = \Sigma^{-1}$ (B is a matrix square root of Σ^{-1} and can be chosen to be real and symmetric since Σ^{-1} is real, positive definite and symmetric). We can write the new optimization problem as

$$\begin{array}{lll} \text{minimize} & J'(\tilde{w}, \epsilon) = \frac{1}{2} \tilde{w}^T \tilde{w} + C \sum_{i=1}^{m} \epsilon_i & \\ \text{subject to} & y_i(\tilde{w}^T \tilde{x}_i + b) \geq 1 - \epsilon_i & i = 0, ..., m \\ & \epsilon_i \geq 0 & i = 0, ..., m \end{array} \tag{5}$$

where $\tilde{x}_i = B^{-1} x_i$. Comparing this expression with (1) we see that we have obtained a new SVM problem where the features are now the $\tilde{x}_i$'s. We can then implement this as an SVM problem on the original features using a kernel given by

$$K(x_i, x_j) = x_i^T B^{-1} B^{-1} x_j = x_i^T \Sigma x_j. \tag{6}$$

or we can transform the features using $\tilde{x}_i = B^{-1} x_i$ and train the SVM using the inner product kernel.

3.1. Estimating Σ

To obtain an empirical estimation of the covariance matrix Σ, we take a set of held-out data from several target speakers and train one model for each speaker, obtaining a set of weight vectors. Using these weight vectors we can estimate the covariance matrix. To obtain a robust estimation of the covariance matrix we use the *corpcor* R package [5]. This package implements a shrinkage approach for inferring the covariance matrix. The covariance matrix is computed on the basis of the shrunken correlation matrix and the shrunken variances. Empirical variances for each component are computed and then shrunk toward their median. Similarly, the empirical correlation matrix is shrunk toward the identity matrix. The shrinkage intensity for the variances and the correlation matrix, λ_{var} and λ_{cor}, respectively, should be tuned. Their optimal values might depend on the input features to the SVM, on the amount of held-out target models available to estimate the covariance matrix, and so on. The estimated covariance matrix will be positive definite as long as λ_{var} is larger than 0.

When the dimension of the features and, hence, that of the weight vector, is large, we may need to impose some structure on the covariance matrix in order to estimate it robustly. In this paper we restrict the covariance matrix to be block diagonal, with blocks defined depending on the structure of the system's features. Each block in the block diagonal matrix is then estimated using the procedure explained above. A single pair of λ_{var} and λ_{cor} is chosen for all blocks in the matrix.

4. Experiments

Experiments were conducted on a text-independent speaker verification task. The goal is, given a speech sample and a claimed speaker identity, to decide whether the claim is true (the speaker is the target speaker) or false (the speaker is an impostor). This is a binary classification task for which SVMs have repeatedly been found to outperform other classification methods.

4.1. Databases and Error Measures

Experiments were conducted using data from the NIST speaker recognition evaluations (SRE) from 2005 and 2006. The data from 2005 was used for parameter tuning. Each speaker verification trial consists of a test sample and a speaker model. The samples are one side of a telephone conversation with approximately 2.5 minutes of speech per side. We consider the 1-side training condition in which we are given a single sample to train the speaker model. This sample corresponds to a positive example when training the SVM model for the speaker. The data used as negative examples for the SVM training is taken from 2003 and 2004 NIST evaluations along with some FISHER data, resulting in a total of 4,394 conversation sides. The SRE2005 and SRE2006 tasks contain 26,270 and 24,013 trials, respectively.

The data used to obtain the Σ matrix was drawn from the 2004 NIST evaluation data. 2402 conversation sides were used from this database, including 206 distinct speakers, giving an average of 11.7 conversation sides per speaker.

The performance measures used in this paper are equal error rate (EER) and NIST's minimum detection cost function (DCF), defined as the Bayesian risk with probability of target equal to 0.01, cost of false alarm equal to 1, and cost of miss equal to 10. SVMlight [6] is used to train the SVM models.

4.2. Systems

We implement the proposed method on two systems, an MLLR-based and a prosodic system.

The **MLLR** system [7] uses the speaker adaptation transforms used in the speech recognition system as features for speaker verification. A total of 16 affine 39x40 transforms are used to map the Gaussian mean vectors from speaker-independent to speaker-dependent speech models; eight transforms each are estimated relative to male and female recognition models, respectively. The transforms are estimated using MLLR, and can be viewed as a text-invariant encapsulation of the speaker's acoustic properties. The transform coefficients form a 24,960-dimensional feature space. Each feature dimension is rank-normalized by replacing the value with its rank in the background data and scaling the ranks to lie in the interval [0, 1]. This normalization method was found to be better than variance normalization [8]. The resulting normalized feature vectors are then modeled by SVMs trained to perform classification using a linear kernel. Parameter C is set to 1 for impostor samples and to 500 for target samples, to compensate for the imbalance in the amount of samples from the two classes.

Nuisance attribute projection (NAP) [9] is used to improve the performance of this system. Directions of session variability are estimated from data containing several conversations per speaker. The first N eigenvectors of the within-speaker covariance matrix are assumed to correspond to *noisy* directions. These directions are then eliminated from the feature vectors, resulting in (it is hoped) "session-independent" features.

The **SNERF** system models syllable-based prosodic features (or more generally, NERFs, nonuniform extraction region features) [10]. Features are based on estimated F0, energy, and duration information extracted over syllables inferred via automatic syllabification based on automatic speech recognition output. A total of 140 features are extracted for each syllable. These sequences of vectors (one vector for each syllable in the sample) are transformed into single fixed-length vectors (one per sample) via a particular implementation of the Fisher score [11]. A Gaussian mixture model (GMM) is obtained for each feature and sequence of features for two and three consecutive syllables or pauses. This allows us to model temporal dependencies, which we would ignore otherwise. We call each of these sequences a *token*. Given a sequence of prosodic features from a sample, the transform is composed by the posterior probabilities of each Gaussian in the GMM of each token. These posterior probabilities are concatenated and rank-normalized (as for the MLLR features) into the final vector x_i. SVMs are then trained to perform regression on the class labels (which was found to be better than classification for these features). The C parameter is set to its default value (the inverse of the average norm of the train feature vectors) for the impostor samples and 500 times that value for the target samples.

The resulting transformed features are spatially related, since each of them corresponds to a certain Gaussian in the GMM of a token. The smoothing method proposed in [2] can then be applied. This method takes into account the spatial structure of the features to implicitly impose a prior distribution on the weight vector of the SVM. We refer to this method as the *smoothing kernel* method. In [11] we compared two methods for training the GMMs with respect to which the transform from sequences to fixed-length vectors is performed: EM and VQ. In this paper we use the VQ method, since, after smoothing, this method slightly outperforms the EM method.

4.3. Results

Table 1 shows the results for the SNERF system. The table compares the prior kernel proposed in this paper with the smoothing kernel presented in [2]. The prior kernel is estimated using the weight vectors from 206 speaker models trained on the held-out data using all conversations available for each speaker (an average of 11.7 conversation sides per speaker). Since these conversations are also present in the set of impostor samples it is sometimes necessary (depending on the SVM parameter setting) to ignore the target samples from the impostor set when training each of the target models.

The smoothing kernel for these features can be obtained only per token, since it relies on the spatial structure between the transformed features, which can be defined only for features that correspond to the same underlying GMM. On the other hand, the blocks used to estimate Σ can be freely chosen. We show two reasonable choices: one block per token (as for the smoothing kernel) and one block for each feature sequence of a certain length. The second case is then more general than the first one. We found that going beyond this second option (for example, by having one block for all transformed features corresponding to all GMMs for the same prosodic features) led to worse results than the two options presented here. Table 1 shows no significant difference between the smoothing kernel and the prior kernel with the second choice of covariance matrix structure. In both cases a significant gain (p-value smaller than 0.001) of 13% in EER is achieved with respect to the baseline. The optimal λ parameters, tuned on SRE05 trials, are $\lambda_{var} = 1$ and $\lambda_{cor} = 0$; that is, no shrinkage is done on the correlation matrix and complete shrinkage is done on the variances (this results on all variances being equal to the median of their empirical estimates). We conclude then that all components of the weight vector should be equally important a priori, but that the correlations between them should not be assumed zero as in the standard SVM formulation.

The prior kernel offers the advantage over the smoothing kernel of being applicable to more general speaker verification systems. Table 2 shows the results for the MLLR-SVM system, one system for which the smoothing kernel is not (at least directly) applicable. A more detailed analysis is presented in this case. First, results on the system without NAP are shown. We see a significant gain of around 9% in EER when the prior kernel is applied. The next block of results uses features where 128 NAP directions have been eliminated. Since NAP is supposed to eliminate the directions in the feature space that vary with session, resulting in less noisy features, and since the same data used to estimate the NAP directions (a set of 206 SRE04 speakers) is used to estimate the prior kernel, it would be reasonable to think that once NAP is applied to the system, the gain obtained with prior modeling could be reduced or disappear. Fortunately, this is not the case. When we use all conversations available for the 206 speakers to estimate the prior kernel (last row in the table), we obtain a significant gain (p-value smaller than 0.001) of around 10% in EER for SRE06.

The λ parameters obtained by tuning on SRE05 are $\lambda_{var} = 0.25$ and $\lambda_{cor} = 0.90$. In the extreme, when $\lambda_{var} = 0$ and $\lambda_{cor} = 1$, the matrix Σ is diagonal, with diagonal components given by s_j^2, the empirical variance for weight w_j. In this case, $w^T \Sigma^{-1} w = \sum_j (w_j / s_j)^2$, that is, the size of each weight is divided by its a priori standard deviation. If a certain component j has a small standard deviation (meaning that its value was always close to zero in the held-out target models), a large value of w_j will be highly penalized. Hence, when λ_{cor} is close to 1

	SRE05		SRE06	
	EER	DCF	EER	DCF
Inner product kernel	13.27	0.518	12.84	0.566
Smoothing kernel per token	12.18	0.462	11.16	0.506
Prior kernel per token	12.14	0.472	11.65	0.502
Prior kernel per feature and sequence length	11.65	0.462	11.11	0.501

Table 1: Comparison of SNERF systems. Prior kernels are estimated using 206 speakers and all conversations available for each speaker (an average of 11.7 conversations per speaker).

		SRE05		SRE06	
		EER	DCF	EER	DCF
Without NAP	Inner product kernel	5.42	0.164	3.61	0.177
	Prior kernel, 206 speaker, 11.7 conv/speaker on ave	4.98	0.157	3.29	0.176
With NAP	Inner product kernel	5.06	0.136	3.18	0.155
	Prior kernel, 206 models, 1 conv/model	5.38	0.150	3.45	0.163
	Prior kernel, 206 models, 4 conv/model	5.14	0.140	3.24	0.155
	Prior kernel, 206 models, 8 conv/model	4.69	0.134	2.91	0.151
	Prior kernel, 1648 models, 1 conv/model	5.34	0.147	3.34	0.159
	Prior kernel, 206 models, 11.7 conv/model on ave	4.65	0.129	2.86	0.149

Table 2: Comparison of MLLR systems. The number of models used to estimate Σ and the number of conversations used to train each model are indicated. In all cases the Σ matrix is block diagonal with 16 blocks corresponding to each of the MLLR transforms.

and λ_{var} close to 0, we are restricting components that were not useful in the held-out set to have small values. Since the held-out models were trained with many conversation sides from the target speaker (a minimum of 9), this prior knowledge should be valuable when training a new model with a single conversation side from the target speaker.

To test whether it is in fact necessary to train the held-out models using several conversations per speaker, we tried four other conditions: training 206 models with 1, 4, and 8 conversation sides per speaker; and training 8*206=1648 models with 1 conversation side for each model, using the same conversations used to train the 206 models with 8 conversation sides each. To make results comparable, we kept the λ parameters fixed at the values mentioned above, which were optimized for the last row in the table. For all cases in this table, except when using 8 or more conversations sides to train each model, the optimal value of the λ parameters was 1, resulting in a diagonal covariance matrix proportional to the identity, in which case the inner product kernel results are recovered. Hence, we see that until at least 8 conversation sides per speaker are used to train each held-out model, no gain is obtained from prior modeling. This supports the intuition that in order for this procedure to be successful the held-out models must be good models. That is, we want to compute the prior covariance of the weight vector when the models are close to what they would be if we had infinite data from the speaker. This way, when little data is available for training, the prior kernel will push the weight vector toward what we would expect it to be if we had much more data available from that speaker.

5. Conclusions

When training a linear SVM, an implicit assumption is made that the prior distribution of the weight vector is Gaussian with zero mean and identity covariance matrix. In this paper we relax this assumption by allowing more general covariance matrices. This new problem can be easily implemented as a linear transform of the features or as a kernel of the form $x_i^T \Sigma x_j$. For a speaker verification task, the required covariance matrix can be estimated from a set of held-out speakers from whom several conversations are available, such that good models can be trained for each speaker. We show results for a state-of-the-art MLLR-based system and a prosodic system. In both cases the proposed kernel results in a gain of around 10% in EER with respect to the inner product kernel.

6. Acknowledgments

We thank Joseph Campbell for the initial discussion that inspired this work. This work was supported by NSF IIS-0544682 at SRI International and NSF CNS-0652510 at Stanford University. The views herein are those of the author and do not necessarily represent the views of the funding agency.

7. References

[1] P. Sollich, “Probabilistic interpretation and Bayesian methods for support vector machines,” in *Proc. ICANN*, Edinburgh, Sept. 1999, pp. 91–96.

[2] L. Ferrer, K. Sonmez, and E. Shriberg, “A smoothing kernel for spatially related features and its application to speaker verification,” in *Proc. Interspeech*, Antwerp, Aug. 2007.

[3] V. Vapnik, *The nature of statistical learning theory*, Springer, 1999.

[4] A. Smola and B. Schölkopf, “A tutorial on support vector regression,” *NeuroCOLT2 Technical Report NC2-TR-1998-030*, 1998.

[5] J. Schaefer, R. Opgen-Rhein, and K. Strimmer, “corpcor: Efficient estimation of covariance and (partial) correlation,” http://cran.r-project.org/web/packages/corpcor/.

[6] T. Joachims, “Making large-scale SVM learning practical,” in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. 1999, MIT Press.

[7] A. Stolcke, S. S. Kajarekar, L. Ferrer, and E. Shriberg, “Speaker recognition with session variability normalization based on MLLR adaptation transforms,” *IEEE Trans. Audio, Speech, and Lang. Process.*, Sept. 2007.

[8] A. Stolcke, S. Kajarekar, and L. Ferrer, “Nonparametric feature normalization for svm-based speaker verification,” in *Proc. ICASSP*, Las Vegas, Apr. 2008.

[9] W. Campbell, “Compensating for mismatch in high-level speaker recognition,” in *Proc. Odyssey-06*, Puerto Rico, USA, June 2006.

[10] E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, and A. Stolcke, “Modeling prosodic feature sequences for speaker recognition,” *Speech Communication*, vol. 46, no. 3-4, pp. 455–472, 2005.

[11] L. Ferrer, E. Shriberg, S. Kajarekar, and K. Sönmez, “Parameterization of prosodic feature distributions for SVM modeling in speaker recognition,” in *Proc. ICASSP*, Honolulu, Apr. 2007.