Within-Class Covariance Normalization for SVM-based Speaker Recognition

Andrew O. Hatch[1,2], *Sachin Kajarekar*[3], *and Andreas Stolcke*[1,3]

[1]The International Computer Science Institute, Berkeley, CA, USA
[2]The University of California at Berkeley, USA
[3]SRI International, Menlo Park, CA, USA
ahatch@icsi.berkeley.edu, sachin@speech.sri.com, stolcke@speech.sri.com

Abstract

This paper extends the within-class covariance normalization (WCCN) technique described in [1, 2] for training generalized linear kernels. We describe a practical procedure for applying WCCN to an SVM-based speaker recognition system where the input feature vectors reside in a high-dimensional space. Our approach involves using principal component analysis (PCA) to split the original feature space into two subspaces: a low-dimensional "PCA space" and a high-dimensional "PCA-complement space." After performing WCCN in the PCA space, we concatenate the resulting feature vectors with a weighted version of their PCA-complements. When applied to a state-of-the-art MLLR-SVM speaker recognition system, this approach achieves improvements of up to 22% in EER and 28% in minimum decision cost function (DCF) over our previous baseline. We also achieve substantial improvements over an MLLR-SVM system that performs WCCN in the PCA space but discards the PCA-complement.

Index Terms: kernel machines, support vector machines, feature normalization, generalized linear kernels, speaker recognition.

1. Introduction

In recent years, support vector machines (SVMs) have become one of the most important and widely-used classification techniques within the field of speaker recognition. Most top-performing speaker recognition systems use output "scores" obtained from SVM-based speaker models to arrive at a final decision for a given speaker trial. As with every SVM-based classifier, these speaker models are trained using some predefined kernel function k. Proper selection of the kernel function can be critical to the success of an SVM-based system, particularly in cases where the amount of available training data for either the impostor class or the target speaker class is very limited (e.g. the 1-conversation training condition in speaker recognition).

With some exceptions (e.g. the rank normalization technique described in [3, 4]), most of the existing work on kernel selection for speaker recognition has focused on *generalized linear kernels*—that is, kernels of the form, $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{R} \mathbf{x}_2$, where $\mathbf{R}$ is a positive semidefinite parameter matrix. Approaches for training $\mathbf{R}$ include the technique described in [5], which essentially involves setting $\mathbf{R}$ equal to $\mathbf{C}^{-1}$, where $\mathbf{C}$ is the covariance matrix of the training data. A diagonal parameterization for $\mathbf{R}$ is derived in [6] for count-based features (e.g. phone n-grams). These parameterizations have both yielded substantial improvements over other kernels on a variety of speaker recognition tasks and feature sets. Nonetheless, both parameterizations are somewhat limited by the fact that they are *unsupervised*—that is, they do not take speaker labels into account when training $\mathbf{R}$. This limitation is addressed, at least partially, by Solomonoff et al. in [7] and in [8], where the authors use speaker labels to identify orthonormal vectors or "directions" in feature space that maximize task-relevant information while minimizing noise. Solomonoff's approach has been shown to be quite useful for filtering out channel noise and for performing feature reduction. However, the approach in [7, 8] does not prescribe any scheme for weighting the directions in feature space that are retained. Thus, this approach does not fully answer the question of how to train $\mathbf{R}$ for a generalized linear kernel.

In this paper, we expand on the *within-class covariance normalization* (WCCN) technique for training generalized linear kernels that was recently introduced in [1, 2]. The WCCN technique prescribes setting $\mathbf{R}$ equal to $\mathbf{W}^{-1}$, where $\mathbf{W}$ is the expected within-class covariance matrix over all classes (i.e. speakers) in the training data. WCCN uses information about class labels from the training data to identify orthonormal directions in feature space that maximize task-relevant information. However, unlike other techniques in the literature, WCCN optimally weights each of these directions to minimize a particular upper bound on error rate [1, 2]. Thus, the WCCN approach can, in principle, harness whatever task-relevant information is contained in each of the "directions" of the underlying feature space—even directions that are largely dominated by noise.

We describe a set of experiments where we combine WCCN with a version of the principal component analysis (PCA) technique described in [9]. Our algorithm provides a practical approach for applying WCCN to large feature sets, where inverting or simply estimating $\mathbf{W}$ is impractical for computational reasons. In experiments on SRI's latest MLLR-SVM speaker recognition system (i.e. feature set), our combined WCCN approach achieves relative improvements of up to 22% in equal-error rate (EER) and 28% in minimum DCF below SRI's previous baseline.

The paper is organized as follows: In section 2, we summarize the WCCN approach and discuss practical considerations for how to apply WCCN to large feature sets. In section 3, we describe the approach used in [9] for breaking feature vectors down into PCA and PCA-complement components. This is followed by section 4, where we describe the experimental procedure that we use to perform feature normalization and to train SVM-based speaker models. Finally, in sections 5 and 6, we describe a set of experiments, provide results, and end with a set of conclusions.

2. Within-Class Covariance Normalization

The concept of *within-class covariance normalization* (WCCN) for SVM training was recently introduced in [1] and then extended in [2]. To derive the WCCN approach, the authors first construct a set of upper bounds on the rates of false positives and false negatives in a linear classifier (i.e. a binary classifier that uses a linear or affine decision boundary). Under various conditions, the problem of minimizing these upper bounds with respect to the parameters of the linear classifier leads to a modified formulation of the *hard-margin support vector machine* (SVM) [10, 11]. Given a generalized linear kernel of the form, $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{R} \mathbf{x}_2$, where $\mathbf{R}$ is a positive semidefinite parameter matrix, this modified SVM formulation implicitly prescribes the parameterization, $\mathbf{R} = \mathbf{W}^{-1}$, where $\mathbf{W}$ is the expected within-class covariance matrix over all classes. We can represent $\mathbf{W}$ mathematically as

$$\mathbf{W} \triangleq \sum_{i=1}^{M} p(i) \cdot \mathbf{C}_i,$$
$$\mathbf{C}_i \triangleq \mathbb{E}\,(\mathbf{x}_i - \bar{\mathbf{x}}_i)(\mathbf{x}_i - \bar{\mathbf{x}}_i)^T \quad \forall i.$$

Here, $\mathbf{x}_i$ represents a random draw from class i, M represents the total number of classes, and $\bar{\mathbf{x}}_i$ represents the expected value of $\mathbf{x}_i$. We use $\mathbf{C}_i$ and $p(i)$ to represent the covariance matrix and the prior probability of class i. (Note that in this paper, the term, "class" is synonymous with "speaker.") Given $\mathbf{W}$, where $\mathbf{W}$ is full-rank, we can implement a generalized linear kernel with $\mathbf{R} = \mathbf{W}^{-1}$ by using the following feature transformation, Φ:

$$\Phi(\mathbf{x}) \triangleq \mathbf{A}^T \mathbf{x}. \tag{1}$$

Here, $\mathbf{A}$ is defined as the Cholesky factorization of $\mathbf{W}^{-1}$:

$$\mathbf{A}\mathbf{A}^T \triangleq \mathbf{W}^{-1}.$$

In practice, empirical estimates of $\mathbf{W}$ are typically quite noisy; thus, a certain amount of smoothing is usually required to make the WCCN approach work. In this paper, we use the following smoothing model:

$$\hat{\mathbf{W}}_s \triangleq (1-\alpha) \cdot \hat{\mathbf{W}} + \alpha \cdot \mathbf{I}, \qquad \alpha \in [0, 1]. \tag{2}$$

Here, $\hat{\mathbf{W}}_s$ represents a smoothed version of the empirical expected within-class covariance matrix, $\hat{\mathbf{W}}$, and $\mathbf{I}$ represents an $N \times N$ identity matrix where N is the dimensionality of the feature space. The α parameter represents a tunable smoothing weight whose value is between 0 and 1. It's straightforward to show that in the above model, the eigenvectors of $\hat{\mathbf{W}}_s$ are constant with respect to α. Thus, we can compute the WCCN feature transformation, Φ, in (1) for any value of α without having to recompute the eigenvectors of $\hat{\mathbf{W}}_s$.

2.1. WCCN for Large Feature Sets

In this paper, we examine the problem of how to apply WCCN to large feature sets, where inverting or simply estimating $\hat{\mathbf{W}}$ is impractical for computational reasons. For large feature sets, we can use kernel principal component analysis (KPCA) to first reduce the dimensionality of the feature space to a more manageable size before performing WCCN. One potential problem with this approach, however, is that by filtering out various orthogonal vectors or "directions" in feature space (i.e. by performing feature reduction), we lose a significant amount of the information contained in the original feature set. To avoid this problem, we use the PCA decomposition described in [9], where the feature space is divided into two sets: a set that represents the top N features obtained from performing PCA, where N is the number of training vectors (i.e. the *PCA set*) and a *PCA-complement* set, which represents all of the information contained in the original features but not in the PCA set. Since all of the covariance information in the training data is confined to the PCA set (the PCA-complement is $\mathbf{0}$ for all feature vectors in the training data but generally non-zero for feature vectors outside of the training data), we can perform WCCN on the PCA set, which has reduced dimensionality, and then concatenate the resulting feature set with the PCA-complement. This procedure is described in the following sections.

3. Kernel PCA and the PCA-Complement

This section provides an overview of kernel PCA and also describes the PCA-complement approach used in [9]. We begin by defining $\mathbf{X}$ to be a column matrix containing scaled, mean-centered versions of the feature vectors in the training set:

$$\mathbf{X} \triangleq \sqrt{\frac{1}{N}} \cdot [(\mathbf{x}_1 - \bar{\mathbf{x}}), \ldots, (\mathbf{x}_N - \bar{\mathbf{x}})].$$

Here $\mathbf{x}_i$ represents the ith training vector, and $\bar{\mathbf{x}}$ represents the average over all N training vectors. Given the above definition, we can represent $\hat{\mathbf{C}}$ (i.e. the empirical covariance matrix of the data) as follows:

$$\hat{\mathbf{C}} = \mathbf{X}\mathbf{X}^T,$$
$$\triangleq \mathbf{U}\Sigma^2\mathbf{U}^T. \tag{3}$$

In the second line of the above equation, we define $\mathbf{U}\Sigma^2\mathbf{U}^T$ to be the eigendecomposition of $\hat{\mathbf{C}}$. We can represent the corresponding eigendecomposition for $\mathbf{X}^T\mathbf{X}$ as follows:

$$\mathbf{X}^T\mathbf{X} \triangleq \mathbf{V}\Sigma^2\mathbf{V}^T. \tag{4}$$

Here, we define $\mathbf{V}$ to be a column matrix containing the eigenvectors of $\mathbf{X}^T\mathbf{X}$ and Σ^2 to be a diagonal matrix containing the corresponding eigenvalues. If $\mathbf{X}^T\mathbf{X}$ is full-rank, then we can combine (3) with (4) to arrive at the following expression for $\mathbf{U}$, the eigenvector matrix of $\hat{\mathbf{C}}$:

$$\mathbf{U} = \mathbf{X}\mathbf{V}\Sigma^{-1}. \tag{5}$$

The columns of $\mathbf{U}$ represent the set of all eigenvectors of $\hat{\mathbf{C}}$ whose corresponding eigenvalue is non-zero. Thus, we can perform PCA by projecting the input feature vectors onto the column vectors of $\mathbf{U}$. This leads to the following feature transformation, Φ_{PCA}:

$$\Phi_{PCA}(\mathbf{x}) \triangleq \mathbf{U}^T\mathbf{x},$$
$$= \Sigma^{-1}\mathbf{V}^T\mathbf{X}^T\mathbf{x}. \tag{6}$$

This transformation reduces the dimensionality of the underlying feature space down to N features, where N is the size of the training set. Since the input feature vectors appear in the form of inner products, which can be replaced with kernel functions, this feature transformation is referred to as *kernel PCA* [12].

We use $\Phi_{\overline{PCA}}$ to represent the feature transformation for the PCA-complement, which is defined as follows:

$$\Phi_{\overline{PCA}}(\mathbf{x}) \triangleq (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\mathbf{x}. \tag{7}$$

The PCA-complement represents the portion of the original feature space that is orthogonal to the training set. Thus, $\Phi_{\overline{PCA}}(\mathbf{x}) = \mathbf{0}$ (i.e. a null vector) for all $\mathbf{x}$ in the training set.

4. Experimental Procedure

The experiments in this paper compare two different feature normalizations: WCCN and standard *covariance normalization* (CN), where $\mathbf{R} = \hat{\mathbf{C}}_s^{-1}$. (Here, $\hat{\mathbf{C}}_s$ represents a smoothed version of $\hat{\mathbf{C}}$, the empirical covariance matrix of the training data.) Since $\Phi(\mathbf{x})_{\overline{PCA}} = \mathbf{0}$ for all $\mathbf{x}$ in the training set, we have no way of coming up with a meaningful estimate of the covariance matrix for the PCA-complement (any empirical covariance estimate will simply be $\mathbf{0}$). Thus, WCCN and standard CN are only applied to the PCA feature set. The normalized PCA features are then concatenated with a weighted version of the PCA-complement to form the final feature representation.

Our experimental procedure is summarized below:

1. Perform per-feature within-class *variance* normalization on all of the input features (i.e. scale all features to have an average within-class variance of one on the training data). The resulting features provide us with a first-cut approximation of what we would obtain by performing full WCCN on the original feature set. This is simply a preprocessing step for performing KPCA, which is not invariant to scaling operations on the input features. Note that the smoothing model of (2) is also not invariant to scaling operations.
2. Compute $\Phi_{PCA}(\mathbf{x})$ for every feature vector $\mathbf{x}$ in the training and test sets. This gives us the PCA feature set.
3. Compute $\Phi_{\overline{PCA}}(\mathbf{x})$ for every feature vector $\mathbf{x}$ in the training and test sets. This gives us the PCA-complement feature set.
4. Perform either within-class covariance normalization (WCCN) or standard covariance normalization (CN) on the PCA feature set. Both normalizations can be represented in the form of a matrix multiplication. We use the smoothing model shown in equation (2) for both WCCN and standard CN. The smoothing parameter α is tuned on a set of held-out cross-validation data.
5. Concatenate a scaled version of the normalized PCA feature set with a scaled version of the PCA-complement feature set to arrive at our final feature representation, Φ:

$$\Phi(\mathbf{x}) \triangleq \left[\begin{array}{c} (1-\sigma) \cdot \mathbf{A}^T \Phi_{PCA}(\mathbf{x}) \\ \sigma \cdot \Phi_{\overline{PCA}}(\mathbf{x}) \end{array} \right], \quad \sigma \in [0,1]. \quad (8)$$

   Here, $\mathbf{A}^T$ represents the transformation matrix derived in step 4 to perform either WCCN or standard CN on the PCA feature set. Thus, $\mathbf{A}^T\Phi_{PCA}(\mathbf{x})$ represents the normalized PCA component of feature vector $\mathbf{x}$. We use the parameter σ to control the relative weight applied to the two feature sets (i.e. the PCA set and the PCA-complement set). This parameter is tuned on a held-out cross-validation set.
6. Use the final feature representation to train and test SVM-based speaker models.

Given a standard linear kernel, $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T\mathbf{x}_2$, it's fairly straightforward to show that when $\sigma = 0.5$ and $\mathbf{A} = \mathbf{I}$ (i.e. $\mathbf{A}$ is the idenitity matrix), then the following equality holds for any pair of input feature vectors, $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$k(\mathbf{x}_1, \mathbf{x}_2) = 4 \cdot k(\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2)). \quad (9)$$

The equality in (9) follows directly from the definitions for Φ, Φ_{PCA}, and $\Phi_{\overline{PCA}}$ in equations (8), (6), and (7). Equation (9) shows that when $\sigma = 0.5$ and $\mathbf{A} = \mathbf{I}$, then applying the feature transformation, Φ, to the input feature vectors does not affect the kernel function k beyond a scaling factor. Thus, by concatenating the PCA set with the PCA-complement set, we preserve all of the information contained in the original feature set, at least for the purpose of computing linear kernels.

5. Experiments and Results

In this section, we describe the tasks, datasets, and features used in our experiments. The results of these experiments are discussed in section 5.4.

5.1. MLLR-SVM System

We used an MLLR-SVM system similar to the one described in [4] to compute feature vectors for our experiments. The MLLR-SVM system uses speaker adaptation transforms from SRI's DECIPHER speech recognition system as features for speaker recognition. A total of 8 affine transforms are used to map the Gaussian mean vectors from speaker-independent to speaker-dependent speech models. The transforms are estimated using maximum-likelihood linear regression (MLLR), and can be viewed as a text-independent encapsulation of the speaker's acoustic properties. For every conversation side, we compute a total of 24960 transform coefficients, which are used as features. Note that this system uses twice as many features as the original MLLR-SVM system described in [3, 1]. The input feature vectors are identical to those used in [4]. However, besides applying the feature transformation Φ to the input feature vectors, our system differs from the MLLR-SVM system used in [4] in the following ways: 1) our system does not apply rank normalization [3] to the input feature vectors and 2) our system does not apply TNORM [13] to the output SVM scores. We have yet to experiment with applying these normalizations to a system that uses WCCN.

5.2. Task and Data

Experiments were performed on the 1-conversation training condition of two NIST-defined tasks: SRE-2004 and a subset of SRE-2003. Note that these tasks and datasets are the same as those described in previous reports (see [4, 1]). The SRE-2003 subset was divided into two splits of disjoint speaker sets, both comprised of ∼3600 conversation sides and ∼300 speakers. Each split comprises ∼580 speaker models and ∼9800 speaker trials. These splits were alternately used for training (i.e. computing covariance estimates and feature transformations) and for testing. We used SRE-2004 to tune α and σ for testing on SRE-2003, and vice-versa. To simplify the tuning process, α was optimized for the case where $\sigma = 0$. The resulting α parameter was then held fixed while tuning σ. Further details on the tasks and datasets can be found in [4].

5.3. SVM Training

We used SVM^{light} [14] to train SVM-based speaker models for each task. Each speaker model was trained with a linear kernel using the default value of the SVM hyperparameter C. A held-out dataset composed of 425 conversation sides taken from the Switchboard-2 corpus and 1128 conversation sides taken from the Fisher corpus was used as negative examples for the SVM training.

5.4. Results

Table 1 shows results on the MLLR-SVM system for various feature representations. Here, the labels "WCCN" and "CN" denote within-class covariance normalization and standard covariance normalization, where α is tuned on the cross-validation set. The σ parameter is optimized on the cross-validation set for systems that are labeled "$\overline{\text{PCA}}$." For systems that are *not* labeled "$\overline{\text{PCA}}$," σ is set equal to zero (i.e. the PCA-complement is omitted from the final feature representation). The "baseline" label represents the MLLR-SVM system without any feature normalization.

As shown in table 1, the WCCN approach provides improvements that are quite substantial, at least in most cases, over standard CN (see the "improvement over PCA+CN+$\overline{\text{PCA}}$" results). It's worth noting that the improvements obtained over the baseline are significantly larger on SRE-2003 than on SRE-2004. However, this is to be expected, since the feature transformations and normalizations used in these experiments were trained only on held-out SRE-2003 data, which represents a different set of channel and recording conditions than SRE-2004.

We note that the "PCA," "PCA+CN," and "PCA+WCCN" results are all obtained from PCA feature sets whose dimensionality is reduced to ∼3600 (i.e. the number of training examples in each split of the SRE-2003 subset). In spite of this reduced dimensionality, the "PCA+WCCN" system significantly outperforms the "baseline" system, where each feature vector is composed of 24960 features.

Table 1 also shows that adding the PCA-complement to the PCA feature set leads to significant relative reductions in error rate (see the "improvement over PCA+WCCN" results). To the best of our knowledge, the results for the "PCA+WCCN+$\overline{\text{PCA}}$" system are the best recorded so far in the literature for an MLLR-SVM system. Even without using rank normalization or TNORM—two techniques used in [4] which should presumably lead to reductions in error rate (we have not yet integrated these normalizations into our system)—our system outperforms the MLLR-SVM system in [4] by at least 15% on the SRE-2003 subset and by a smaller, but still significant margin on SRE-2004. These experiments point to the utility of using WCCN in conjunction with the PCA-complement when training SVM-based speaker models.

6. Conclusions

We describe a practical procedure for applying within-class covariance normalization (WCCN) to an MLLR-SVM speaker recognition system where the feature vectors reside in a high-dimensional space. When applied to a state-of-the-art MLLR-SVM speaker recognition system, this approach achieves improvements of up to 22% in EER and 28% in minimum decision cost function (DCF) over our previous baseline. We also achieve substantial improvements over an MLLR-SVM system that performs WCCN on the PCA set but discards the PCA-complement. These results point to the utility of using WCCN in conjunction with the PCA-complement when training SVM-based speaker models.

7. Acknowledgements

The material in this paper is based upon work supported by the NSF under grant No. 0329258. Additional funding was provided by a DoD KDD award via NSF IRI-9619921. The views herein are those of the authors and do not reflect the views of the funding agencies.

	SRE-03 subset		SRE-04	
Φ	EER%	DCF	EER%	DCF
baseline	2.91	0.117	5.97	0.282
PCA	3.89	0.158	7.35	0.318
PCA+CN	2.92	0.123	6.43	0.289
PCA+WCCN	2.30	0.108	5.52	0.260
PCA+$\overline{\text{PCA}}$	2.91	0.117	5.97	0.282
PCA+CN+$\overline{\text{PCA}}$	2.33	0.092	5.87	0.266
PCA+WCCN +$\overline{\text{PCA}}$	2.08	0.091	5.27	0.247
improvement over baseline	$\mathbf{28.5\%}$	$\mathbf{22.2\%}$	$\mathbf{11.7\%}$	$\mathbf{12.4\%}$
improvement over PCA+WCCN	$\mathbf{9.6\%}$	$\mathbf{15.7\%}$	$\mathbf{4.5\%}$	$\mathbf{5.0\%}$
improvement over PCA+CN+$\overline{\text{PCA}}$	$\mathbf{10.7\%}$	$\mathbf{1.1\%}$	$\mathbf{10.2\%}$	$\mathbf{7.1\%}$

Table 1: EERs and minimum DCFs for various feature transformations/normalizations on the MLLR-SVM system. Here, "baseline" represents the raw MLLR-SVM system without any feature normalization. The labels "WCCN" and "CN" denote within-class covariance normalization and standard covariance normalization, and "$\overline{\text{PCA}}$" denotes a system that uses the PCA-complement with σ optimized on the cross-validation set. The "improvement" entries represent the relative improvement of PCA+WCCN+$\overline{\text{PCA}}$ over the given system.

8. References

[1] A. Hatch and A. Stolcke, "Generalized linear kernels for one-versus-all classification: application to speaker recognition," in *to appear in proc. of ICASSP*, Toulouse, France, 2006.

[2] A. Hatch and A. Stolcke, "Adaptive linear kernels for binary classification of multicluster data," in *Technical Report*, 2006, http://www.icsi.berkeley.edu/∼ahatch/alk.pdf.

[3] A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman, "MLLR transforms as features in speaker recognition," in *Proceedings of Interspeech*, Lisbon, Portugal, 2005.

[4] A. Stolcke, L. Ferrer, and S. Kajarekar, "Improvements in MLLR-Transform-based Speaker Recognition," in *Proc. IEEE Odyssey 2006 Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, June 2006, http://www.speech.sri.com/papers/papers/odyssey2006-mllr.ps.gz.

[5] W. M. Campbell, "A Sequence Kernel and its Application to Speaker Recognition," in *Neural Information Processing Systems 14*, 2001.

[6] W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek, "Phonetic speaker recognition with support vector machines," in *Neural Information Processing Systems 16*, 2003.

[7] A. Solomonoff, C. Quillen, and W. Campbell, "Channel Compensation For SVM Speaker Recognition," in *Proc. of Odyssey: The Speaker and Language Recognition Workshop*, Toledo, Spain, 2004.

[8] A. Solomonoff, W. Campbell, and I. Boardman, "Advances In Channel Compensation For SVM Speaker Recognition," in *Proc. of ICASSP*, Philadelphia, PA, 2005.

[9] S. Kajarekar, "Four weightings and a fusion: a cepstral-svm system for speaker recognition," in *Proc. of ASRU*, San Juan, Puerto Rico, 2005.

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

[11] N. Cristianini and J. Shawe-Taylor, *Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, 2000.

[12] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, Cambridge, 2004.

[13] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, "Score normalization for text-independent speaker verification systems," in *Digital Signal Processing*, San Juan, Puerto Rico, Jan. 2000, vol. 10.

[14] T. Joachims, "Making large-scale SVM learning practical," in *Advances in kernel methods — support vector learning*, B. Schoelkopf, C. Burges, and A. Smola, Eds. MIT-press, 1999.