Journal of Machine Learning Research 10 (2009) 2079-2114 Submitted 6/08; Revised 4/09; Published 9/09

An Anticorrelation Kernel for Subsystem Training in Multiple Classifier Systems

Luciana Ferrer[*] LFERRER@SPEECH.SRI.COM
Speech Technology and Research Laboratory
SRI International
333 Ravenswood Ave
Menlo Park, California, 94025, USA

Kemal Sönmez SONMEZK@OHSU.EDU
Division of Biomedical Computer Science, School of Medicine
Oregon Health and Science University
3181 S.W. Sam Jackson Park Rd.
Portland, Oregon, 97239, USA

Elizabeth Shriberg EES@SPEECH.SRI.COM
Speech Technology and Research Laboratory
SRI International
333 Ravenswood Ave
Menlo Park, California, 94025, USA

Editor: Leon Bottou

Abstract

We present a method for training support vector machine (SVM)-based classification systems for combination with other classification systems designed for the same task. Ideally, a new system should be designed such that, when combined with existing systems, the resulting performance is optimized. We present a simple model for this problem and use the understanding gained from this analysis to propose a method to achieve better combination performance when training SVM systems. We include a regularization term in the SVM objective function that aims to reduce the average class-conditional covariance between the resulting scores and the scores produced by the existing systems, introducing a trade-off between such covariance and the system's individual performance. That is, the new system "takes one for the team", falling somewhat short of its best possible performance in order to increase the diversity of the ensemble. We report results on the NIST 2005 and 2006 speaker recognition evaluations (SREs) for a variety of subsystems. We show a gain of 19% on the equal error rate (EER) of a combination of four systems when applying the proposed method with respect to the performance obtained when the four systems are trained independently of each other.

Keywords: system combination, ensemble diversity, multiple classifier systems, support vector machines, speaker recognition, kernel methods

∗. This author performed part of the work presented in this paper while at the Information Systems Laboratory, Department of Electrical Engineering, Stanford University.

1. Introduction

The work presented in this paper is motivated by our work on the task of speaker verification. In the last decade, many successful speaker verification systems have relied on the combination of various component systems to achieve superior performance. In many cases, as in Ferrer et al. (2006) and Brummer et al. (2007), the combination leads to significant improvements. However, there are cases in which combining several comparably good systems does not result in improvements over the single best system (Reynolds et al., 2005). Most of these systems perform the combination of information sources at the score level[1] (Reynolds et al., 2003; Ferrer et al., 2006; Brummer et al., 2007; Huenupán et al., 2007; Dehak et al., 2007): systems that model each type of feature using a certain model are independently developed and their scores are combined to produce the final score and the decision. When training each individual system, all other systems available for combination are usually ignored while, in fact, the ultimate goal of the systems is to perform well in combination with all the other systems and not necessarily individually.

It is easy to see that system combination at the score level is not guaranteed to give strictly better performance than those of the individual systems being combined. In the extreme case, if all classifiers were generating exactly the same output for each sample, the combined classifier would not have better performance than the individual ones, independently of the combination procedure used. Intuitively, what we wish is to have enough diversity across systems such that classifiers contribute complementary information leading to a better final decision when systems are combined. System diversity has been the subject of a large amount of research in recent years, with two main goals: defining a measure of diversity that can predict the performance of the combination, and designing procedures for achieving diversity in an ensemble of systems. With the goal of motivating and placing our work in perspective, in the next section we present a discussion focused on existing techniques for measuring and designing for system diversity.

The contributions of this paper are: (1) the development of a simple model for the combination problem for a binary classification task under the assumption that the distribution of scores for each of the classes is Gaussian, and (2) a procedure for improving diversity in an ensemble including SVM classifiers. We find an upper bound on the EER of an ensemble combination and show that, for a two-system combination, this upper bound is a function of the performance of the individual systems and the correlation coefficient obtained from the average class-conditional covariance matrix of the scores from the two systems. Based on this result, we propose the inclusion of a regularization term in the SVM objective function when training a new SVM system for combination with a set of preexisting systems, which introduces a trade-off between the performance of the resulting model and its average class-conditional covariance with the preexisting systems. Ferrer et al. (2008b) presented empirical results using the proposed method. Here, we extend this previous work by developing a framework under which to understand the method, considering the cases of multiple preexisting systems and nonlinear kernels, and including new results on simulated and on the NIST 2005 and 2006 speaker-verification evaluation data. Results show that a gain of 19% on EER can be achieved when using the proposed method with respect to the results obtained when systems are trained without knowledge of the others.

The paper is organized as follows. Section 2 gives a review of the related research. Section 3 describes a simple model for the system combination problem under consideration. Using the

1. The term *score* is commonly used in the speaker verification community to refer to the numerical output of a system, which may or may not be a probability measure.

conclusions obtained from this development, Section 4 proposes a method for achieving improved combination performance. In Sections 5 and 6 we present results on simulated data and speaker verification data, respectively. Section 7 presents our conclusions.

2. Multiple Classifier Systems

In this section we review the literature related to our work, motivating and putting in perspective the research presented in the rest of the paper.

2.1 Measuring Diversity

For regression problems, the measures of ensemble diversity are well developed. Krogh and Vedelsby (1995) showed that the quadratic error for a certain input value of a convex combination of estimators trained on a single data set is guaranteed to be less than or equal to the weighted average quadratic error of the component estimators. The difference between the two is given by an ambiguity term that measures the variability among ensemble members for the particular input. On a related development, Ueda and Nakano (1996) give a decomposition of the mean square error of an ensemble classifier into three terms: average bias, variance and covariance of the ensemble members. One would then wish to reduce the covariance term without affecting the bias and variance terms by reducing the correlation between the members. This term can, in fact, be negative.

Classification problems where ensemble members output an estimate of the posterior probabilities of the classes (as opposed to just the estimated label) can also be considered regression problems. Tumer and Ghosh (1996) and, later, Fumera and Roli (2005) present a framework for studying this case. They consider the probability of error as the measure of performance. If the actual posterior probabilities of the classes given the features were known, the probability of error could be minimized by choosing, for each value of the input features, the class that has the highest posterior. This rule determines a boundary between the regions where each class is predicted and results in a certain probability of error that is called the *Bayes error*. In practice, the posterior probabilities are not known. Tumer and Ghosh assume an additive error model where the individual classifiers output, for each class w_k, a function of the features x such that $f_k(x) = P(w_k|x) + \varepsilon_k(x)$. The predictions are now made based on the output of the classifier instead of the actual posterior probabilities. Tumer and Ghosh consider the case in which the effect of using the predictions is a shift in the boundary by a certain amount b. This results in the probability of error being larger than the Bayes error by a certain amount they call the *added error*. An expression for the expected value of the added error is computed assuming that the ε_k are random variables with a certain mean and variance independent of the value of x, that they are uncorrelated across classes, that the b is small such that first-order approximations can be used and that the posterior probabilities are monotonic around the decision boundaries. Using the expression derived for the added error of a single classifier, they derive the corresponding expression for a classifier formed by the average estimates obtained from a set of N individual classifiers. Their final expression depends monotonically on the pairwise correlations between the error functions of the individual classifiers, supporting the intuition that good combination performance can be achieved if the classifiers being combined have complementary information such that when one classifier makes a mistake, some other classifier has the right answer. Fumera and Roli generalize Tumer and Ghosh's work by allowing the combination to be a weighted average instead of simple average and arrive at an optimization problem for the

combination weights. This problem is solved explicitly for the case of unbiased and uncorrelated estimation errors.

Two assumptions made by Tumer and Ghosh, as well as Fumera and Roli, are not well suited for the problem of interest in this paper but can be easily replaced for other assumptions without much consequence on the theory: the use of the classification error as measure of performance and the uncorrelatedness of the estimation errors across classes. Classification error is usually considered inappropriate for problems in which the prior probabilities of the classes are very different (as is the case in most speaker verification applications). For these cases, cost functions are usually defined where each type of error is assigned a different cost and the expected value of the cost function is used as performance measure. Derivations in Tumer and Ghosh (1996) can be easily modified to allow for cost functions. The Bayes classifier would now choose the class for which the loss is minimized instead of the posterior probability maximized. As for the estimation errors, the assumption of zero correlation can be easily relaxed when only two classes are considered. In fact, in our case we can assume that only one of the posteriors is estimated and the other one is calculated so that the sum is equal to one. In that case, the estimation errors would also sum to one. Hence, only one estimation error stays in the derivations and the zero correlation assumption is not needed.

Even though the results described above could be considered enough motivation for our development of the anticorrelation method in Section 4, in the next section we propose a different development that we believe presents an interesting alternative for the above framework. First, we do not use the additive error model with fixed bias and variance for the posterior probability estimates. In binary problems like speaker verification one can train classifiers that output a score instead of a probability. This score can be, for example, the output of a support vector machine or the logarithm of the ratio between the class likelihoods. These scores are assumed to be monotonically related to the posterior probability of one of the two classes and, hence, could be easily converted into this posterior. Nevertheless, in our speaker verification experiments we have found that combining the scores directly leads to better results than combining the estimated posteriors when using a linear combiner. In a combination experiment of all pairs of systems from a set of 13 systems, an average relative gain of 7.6% was found when combining scores versus posterior probabilities (obtained by learning a logit mapping of the scores) using linear logistic regression. The best performance across all combinations including any number of systems is obtained when combining the scores of six systems and this combination is 12% better than the best combination of posterior probabilities. Considering these results, we eliminate the assumption that the output of the classifiers is a posterior probability, allowing it to be a general continuous value (score). We then replace the additive error assumption with an assumption that the distribution of the scores for each class is Gaussian, which, as we will see, is a good approximation for most speaker verification scores.[2] Finally, we focus on a different measure of performance, the equal error rate, which is widely used on binary classification problems for its simplicity and for being independent of the class priors. This measure of performance depends on the overall shape of the posterior probabilities and not just on the difference between them, as is the case for the probability of error or the expected cost. Hence, no simple extension of the Tumer and Ghosh derivations could be made for this measure even if we

2. We believe this is the reason for the scores combining better with a linear combiner than the mapped posterior probabilities since when class distributions are Gaussian, the Bayes classifier is linear. Hence, the linear combiner is a good choice when combining speaker verification scores. The Gaussian approximation, on the other hand, is not good after the scores have been mapped to posterior probabilities making a linear combiner on the posterior probabilities suboptimal.

were willing to consider the system output to be a probability. Under these assumptions we arrive at an explicit expression for an upper bound on the optimal EER of the combination that is a function of the EER of the ensemble members and the pairwise class-conditional correlations.

2.2 Creating Diversity

The work described above, including the development we will show in Section 3, defines ways of quantifying the diversity in the ensemble and how this diversity affects the performance of the combined classifier under different sets of assumptions, but does not describe ways of achieving this diversity. Brown et al. (2005a) present a taxonomy of the methods for creating diversity found in the literature as of 2005. They divide the methods into implicit and explicit, where implicit methods are those that try to create diversity using randomization methods, while explicit methods do the same by directly taking into account some measure of the diversity being achieved. They further divide the methods into three groups based on how they affect the learners to create diversity: (1) by modifying the starting point in the hypothesis space, (2) by changing the set of accessible hypotheses or, (3) by defining how the hypothesis space is traversed. Modifying the starting point in the hypothesis space is applicable only for learners for whom the final hypothesis reached depends on some random initialization component, as is the case for neural networks. SVMs, on the other hand, do not fall into this category. Bagging (Breiman, 1996) and boosting (Freund and Schapire, 1997) are diversity creation methods of type 2, since each member of the ensemble is obtained by changing the training data, resulting in a different set of accessible hypotheses. In the case of bagging, the method is implicit since the training data for each ensemble is chosen randomly from the original set, while in the case of boosting, the training samples are weighted when training a new member of the ensemble in a way that ensures diversity, making it an explicit method of diversity creation. Allowing each member of the ensemble to use only a subset of features falls into the type 2 methods (Oza and Tumer, 2001). Finally, type 3 methods are those that directly aim at improving diversity by including a term measuring diversity in the objective function of the learner. The negative correlation (NC) learning algorithm is a notable example in this category (Liu, 1999; Rosen, 1996).

The goal of NC Learning is to minimize the squared error of an ensemble output computed as the average of the individual outputs in a regression context. This is done by adding a penalty term in the objective function of each individual neural network forming the ensemble. This penalty was shown (Brown et al., 2005b) to directly control the covariance term in the bias-variance-covariance trade-off. Zanda et al. (2007) extend the NC learning framework to classification problems by reinterpreting the Tumer and Ghosh model in a regression context. Our proposed anticorrelation method follows the spirit of the NC learning technique of explicitly creating diversity through the modification of the learner's objective function when the learners are SVMs instead of neural networks.

Much of the work on diversity creation has focused on the generation of large ensembles, where the number of classifiers is part of the design choices. The size of the ensemble is in fact another variable to be optimized. In this paper, we constrain the ensemble to contain a relatively small number of systems. These systems are different in nature, using different sets of features and different modeling methods and can be quite complex. Each of them might have been developed over several months or years of research. Hence, we do not take the size of the ensemble as a variable. The individual systems are fixed before hand, and all we are allowed to do is (perhaps only slightly) modify the training procedure of one or more of them in order to increase the diversity of the ensemble. The

speaker verification system described in this paper can, in fact, be considered as a cascade of two diversity creation methods. Given an enormous set of highly heterogeneous features coming from a few different sources in the speech signal (for example, prosodic or spectral information), we train separate classifiers for each type of feature, perhaps also varying the type of classifier used. This can be seen as a type 2 diversity creation method. The second stage of diversity creation focuses on making each new classifier added to the ensemble as new as possible. To achieve this we use a type 3 diversity creation method, where we add a penalty term in the SVM objective function, which explicitly aims at reducing the correlation between the new classifier and the preexisting ensemble. The proposed method is shown to be equivalent to defining a new kernel that we call the *anticorrelation kernel*.

Kocsor et al. (2004) introduced a method called *margin maximizing discriminant analysis* (MMDA) to obtain sucessive, mutually orthogonal, SVMs for a certain feature vector. In this method, presented as a nonparametric extension of linear discriminant analysis, the SVM optimization problem is modified by adding an orthogonality constraint with respect to the weight vectors from previous SVMs. The MMDA method can be seen as a particular case of the one proposed in this paper when the penalty coefficient is set to infinity. Furthermore, the constraints used here are more general, reducing to the ones used in MMDA when all systems in the ensemble are SVMs, use the same input features, and these features have identity within-class covariance matrices. Kernel-based methods for ensemble systems have also been used, for example, in Pavlidis et al. (2002) and Lanckriet et al. (2004). Pavlidis et al. compare three methods for combination of heterogeneous information for gene function detection: early, intermediate and late integration. Early integration (what we here call *feature-level combination*) consists of concatenating the features from the two different information sources into a single feature vector. Intermediate integration performs the combination at the kernel level, and late integration performs the combination at the last stage. This is what we are calling *score-level combination*. No explicit attempt at increasing diversity is made in the paper. Lanckriet et al. (2004) propose a method for combining kernel classifiers by learning a new kernel matrix that is a linear combination of the kernel matrices of the classifiers in the ensemble. This method requires prior knowledge of the test data, since instead of learning a function, they learn the labels on the set of unlabeled samples. This particular scenario is not applicable to speaker recognition where the speaker models are usually trained before any test sample is available.

2.3 System Combination

Once a diverse ensemble has been trained, the output of the individual classifiers must be combined into a single decision or score. Linear combiners are the most widely used methods for fusion of classifier outputs. In many cases the weights of the linear combination are determined based on the classification performance of the classifiers. For a survey of these methods, see Kuncheva (2004), Chapter 5. Obtaining the weights by training a "supra" classifier or combiner that takes the output of the individual classifiers as input is in many cases a better option. The main problem with this approach is that using the output of the individual classifiers on the data used for training them as training data for the combiner may result in suboptimal performance. The stacked generalization method (Kuncheva, 2004, Chapter 3) can be used in these cases to generate the training data for the combiner.

The focus in this paper is on diversity creation rather than on the methods used for combining the ensemble. Hence, we choose a simple but effective combination method: a linear supra-classifier

trained with data generated by the stacked generalization method using logistic regression. Linear logistic regression was shown in our previous work (Ferrer et al., 2008a) to perform comparably to or better than other linear and nonlinear classifiers for the combination of speaker verification scores, and it is one of the most commonly used methods for combining speaker verification systems (Brummer et al., 2007). Other common combination procedures used in speaker verification include neural networks (Reynolds et al., 2003; Ferrer et al., 2006), support vector machines (SVM) (Ferrer et al., 2006), and weighted summation using empirically determined weights (Dehak et al., 2007).

3. A Simple Model for System Combination

Consider a binary classification task with classes $y \in \{a,b\}$ for which N separate classifiers are available. In speaker verification, class *a* corresponds to *impostor* and *b* to *true-speaker*. Classifier i produces a score, f_i, which can be thresholded to obtain the final decision. That is, the estimated class $\hat{y}$ is given by

$$\hat{y} = \begin{cases} a & \text{if } f_i < t_i \\ b & \text{if } f_i \geq t_i, \end{cases} \tag{1}$$

where t_i is a tunable threshold.

Here, we consider a setup where the scores from the individual classifiers are combined into a single score $f_c = f_c(f_1, \ldots, f_N)$. This final score is the one that is later thresholded to obtain the estimated class for the sample. The goal is then to optimize the performance of the final combination, not that of the individual systems.

In this section we develop a simple model for this problem, which will lead us to an intuitive conclusion about what could be done to improve the final performance when training a new system for combination with others.

3.1 Mahalanobis Distance as a Surrogate for EER

Consider for now a single score f, corresponding to a random variable, F.[3] A usual way of measuring performance of a score when Equation (1) is used to estimate the class of the samples is equal error rate (EER), the false acceptance rate when the false rejection and false acceptance rates are equal. In speaker verification, a false acceptance (which we will call $e_{b|a}$) is an impostor trial accepted by the system as the target speaker, and a false rejection ($e_{a|b}$) is a true-speaker trial considered an impostor trial by the system.

If we make the assumption that the conditional distribution of the scores for each class is Gaussian, we can obtain an explicit bound for the EER as a function of the Mahalanobis distance between the class-conditional means. Figure 1 shows how to calculate the EER for this case. We assume that the class-conditional distribution of the scores is given by

$$F \mid Y = y \sim \mathcal{N}(\mu_y, \sigma_y^2), \text{ for } y = a,b,$$

where Y is the random variable corresponding to the trial's class. We further assume, without loss of generality, that $\mu_b \geq \mu_a$, so that (1) is the best way of assigning the labels for a given threshold.

3. Throughout the paper we will adopt the notation of using lower case letters for the samples of the random variable noted by the corresponding capital letter.

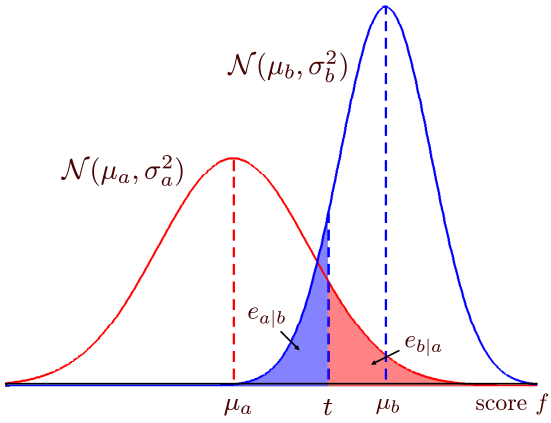

Figure 1: EER calculation when the class-conditional distributions are Gaussian. The EER is equal to $e_{a|b}$ when t is chosen such that $e_{a|b} = e_{b|a}$.

With these assumptions we can compute $e_{a|b}$ and $e_{b|a}$ for a certain value of the threshold t as

$$e_{a|b}(t) = \phi\left(\frac{t-\mu_b}{\sigma_b}\right), \tag{2}$$

$$e_{b|a}(t) = 1-\phi\left(\frac{t-\mu_a}{\sigma_a}\right), \tag{3}$$

where ϕ is the cumulative distribution function of the standard normal distribution $\mathcal{N}(0,1)$.

The EER is given by $e_{b|a}(t^*)$ when t^* is chosen such that $e_{b|a}(t^*) = e_{a|b}(t^*)$. In the appendix we prove the following upper bound on the EER:

$$\text{EER} = e_{b|a}(t^*) = e_{a|b}(t^*) \leq \phi(-\frac{1}{2}\frac{\delta\mu}{\sigma}), \tag{4}$$

where $\delta\mu = \mu_b - \mu_a$, and σ is any value that satisfies $\sigma \geq (\sigma_a + \sigma_b)/2$. Equality is achieved for $\sigma = (\sigma_a + \sigma_b)/2$, but this value of σ does not result in a nice expression later on, when we want to use it to optimize the combination performance. On the other hand,

$$\sigma = \sqrt{(\sigma_a^2 + \sigma_b^2)/2}$$

also satisfies the inequality and does result in a nice expression that we can later use. In the rest of the paper we will use

$$\mathcal{M}^2 = \frac{2\delta\mu^2}{\sigma_a^2 + \sigma_b^2} \tag{5}$$

as a surrogate for the EER of the system. $\mathcal{M}$ is the Mahalanobis distance between two Gaussian distributions with distance between the means $\delta\mu$ and variance $(\sigma_a^2 + \sigma_b^2)/2$. Using (4), we get that

$$\text{EER} \leq \phi(-\frac{\mathcal{M}}{2}).$$

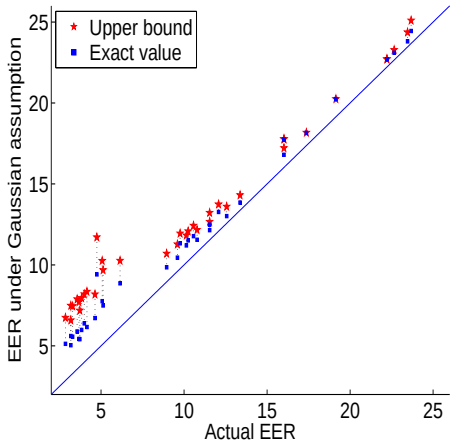

Figure 2: Actual EER for a set of 37 systems versus the upper bound and the exact value under Gaussian assumptions.

Our strategy in the remainder of the paper will be to reduce the value of the upper bound, with the hope that this will result in a reduction of the EER. Since ϕ is monotonically increasing in its argument, decreasing the value of $-\mathcal{M}$ (or increasing the value of $\mathcal{M}$) will decrease the value of the upper bound.

Figure 2 shows a scatter plot of the actual EER for a set of 37 different individual systems using different features or different modeling techniques (some of these systems are described by Ferrer et al. (2006)) versus the upper bound on the EER under Gaussian assumption given by $\phi(-\delta\mu/\sqrt{2(\sigma_a^2+\sigma_b^2)})$ and the exact value of the EER, also under Gaussian assumption, given by $\phi(-\delta\mu/(\sigma_a+\sigma_b))$. We can see that the upper bound is tight for high EER values. When the difference between the class-conditional covariance for both classes is large (which, in our experiments, is the case for the better systems), the upper bound becomes looser. We see that the exact estimation under the Gaussian assumption performs significantly better in this case. The remainder of the error is due to the inaccuracy of the Gaussian assumption. Figure 3 shows the actual distribution of two of the systems used in this paper compared to their Gaussian approximation. Figure 3.a corresponds to one of the low EER systems that drifts away from the diagonal in Figure 2, while 3.b corresponds to a system with EER around 12%. We can clearly see that the Gaussian approximation in this case is inaccurate, which in turn explains the inaccuracy of the upper bound and the exact formula developed in this section. Interestingly, even when the Gaussian approximation is inaccurate, (4) seems to still hold. Our procedure of trying to minimize this upper bound in the hope that the actual EER will be pushed down would then still be valid.

3.2 Maximum Mahalanobis Distance Combination

Returning to the problem of combining the output of several subsystems into a single score, we use the results from the previous section and maximize (5) to find the optimal parameters of the combiner. This will result in a simple formula that can predict the performance of the combination of

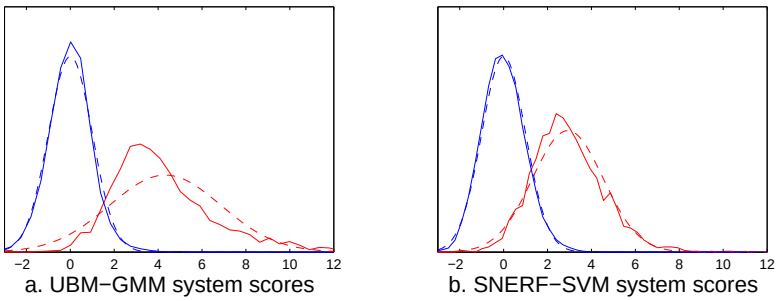

Figure 3: Distribution of scores for two individual systems compared to their Gaussian approximation.

several subsystems given their individual EERs (or $\mathcal{M}$s) and the average class-conditional covariance matrix for the vector of scores. To do this we further assume that the combination is performed with a linear function of the individual scores. This is not a very restrictive assumption for speaker verification since for this task we have repeatedly found that linear combination procedures perform as well (or better) than nonlinear ones (Ferrer et al., 2008a). The form of the combined scores as a function of N individual scores is then

$$f_c = \sum_{i=1}^{N} \alpha_i f_i = \alpha^t f,$$

where $\alpha = (\alpha_1 \dots \alpha_N)^t$ is the vector of weights and $f = (f_1 \dots f_N)^t$ the vector of individual scores. We assume that the class-conditional distribution of the f_i's is jointly Gaussian. Hence, each of the individual scores and the combined score satisfy the assumptions made in the previous section. The class-conditional distributions of F_c, the random variable corresponding to the combined score, are given by

$$F_c \mid Y = y \sim \mathcal{N}(\alpha^t \mu_y, \alpha^t \Sigma_y \alpha), \text{ for } y = a, b,$$

where $\mu_y = (\mu_{y1} \dots \mu_{yN})$ is the vector of means and $\Sigma_y = E[(F - \mu_y)(F - \mu_y)^t | Y = y]$ the covariance matrix for class y for random variable F corresponding to vector f. We can now compute $\mathcal{M}^2$ (Equation 5) for the combined score as a function of the parameters α:

$$\mathcal{M}^2 = \frac{2(\alpha^t \mu_b - \alpha^t \mu_a)^2}{\alpha^t \Sigma_a \alpha + \alpha^t \Sigma_b \alpha} = \frac{\alpha^t \Delta \Delta^t \alpha}{\alpha^t \Sigma \alpha}, \tag{6}$$

where $\Delta = \mu_b - \mu_a$ and

$$\Sigma = 1/2(\Sigma_a + \Sigma_b). \tag{7}$$

We wish to find the α that maximizes this expression. Define $\beta = \Sigma^{1/2}\alpha$, where $\Sigma^{1/2}$ is a symmetric matrix square root of Σ (which can be computed from the eigendecomposition of Σ, which exists and can be chosen to be symmetric since Σ is symmetric). Replacing this in (6), and using the Cauchy-Schwarz inequality,

$$\begin{aligned} \mathcal{M}^2 &= \frac{\beta^t \Sigma^{-1/2} \Delta \Delta^t \Sigma^{-1/2} \beta}{\beta^t \beta} = \frac{\|\Delta^t \Sigma^{-1/2} \beta\|^2}{\|\beta\|^2} \\ &\leq \|\Delta^t \Sigma^{-1/2}\|^2 = \Delta^t \Sigma^{-1} \Delta, \end{aligned}$$

with equality when $\beta \propto \Sigma^{-1/2}\Delta$. Hence, the optimal α is a vector in the direction of $\Sigma^{-1}\Delta$.[4]

Note that we have arrived at the definition of the Mahalanobis distance in multiple dimensions ($\Delta^t\Sigma^{-1}\Delta$), which is an intuitive result. We now know that the EER of the combination of the individual systems will be at most $\phi(-\sqrt{\Delta^t\Sigma^{-1}\Delta}/2)$. If we can devise a way of increasing $\Delta^t\Sigma^{-1}\Delta$, this upper bound will decrease.

Anderson and Bahadur (1962) consider the problem of finding all the admissible linear procedures for classifying into two multivariate normal distributions. They show that $\alpha = (t_a\Sigma_a + t_b\Sigma_b)^{-1}\Delta$ corresponds to an admissible procedure for any t_a and t_b (that is, no other linear procedure will have, simultaneously, strictly better $e_{b|a}$ and $e_{a|b}$) as long as $t_a\Sigma_a + t_b\Sigma_b$ is positive definite. Unfortunately, the values for t_a and t_b that correspond to the EER have to be numerically obtained. No explicit expression is available for the general case. Since our purpose here is to obtain a simple explicit expression for the EER (as a function of the EERs of the individual systems and some other parameters) that we can use to analyze the problem, we have settled for a bound on the EER instead of using the implicit expression developed by Anderson and Bahadur.

3.3 Analysis for Combination of Two Systems

We now focus on the case $N = 2$. In this case, we have that the optimal $\mathcal{M}^2$ ($\Delta^t\Sigma^{-1}\Delta$) is given by

$$\begin{aligned}\mathcal{M}^2 &= \frac{\delta_1^2\sigma_{22} + \delta_2^2\sigma_{11} - 2\delta_1\delta_2\sigma_{12}}{\sigma_{11}\sigma_{22} - \sigma_{12}^2} \\ &= \frac{\mathcal{M}_1^2 + \mathcal{M}_2^2 - 2\rho\mathcal{M}_1\mathcal{M}_2}{1-\rho^2},\end{aligned} \tag{8}$$

where δ_i is component i of vector Δ, σ_{ij} is component ij of matrix Σ, $\mathcal{M}_1 = \delta_1/\sqrt{\sigma_{11}}$ and $\mathcal{M}_2 = \delta_2/\sqrt{\sigma_{22}}$ are the Mahalanobis distances for the individual systems, and

$$\rho = \sigma_{12}/\sqrt{\sigma_{11}\sigma_{22}}. \tag{9}$$

Note that ρ is *not* the correlation between the two systems in the usual sense, since Σ is not the covariance matrix of F, but the average class-conditional covariance.

Let us call the upper bound on the EER, $\hat{e}$, that is, $\hat{e} = \phi(-\mathcal{M}/2)$. Figure 4 shows some curves of $\hat{e}_c$, the upper bound on the performance of the combination for two systems, as a function of ρ. To create these plots we take two actual systems (the MLLR-SVM and the SNERF-SVM, as described in Section 6.2) and compute the upper bound on the EERs based on their Gaussian approximation. We also compute matrix Σ and from there, ρ. This gives a single point in this graph (marked with a star). We can now vary ρ, keeping $\hat{e}_1$ and $\hat{e}_2$ (the upper bound on the performance of the individual systems) fixed, to obtain a curve. We can also obtain curves for different values of $\hat{e}_2$. These curves show us the relation between the performance of system 2, the value of ρ between the two systems, and the performance of their combination. We can see that for this particular set of systems, we could degrade the second system 100% (that is, from 15.6% to 31.3%) and still get a gain in the

4. Note that this is the same direction one would obtain with linear discriminant analysis (LDA). As mentioned in the introduction, for the experiments in this paper we will use linear logistic regression (LLR) to train the combination weights. The reason for using LLR instead of LDA, despite the result obtained in this section, is that the results in this section are obtained under the assumption of class-conditional Gaussianity. When Gaussianity is not closely satisfied, LLR is believed to be a safer choice being more robust to outliers (Hastie et al., 2001, Section 4.4). In fact, a set of initial experiments showed that LLR was significantly better than LDA on our speaker verification data.

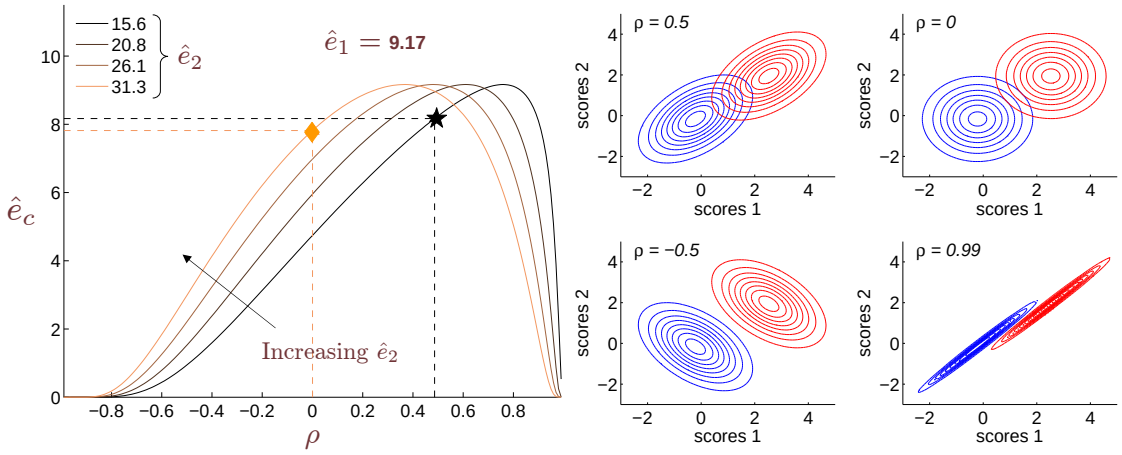

Figure 4: Left: Curves of $\hat{e}_c$ (the upper bound on the EER of the combination) for two systems as a function of ρ, for $\hat{e}_1$ fixed and various values of $\hat{e}_2$, where $\hat{e}_i$ denotes the upper bound on the EER of system i. Right: Simulated contour plots for the scores of two systems assuming class-conditional distributions are Gaussian with equal covariance matrix. Marginal distributions are kept fixed for all four figures, only ρ is changed. These plots correspond to four different points in the darker curve from the left plot.

performance of the combination (or at least in the upper bound) over what we get with the original pair of systems if, at the same time, we were able to decrease the ρ from 0.48 to 0 (this point is denoted by a diamond in the graph).

At first sight, these curves may seem to go against intuition, since when ρ increases, after reaching a peak (occurring at the minimum of $\mathcal{M}_1/\mathcal{M}_2$ and $\mathcal{M}_2/\mathcal{M}_1$), they go down again. That is, very high values of ρ result in extremely good combination performance. Similarly, when ρ turns negative, the combination performance improves, reaching zero EER for $\rho = -1$. All these cases can be easily understood using contour plots of the scores from two systems for varying values of ρ. The right plot of Figure 4 shows four different cases. Here we keep the marginal distributions of the systems fixed and vary only ρ, which implies that the performance of the two individual systems stays fixed for the four different plots. That is, the four plots correspond to four different points in a single curve like the ones in the left plot. Furthermore, we set the covariance matrix between the two systems to be the same for both classes. In this way, the upper bound (4) is exact and ρ is the within-class correlation (assuming both classes have the same prior). We can see that the separation between the two classes is highly dependent on the value of ρ. The first plot shows a typical case, where $\rho = 0.5$. The second plot shows the case of $\rho = 0$. We can already see that the overlap between the two classes has been significantly reduced, even though the marginal distributions have not changed. The third plot shows the case of negative ρ. This implies that both systems produce errors in a negatively correlated way, which makes the combination of those two systems extremely effective at reducing the error rate. The fourth plot illustrates the case in which ρ is larger than the value for which the $\hat{e}_c$ curve peaks, showing good separation between the two classes. Perfect classification with $\rho = 1$ is possible only when both classes have exactly the same

covariance matrix except possibly for a scalar factor (so that the contour plots become parallel lines) and the mean vectors for both systems are different. This case corresponds to a zero value in the denominator of (8) and a nonzero at the numerator.

In practice, if we take pairs of all 37 systems plotted in Figure 2 and compute the actual ρ and the ρ at the peak (that is, $\rho_{peak} = \min(\mathcal{M}_1/\mathcal{M}_2, \mathcal{M}_2/\mathcal{M}_1)$) we find that, on average, the actual ρ is 0.26 to the left of ρ_{peak}. Considering this empirical fact it seems unreasonable to try to increase ρ in order to improve the combination performance, since it would require a large increase in order to go past the peak into an area where the combination performance is better than the original. Furthermore, this would work only if the class-conditional covariance matrices were equal, which is usually not the case. Hence, in this paper, our strategy will be to try to decrease the value of ρ. Of course, we also require that the performance of each individual system stays reasonably close to its original performance, which we hope will result in a new point (in a plot like the one in the left of Figure 4) located toward the left and lower than the original point. In the next section we introduce our method for achieving this goal.

4. Anticorrelation Kernel

Suppose that two separate classifiers S and B are available, where S is required to be an SVM, but B can be any classifier that produces a score for each sample. We will consider B to be a black box from which we have only the scores that it produces. As we have been assuming, the final classification decision will be made based on a combination of the outputs generated by both classifiers. Our strategy will be to train system S using information about system B in order to improve the combination performance over the one obtained when system S is designed with no knowledge of system B.

4.1 Support Vector Machines

Consider a labeled training set with m samples, $T = \{(x_j, y_j) \in \mathcal{R}^d \times \{-1,+1\}; j = 1,...,m\}$, where x_j is the feature and y_j the class corresponding to sample j. The goal is to find a function $f(x) = w^t x + c$, such that $\text{sign}(f(x))$ is the predicted class for feature vector x. The standard (primal) SVM formulation for classification is given by Vapnik (1999):

$$\begin{array}{lll} \text{minimize} & J(w,\varepsilon) = \dfrac{1}{2} w^t w + C \displaystyle\sum_{j=1}^{m} \varepsilon_j & \\ \text{subject to} & y_j(w^t x_j + c) \geq 1 - \varepsilon_j & j = 1,...,m \\ & \varepsilon_j \geq 0 & j = 1,...,m. \end{array} \tag{10}$$

Minimizing the norm of the weight vector is equivalent to maximizing the margin between the samples and the hyperplane. The *slack* variables ε_j allow for some samples to be at a distance smaller than the margin to the separating hyperplane or even on the wrong side of the hyperplane. 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, if we wish to transform the input features with a certain function $\phi(x)$ we only need to be able to compute the inner products between the transforms for any pair of samples, that is, we only need to know the function $K(x_k, x_l) = \phi(x_k)^t \phi(x_l)$. This fact is what allows for complex transforms of the input features to be used, as long as the kernel

function $K(x_k, x_l)$ can be easily computed. In the next section we will develop the proposed method considering an inner product kernel. The general case will be treated in Section 4.5.

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 (Vapnik, 1999; Smola and Schölkopf, 1998) 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, which again allows for the kernel trick to be used. Hence, even though the derivations in this paper will be done considering a classification problem for simplicity, the method described and the interpretations given can be equally applied to SVM regression problems.

4.2 Modified Support Vector Machines

As we saw in Section 3.3, reducing the ρ value between system S and system B could lead to an improvement in combination performance as long as the performance of the individual systems does not degrade too greatly. We propose to add a term in the SVM objective function ($J(w, \varepsilon)$ in Equation (10) that introduces a cost for a model that results in high value of ρ. Ideally, we would like this term to be given by $\lambda\rho$ so that low values of ρ are encouraged, including negative ones. Unfortunately, this term would make the new objective function nonconvex, making the optimization problem much more complex. To see this, let us derive an expression for ρ as a function of the SVM weights. Let $S = w^t X + c$, where w and c are the SVM parameters. We can compute $\sigma_{12} = \sigma_{SB}$ (component $1,2$ of Equation 7) as[5]

$$\sigma_{SB} = \frac{1}{2} \sum_{y=\{a,b\}} E[(B - \mu_{b,y})(S - \mu_{s,y})|Y = y] = w^t K,$$

where we are using the notation $\mu_{v,y} = E[V|Y = y]$ for any random variable V (scalar or vectorial), and where

$$K = \frac{1}{2} \sum_{y=\{a,b\}} E[(B - \mu_{b,y})(X - \mu_{x,y})|Y = y]. \tag{11}$$

K is simply the vector of average class-conditional covariances between each input feature and the scores from system B. The value of vector K can be estimated from the training set T as

$$\tilde{K} = \frac{1}{2} \sum_{y=\{a,b\}} \frac{1}{m_y} \sum_{j|y_j=y} (b_j - \tilde{\mu}_{b,y})(x_j - \tilde{\mu}_{x,y}), \tag{12}$$

where b_j is the score generated by system B for sample j, m_y is the number of samples in T from class y, $\tilde{\mu}_{b,y} = \frac{1}{m_y} \sum_{j|y_j=y} b_j$, and $\tilde{\mu}_{x,y} = \frac{1}{m_y} \sum_{j|y_j=y} x_j$.

Similarly, we can compute $\sigma_{11} = \sigma_{SS}$ (component $1,1$ of Equation 7) as $w^t M w$, where M is the average class-conditional covariance matrix for the feature vector X; and $\sigma_{22} = \sigma_{BB}$ (component $2,2$ of Equation 7) as the average class-conditional variance for the B scores that we will call v. We can now write ρ^2 as

$$\rho^2 = \frac{\sigma_{SB}^2}{\sigma_{SS}\sigma_{BB}} = \frac{w^t K K^t w}{v\, w^t M w}.$$

5. We use B and S to refer to the systems and the random variables corresponding to the scores produced by these systems. The actual meaning should be clear from the context.

This expression and its square root are nonconvex functions of w. Hence, adding the term $\lambda\rho$ to the objective function of the SVM problem would make the optimization problem nonconvex. On the other hand, the numerator of ρ^2, σ_{SB}^2 is a convex function of w and, using it as a regularization term results in a new problem that is equivalent to a standard SVM problem with a new kernel function. To see this, write

$$\begin{aligned} J_\sigma(w,\varepsilon) &= \frac{1}{2}w^t w + \frac{\lambda}{2}w^t KK^t w + C\sum_i \varepsilon_i \\ &= \frac{1}{2}w^t A w + C\sum_i \varepsilon_i, \end{aligned}$$

where $A = I + \lambda KK^t$ is a symmetric positive semidefinite matrix. We can now change variables $\tilde{w} = Bw$, with B symmetric and $B^tB = A$ (i.e., B is a matrix square root of A and, since A is a positive definite symmetric matrix, it always exists and can be chosen to be real and symmetric) and write it as

$$\begin{aligned} &\text{minimize} && J_\sigma(\tilde{w},\varepsilon) = \tfrac{1}{2}\tilde{w}^t\tilde{w} + C\textstyle\sum_j \varepsilon_j \\ &\text{subject to} && y_j(\tilde{w}^t z_j + c) \geq 1 - \varepsilon_j && j = 1,\ldots,m \\ &&& \varepsilon_j \geq 0 && j = 1,\ldots,m, \end{aligned}$$

where

$$z_j = B^{-1}x_j. \tag{13}$$

We see that the appropriate choice of regularization term led us to a very simple new optimization problem. The disadvantage of this choice is that it does not *directly* achieve our goal of minimizing ρ. For example, negative values of ρ corresponding to negative values of σ_{SB} will generally not be encouraged by the new objective function because the maximum margin (that is, the minimum value of w^tw) corresponds, in most practical cases, to positive values of σ_{SB}. Hence, in practice, for each negative value of σ_{SB} the corresponding positive one will result in a smaller value of the objective function and will be preferred to the negative one. Furthermore, minimizing σ_{SB} does not imply minimization of ρ, even for positive values, since the denominator is not being taken into account. Nevertheless, we empirically find that the new optimization problem achieves its goal of reducing ρ. In particular, when λ is large, σ_{SB} is pushed toward zero, forcing ρ to become zero.

Directly finding the matrix B^{-1} in (13) is computationally expensive since in general the dimension d of the feature vectors x_i can be very large and the matrix B is a full matrix of size $d \times d$. Nevertheless, since matrix A has a very particular structure, we can find an expression for its inverse using the matrix inversion lemma, by which $A^{-1} = I - \frac{\lambda}{1+\lambda K^tK}KK^t$. Hence, one way of implementing the proposed method is to define a kernel $K(x_k,x_l) = x_k^tB^{-t}B^{-1}x_l = x_k^tA^{-1}x_l$ to be used by the SVM. This kernel satisfies the mercer conditions (i.e., it is a valid kernel) since A is a positive semidefinite matrix. Using the expression for A^{-1} above we get

$$K(x_k,x_l) = x_k^tx_l - \frac{\lambda}{1+\lambda K^tK}x_k^tKx_l^tK. \tag{14}$$

We call this kernel the *anticorrelation kernel*. The computation of $K(x_k,x_l)$ in Equation (14) requires only the calculation of three inner products, which makes the method computationally feasible. Another approach is to transform directly the features using (13). This is also computationally feasible because the inverse of the matrix B has a simple expression. To find this expression we first

note that the matrix B^{-1} is a matrix square root of A^{-1}. Hence, we need to find a matrix that when multiplied by its transpose results in $A^{-1} = I - \frac{\lambda}{1+\lambda K^t K} K K^t$. It is easy to show that

$$B^{-1} = I - \frac{\alpha}{K^t K} K K^t$$

satisfies this condition when $\alpha = 1 \pm \frac{1}{\sqrt{1+\lambda K^t K}}$. Hence, given a certain value for λ we can find the corresponding α and transform each feature vector using (13). This means that we can implement the proposed method by transforming the input features using the following expression:

$$z_i = x_i - \alpha \frac{K^t x_i}{K^t K} K. \tag{15}$$

In the case of speaker verification, a separate K vector is computed for each target model being trained. Hence, doing the transformation in the feature domain is inefficient, since there is not a single transformation for each feature vector x_i, but one for each target model. The kernel implementation might be preferable in this case. Furthermore, as we will see in Section 4.5, when the original SVM problem uses a kernel other than the inner-product one, implementing the anticorrelation method as a kernel may be the only feasible option.

4.3 Interpretation of the Modified Problem

To give an interpretation of the new SVM problem, we first need to understand the meaning of the direction given by the vector K. The average class-conditional covariance between the scores from system B and the scores from system S is given by $w^t K$. For a fixed value of $\|w\| = c$, the w that maximizes the absolute value of $w^t K$ is given by $w = cK/\|K\|$. Hence, K gives the direction for the vector w of SVM weights for which the average class-conditional covariance between the two systems is maximum. A w orthogonal to K would result in zero average class-conditional covariance between the two systems. The term $\|w^t K\|^2$ that we have added to the objective function of the SVM problem has the effect of penalizing any w vector with a large component in the direction of K. Our goal is to find a w as orthogonal to K as possible without degrading the performance of the system so much that the overall combination starts to degrade. This balance can be achieved by tuning the parameter λ.

We can interpret the kernel given by (14) in a similar way. When λ is small this kernel is close to the linear kernel. When λ grows to infinity the kernel subtracts the product of the projections of the points x_k and x_l into the vector K from the linear kernel. The resulting value of the kernel will be small if x_k and x_l are both aligned with K. Since the SVM will make an effort to separate only points from different classes that give a high kernel value (that is, that are more "similar"), this means that we consider vectors whose directions are close to that of K to be unimportant and, consequently, we emphasize the importance of the vectors whose directions are orthogonal from that of K. This results in a more effective usage of the features, ignoring those directions that would lead to high average class-conditional covariance between the systems and taking advantage of the rest.

Finally, if we choose to implement the method as a feature transform instead of a kernel function, the resulting features have a very simple interpretation. When $\lambda = \infty$, Equation (15) becomes $z_j = x_j - \frac{K^t x_j}{K^t K} K$, which is the expression for subtracting from x_j its component on direction K. If λ is not ∞ then only a part of the component is subtracted.

4.4 Extension for Multiple Preexisting Scores

An extension to the presented method can be considered where N previous systems are available, $B_1,\ldots,B_N$, and we wish to train S to combine well with them. A generalization of the formulas above can be derived for this setup. We rewrite the objective function as

$$\begin{aligned} J_\sigma(w,\varepsilon) &= \frac{1}{2}w^t w + \sum_{k=1}^{N}\frac{\lambda_k}{2}w^t K_k K_k^t w + C\sum_i \varepsilon_i \\ &= \frac{1}{2}w^t A w + C\sum_i \varepsilon_i, \end{aligned}$$

where now A is given by $I+\sum_{k=1}^{N}\lambda_k K_k K_k^t$ and it is still positive definite and symmetric. The same approach used above can be used here to simplify the problem to a standard SVM problem. We can still use the inversion lemma by considering matrices

$$\begin{aligned} K &= [K_1 \ldots K_N], \\ \Lambda &= \mathrm{diag}(\lambda_1 \ldots \lambda_N), \end{aligned}$$

so that $I+\sum_{k=1}^{N}\lambda_k K_k K_k^t = I + K\Lambda K^t$. We can then use the lemma to get $A^{-1} = I - K(\Lambda^{-1} + K^t K)^{-1}K^t$. When $\lambda_k = \infty$ for all k, $A^{-1} = I - K(K^t K)^{-1}K^t$. This matrix is idempotent (and symmetric), hence $B^{-1} = A^{-1}$. The transformed features z_i for this case are then given by $z_i = x_i - K(K^t K)^{-1}K^t x_i$, which is the projection of x_i on the complementary space to that spanned by vectors K_1 through K_N.

4.5 Extension for General Kernels

The development on Section 4.2 was done using inner-product kernel SVMs as the starting point. In this section we show that the method can be implemented for any kernel function.

Consider a problem for which $K_0(x,y) = \phi_0(x)^t\phi_0(y)$ has been found to perform better than the inner-product kernel. One way of implementing the anticorrelation method in this case is to simply transform the features using $\phi_0(x)$ and then treat the transformed features as the feature vectors x in Section 4.2. This is conceptually simple, but could be extremely costly computationally if the dimension of $\phi_0(x)$ is large compared to the dimension of x, or impossible if the transformation ϕ_0 is infinite dimensional (as in the case of the Gaussian kernel). Luckily, there is a way of implementing the anticorrelation method without ever computing the transform but only the kernel function between pairs of features.

In Section 4.2 we found that one way of implementing the proposed method is by the use of the *anticorrelation* kernel, defined by Equation (14). In practice, vector K in that equation is computed from data. We call this empirical value for K, $\tilde{K}$ (Equation 12). We can write $\tilde{K}$ as a linear function of the features x_j used to compute it. That is, $\tilde{K} = \sum_j c_j x_j$, where the c_j depend on the m_y's and all the b_i's. If we now replace every x in Equation (14) by $\phi_0(x)$ and K by $\sum_j c_j\phi_0(x_j)$, we get

$$\begin{aligned} K(x_k,x_l) &= \phi_0(x_k)^t\phi_0(x_l) - \frac{\lambda\sum_j c_j\phi_0(x_j)^t\phi_0(x_k)\sum_j c_j\phi_0(x_j)^t\phi_0(x_l)}{1+\lambda\sum_j\sum_i c_j c_j\phi_0(x_i)^t\phi_0(x_j)} \\ &= K_0(x_k,x_l) - \frac{\lambda\sum_j c_j K_0(x_j,x_k)\sum_j c_j K_0(x_j,x_l)}{1+\lambda\sum_j\sum_i c_j c_j K_0(x_i,x_j)}. \end{aligned}$$

The anticorrelation kernel can then be computed exclusively as a function of the original kernel K_0. The processing time is now significantly increased, though, since two sums over the samples used to obtain K are needed every time the kernel is computed (the denominator in the second term can be precomputed and reused, since it does not involve x_k or x_l). The extension for multiple preexisting scores follows the same steps as above. In this paper, the inner-product kernel is used for all experiments.

4.6 Other Approaches

Our goal is to obtain the best possible combination performance given the available systems. The approach presented above is one path toward this goal. Two other ways of approaching this problem are considered here.

4.6.1 FEATURE-LEVEL COMBINATION

When system B is also an SVM system and the features corresponding to the samples used for training system S are also available for system B, an SVM using the features from both systems concatenated into a single vector can be trained. The resulting SVM is in itself a combination procedure, which, ideally, should make optimal use of the features from both systems. This may not be true in practice, though, since a larger feature vector increases the complexity of the system, making it more prone to overfitting the training data. A further refinement of this approach consists of weighting the vector components, assigning weight α to the features from one of the original systems and weight $1-\alpha$ to the other features. This is done by multiplying the components of the square-norm of w in the SVM objective function by the inverse of the corresponding feature weight. That is, we replace $\|w\|^2 = \sum_i w_i^2$ with $\sum_i w_i^2/\beta_i$, where $\beta_i = \alpha$ for the features from one set and $1-\alpha$ for the features from the other set. This allows us to compensate for different lengths in the original vectors or to bias the training procedure to make more use of the features from the better-performing system. Feature-level combination is usually costly and sometimes even infeasible, given the large size of the original feature vectors, and can be considered only if both systems being combined are SVM systems.

4.6.2 FEATURE+SCORE COMBINATION

Another method can be considered in which we present the scores generated by system B as input features to the SVM, along with all the features from system S. Again, a larger weight can be given to the component corresponding to the score from system B than to the features from S.

5. Experiments on Artificial Data

To test the proposed kernel on a simple task, we generated data for two classes with model

$$Z = C\tilde{Z} + m_Y,$$

where $\tilde{Z}$ is a vector of size d where the components are generated independently with normal distribution with zero mean and unit variance. C is a square random matrix intended to create correlation between the features. Its components are drawn from a uniform distribution, and a scaling factor is

applied to force the maximum variance of Z to be 1. The class-dependent mean vector is given by

$$m_Y = \begin{cases} (0\ldots0)^t & \text{if } Y = a \\ (m\ldots m)^t & \text{if } Y = b. \end{cases}$$

We take half of the features and train a linear SVM (inner-product kernel), which serves as system B. The remaining features are used to train system S starting with an inner-product kernel. The anticorrelation kernel is implemented for varying values of λ. We create two separate sets, one for training, with 900 examples of class a and N examples of class b, and one for testing, with 10 times more data than in the training set.

The combination is performed using a linear logistic regression model, trained on the training set with the scores from the two SVM systems, B and S, for each value of λ. Since the scores obtained on the training set are overly optimistic, we use 10-fold cross-validation on the training set to create the B and S scores used to train the combiner (Kuncheva, 2004, Section 3.2.2). The scores from cross-validation for system B are also used to estimate the vector $\tilde{K}$ as in (12). When only a few samples from one of the classes are available, the estimation of $\tilde{K}$ can be noisy. In our simulation we vary the number of samples available for class b, keeping the number for a fixed; hence, in order to keep the variance of the estimator stable across experiments, we use only samples from class a to estimate $\tilde{K}$.

Figure 5 shows the error rates for the test data for system S, system B, and the score-level combination as a function of the value of λ, for $m = 3.0$, $N = 900$ and $d = 250$. The figure also shows results for the feature-level and the feature+score combination procedures, explained in Section 4.6. The weight α for these two systems was tuned using 10-fold cross-validation on the training set. For these two cases and for system B, the error does not depend on λ. On the other hand, the value of ρ between S and B decreases with λ (reaching a value close to zero). We see that, in practice, ρ is effectively reduced as λ increases, even though we use $\lambda\sigma^2_{SB}$ as regularization term instead of $\lambda\rho^2$. The error for system S also varies with λ. The degradation in performance is expected since we are trading off poorer performance in exchange for a lower value of ρ. The small improvement at moderate values of λ, though, is not too surprising. If the direction K corresponded to one that is especially noisy, reducing the importance of that direction can lead to improved performance. We will see more on this in Section 6.5.

The feature-level and feature+score combination methods perform approximately equal at around 1% EER, while, for $\lambda = 0$, the score-level combination has a significantly worse performance of 1.58%. Nevertheless, as λ grows, the performance of the score-level combination using the anticorrelation kernel improves significantly (from 1.58% to 0.56% when λ goes from 0 to 10^4), making it the best-performing system. Overall, we see a reduction in EER of around 50%, relative to the EER of the best combined system when the anticorrelation kernel is not used.

Figure 6 shows the scatter plot of scores (on the training data) for both systems corresponding to $\lambda = 0$ and $\lambda = 10000$. We can see that for the large value of λ, the within-class covariances have been largely reduced. We can also see that the separation of the two classes is better for the larger λ, which explains the performance improvement observed in Figure 5.

Figure 7 shows the results for the score-level combination of systems B and S with $\lambda = 0$ (that is, without using the proposed method), the score-level combination with $\lambda = \infty$, the feature-level combination, and the feature+score combination, for several settings of the simulation parameters N, m, and d. For the score-level combination with $\lambda = \infty$ we also present the results obtained when computing K using our knowledge of the model that generated the data. Since we are creating

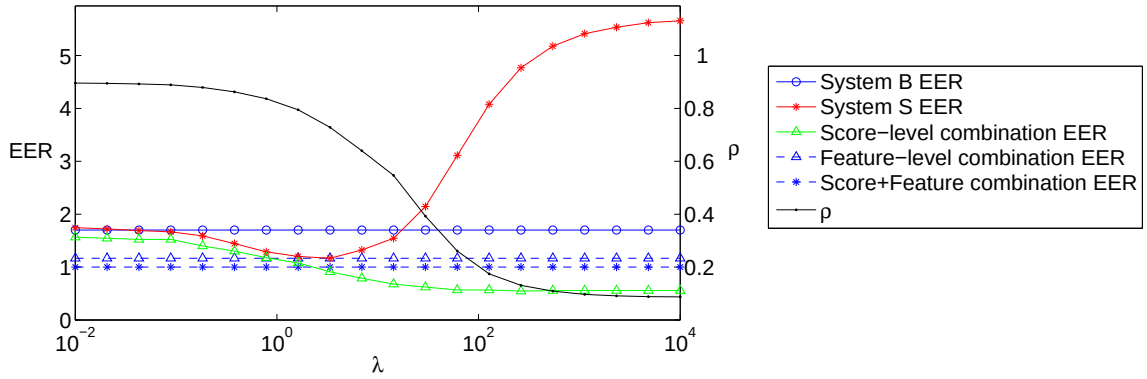

Figure 5: Error of individual systems and their combination, and value of the ρ coefficient as a function of λ for an artificial problem. The EER of the combination is reduced from 1.58% to 0.56% as λ increases.

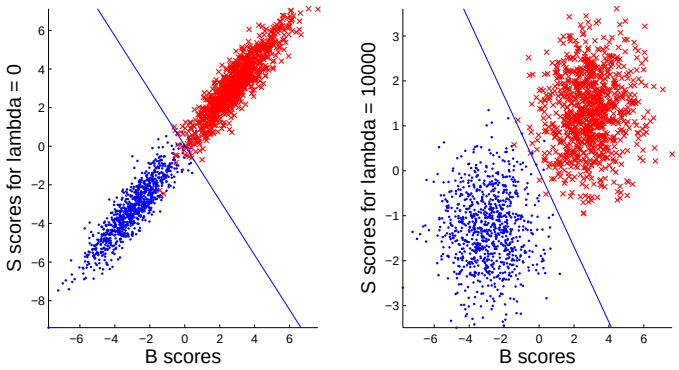

Figure 6: Scores from system B versus scores from system S for two values of λ.

the data ourselves according to a model, we can compute K exactly using (11) instead of (12). It can be shown that, for our setup, $K = \frac{1}{2}\sum_{y=\{a,b\}}(C_{12}C_{11}^t + C_{22}C_{12}^t)w_B$, where w_B is the SVM weight vector for system B and C_{ij} is block ij of size $d/2 \times d/2$ of matrix C. For each set of parameters, 10 different random seeds were used to generate the data, keeping matrix C equal for all 10 experiments. Each bar shows the first quartile, the median, and the third quartile of the set of EERs obtained from the 10 simulations.

The figure shows that feature+score combination is significantly better than score-level or feature-level combinations only when the task is easier ($m = 3.0$) and many samples are available for training ($N = 900$). Feature-level combination is optimal when the task is harder ($m = 1.8$) and the number of features is small ($d = 250$) particularly when enough samples are available for training ($N \geq 300$). Plain score-level combination (without anticorrelation) performs comparably to feature-level combination when the number of training samples is small and the number of features is large, in which case the feature-level combination suffers from the additional complexity. In all cases, though, the anticorrelation method (last two bars) is either comparable or significantly better than

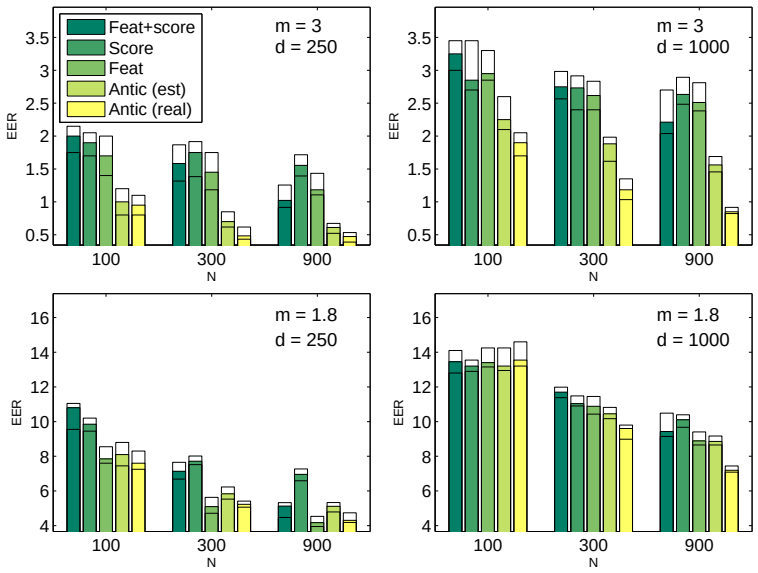

Figure 7: Comparison of EER on simulated data for the score-level combination of system B with S with $\lambda = 0$ (called *Score* in the legend), the score-level combination with $\lambda = \infty$ (*Antic*) using the estimated value for K (*est*) or the value obtained from the model (*real*), the feature-level combination (*Feat*), and the feature+score combination (*Feat+score*) for several values of the simulation parameters. For each pair of m (distance between means) and d (feature vector dimension), three values of N (number of samples from class b) are explored.

plain score-level, feature+score and feature-level combinations. For $m = 3.0$, the anticorrelation method significantly outperforms all other combination methods for both values of d and the three values of N. Gains are smaller or disappear when the task becomes harder ($m = 1.8$).

The difference between the fourth and the fifth bars in each set is due only to the difference in the K vector used. The K is estimated using the data (Equation 12) for the fourth bar and using the model (Equation 11) for the fifth bar. We can see that when the dimension of the feature vector Z_2 is large, the difference between the fourth and the fifth bars becomes larger, indicating that in these situations the estimation of K is noisier. This is also evident from looking at the reduction in ρ achieved for the different values of d when using the data to estimate K. For $d = 250$, the reduction is around 90%, while for $d = 1000$ the reduction is only around 60%. In most experiments with $d = 1000$ the value of ρ when using the estimated value of K does not go under 0.30. This means that for higher-dimensional vectors, the estimation of K is harder than for lower-dimensional ones. Nevertheless, even in those cases, the combination using the anticorrelated system with the estimated K is, in most cases, better than the original score-level combination.

6. Experiments on Speaker Verification

Speaker verification is the task of deciding whether or not a speech sample was produced by a certain target speaker. It is a binary classification task where the two classes are *true-speaker* and *impostor*. To test the proposed method we use a standard UBM-GMM system, a cepstral supervector SVM system, an MLLR-based system, and a prosodic system. We show results using the proposed kernel on all possible combinations involving two systems (two-way combinations) and a variety of combinations involving three and four systems (three-way and four-way combinations).

6.1 Databases and Error Measures

Experiments were conducted using data from the NIST speaker recognition evaluations (SRE) from 2005 and 2006. 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. We consider the 1-side training conditions in which we are given 1 conversation side to train the speaker model. This conversation corresponds to a positive example when training the SVM model for the speaker. The data used as negative examples for the SVM training and to estimate the K vectors is taken from 2003 and 2004 NIST evaluations along with some FISHER data, resulting in a total of 4355 samples. The tasks contain 26,270 trials for SRE05, and 21,343 for SRE06. In both cases around 1/10th of the trials are target trials. Trials are created by reusing the conversations from a few hundred speakers as train and test samples, sometimes as target speakers, sometimes as impostors. A total of 598 distinct models for SRE05 and 584 for SRE06 are created, some of them corresponding to different conversations from the same speaker.

The performance measures used in this section are the EER and NIST's detection cost function (DCF). The DCF is 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. The DCF is affected both by the discrimination power of the system and its calibration, given by the choice of threshold that is believed to minimize it (Brummer and du Preez, 2006). In this paper, we will not explore the calibration issue, which is, in itself, a large field of study in the biometrics community. We will present results in terms of the DCF achieved when choosing the threshold that minimizes it on the test data. This measure is commonly called *minimum* DCF and it measures *how much information the detector could have delivered to the user, if the calibration had been perfect* (Brummer and du Preez, 2006).

The EER and the DCF are two points in the receiver operating characteristic (ROC) curve of a system and they give a more complete picture of the behavior of the system for different operating points than the EER alone. Even though the theory in Section 3 was developed for EER, we will see that improvements are obtained for both performance measures.

6.2 Individual System Descriptions

The systems chosen to run the experiments in this paper are representative of the systems being used in most state-of-the-art speaker recognition systems. Somewhat simplified versions of the best-performing systems were used, in order to facilitate the large amount of computationally costly experiments that were run. A brief description for each of the systems follows.

6.2.1 UBM-GMM SYSTEM (G)

This is probably the most widely used paradigm for speaker verification. A Gaussian mixture model (GMM) is trained using data from many different speakers and recording conditions to create a *universal* background model (UBM). The target speaker models are trained by maximum a posteriori adaptation of the background model to the training data. For a given test sample the logarithm of the ratio of the likelihoods for the target model and the background model is used as a score. The system used here is based on 13 mel frequency cepstral coefficients (MFCCs) without the zeroth-order coefficient, and first-, second-, and third-order difference features, resulting in 52-dimensional feature vectors. The features are modeled by 2048 mixture component GMMs. Only the GMM means are adapted to the observed data, leaving variances and weights untouched. For implementation details on this system see Shriberg et al. (2005).

6.2.2 SUPERVECTOR-SVM SYSTEM (V)

This system (Campbell et al., 2006) is a variation of the UBM-GMM system, where SVMs are used to obtain scores. For each sample, the means of the UBM-GMM are adapted to the sample's data and stacked together in a single high-dimensional feature vector. A set of held-out samples (generally the same samples used to create the UBM-GMM) is used as negative examples when training the SVM, while the target sample is used as the positive example. These features are used to train a model using support vector regression with an inner-product kernel. The signed distance to the hyperplane is then used as the output of the system. For this system we use a 512-component background model. Since the dimension of the original space is 52, the final dimension of the feature vectors is given by $512 \times 52 = 26,624$. Larger background models have been found to give slightly better performance but increase the computational cost of the experiments. We found 512 components to give a good balance between performance and computational cost of the system.

6.2.3 MLLR-SVM SYSTEM (M)

The MLLR-SVM system (Stolcke et al., 2007, 2006) uses the speaker adaptation transforms used in the speech recognition system as features for speaker verification. A total of four affine 39x40 transforms is used to map the Gaussian mean vectors from speaker-independent to speaker-dependent speech models; two transforms each are estimated relative to male and female recognition models, respectively. The transforms are estimated using maximum-likelihood linear regression (MLLR) and can be viewed as a text-invariant encapsulation of the speaker's acoustic properties. The transform coefficients form a 6,240-dimensional feature space. Each feature dimension is rank normalized by replacing the value with its rank in the background data, and scaling ranks to lie in the interval [0, 1]. The resulting normalized feature vectors are then modeled using the same procedure as for the supervector-SVM system. The system described in this paragraph is a simplified version of our best performing MLLR-SVM system which uses a total of 16 transforms and has approximately 25% lower EER than the 4-transform system (Stolcke et al., 2007). Initial experiments (not shown here) indicate that improvements from the anticorrelation method are still obtained when the more complex MLLR system is used, with similar relative gains as the ones shown here.

6.2.4 SNERF-SVM SYSTEM (S)

This system models syllable-based prosodic NERFs (nonuniform extraction region features) (Shriberg et al., 2005). Features are based on estimated F_0, energy, and duration information extracted over syllables inferred via automatic syllabification based on automatic speech recognition output. Prosodic feature sequences are transformed into fixed-length vectors by a particular implementation of the Fisher score (Ferrer et al., 2007). In this paper, only features modeling sequences of two syllables are used. In previous work we have found that these features by themselves yield a performance almost as good as using features extracted for sequences of 1, 2, and 3 syllables together. The resulting feature vector, of dimension 13,343, is first rank-normalized (as in the MLLR system) and modeled using the same procedure as for the supervector-SVM system.

6.3 Application of the Proposed Method to the Speaker Verification Problem

Most speaker verification systems that use SVMs as models consider each train or test utterance as a single sample. If necessary, as in the case of the SNERF features and many other cases presented in the literature (Ferrer et al., 2006; Brummer et al., 2007; Reynolds et al., 2005), a transform is applied to the input features prior to SVM modeling in order to convert them into a single fixed-length vector. In other cases, such as the MLLR system, the features are directly generated as a single fixed-length vector. In our experiments, since we are presenting results on the 1-side training condition from NIST evaluations, this implies that only one positive sample is available during training for each speaker model. This means that the estimation of K in (12) will be given only by impostor samples. These impostor samples are extracted from a held-out set. For each target model in the task definition we require a separate vector K. This results in significant overhead during training since each model from system B must be tested against the held-out set used to compute K. Nevertheless, this has no effect at test time. Once the vector K for each target model is computed, obtaining the score for a new test is almost as fast as for a linear kernel SVM.

6.4 Results

Table 1 shows the results on SRE05 and SRE06 data for the individual systems. Each system is represented by a single letter: **G** for the GMM-UBM, **V** for the Supervector-SVM, **M** for the MLLR-SVM, and **S** for the SNERF-SVM. For the SVM systems (Supervector, MLLR, and SNERFs), we show the baseline results (training the SVM with an inner product kernel) and the results obtained by training the target SVMs using the kernel in (14) with K computed using the scores corresponding to each of the other three systems. This is indicated by the use of a subindex corresponding to the system with respect to which the anticorrelation is performed. For example, M_{G} corresponds to a system that uses the MLLR features and anticorrelation kernel with respect to the GMM-UBM system (that is, with K given by the vector of average class-conditional covariances between the MLLR features and the scores from the GMM-UBM system). A list of subindices corresponds to performing anticorrelation with respect to more than one system as described in Section 4.4. Hence, system $\mathrm{S}_{\mathrm{M,V_M}}$ corresponds to system S anticorrelated with respect to systems M and V_{M}. In all cases the anticorrelation results shown correspond to $\lambda = \infty$, which implies that the resulting weight vector will not have a component in the direction of K. This was shown to be optimal in the simulated experiments and in several preliminary experiments with the systems from this table.

System	SRE05		SRE06	
	DCF	EER%	DCF	EER%
G	0.303	7.259	0.306	5.639
M	0.266	7.096	0.221	4.979
M_G	0.297	8.564	0.243	5.879
M_V	0.289	8.768	0.254	6.119
M_S	0.271	7.586	0.229	4.979
V	0.205	5.465	0.174	3.419
V_G	0.223	6.525	0.179	4.019
V_M	0.196	5.465	0.168	3.179
V_S	0.203	5.506	0.168	3.359
S	0.545	14.233	0.548	12.777
S_G	0.560	15.008	0.562	13.077
S_M	0.577	15.824	0.579	13.497
S_V	0.569	15.253	0.573	13.137
S_{M,V_M}	0.588	16.150	0.592	14.157

Table 1: Results for individual systems (G: GMM-UBM, V: Supervector-SVM, M: MLLR-SVM, S: SNERF-SVM) with inner product kernel and anticorrelation kernel. When the anticorrelation kernel is used, a subindex indicates the name of the system or systems with respect to which the anticorrelation is performed.

It can be seen that in most cases, using the anticorrelation kernel results in a degradation in performance in the system. A notable exception is the result for system V_M (Supervector features using anticorrelation kernel with respect to the MLLR-SVM system). In this case, preventing the use of the direction given by K results in a significant gain in performance. This could happen if vector K corresponded to some noisy direction that, when ignored, allowed for other more robust directions to be used. This effect was also observed in the simulations and will be discussed in more detail in Section 6.5.

Table 2 shows results for all the possible two-way combinations of the four individual systems. For the score-level combinations (indicated with "+") we used a linear model trained on SRE05 data using logistic regression. Feature-level and feature+score combinations are indicated with "$\cup$". The symbol indicates feature concatenation. In the case of feature-level combination, features from both systems are directly concatenated. In the case of feature+score combination, the features from one system are concatenated with the scores from the other (this is indicated with the subscript "scores"). Whenever feature concatenation is performed, a weight is used (as described in Section 4.6) to emphasize the features from one set versus the other. This weight is tuned on SRE05. Since tuning this weight is extremely demanding computationally (it requires running a full classification experiment for each value of the weight), only feature+score combination experiments involving the MLLR systems are run. All possible feature-level combinations are shown, since there are only three of them.

We can see that every time a score-level combination (+) is done between systems X and Y_X,[6] the performance is better than that for the combination of X and Y (with the single exception of SRE05's EER for $V + S_V$). That is, applying the anticorrelation kernel to system *Y* always gives a gain in the combination performance, even though in most cases system Y_X has worse individual performance than system Y. Feature-level combinations, $X \cup Y$, do not show any advantage over the much simpler score-level combination $X + Y$. This simply indicates that the increased model complexity of the feature-level combination cannot be properly handled with the available amount of training samples. Similarly, the feature+score combinations, $X \cup Y_{\text{scores}}$, do not give any consistent improvement over the score-level combinations $X + Y$.

The most notable gain from using the anticorrelation kernel is found for the combination $M + V_M$. The relative gain in EER with respect to system $M + V$ is 16%. As mentioned above, system V_M is in fact better performing than system V. That is, anticorrelating with respect to M does not degrade its performance but improves it. This fact, together with a reduced impostor correlation between the systems, explains the observed large gain.

The last column in Table 2 shows the class-conditional correlation between the two systems being combined for the impostor and the target samples in SRE06 data. As we can see, the impostor correlation is significantly reduced when the proposed method is used, even though it does not reach a zero value. (This problem was also observed in the simulated experiments for large values of the feature vector dimension *d*.) This could mean that the amount of data used for the computation of *K* (4355 samples) is not enough to obtain a robust estimation of the statistics in the test data or that the statistics in the test data are not the same as those in the held-out set used to compute *K*. Furthermore, we can see that the target correlation remains almost unchanged by the application of the anticorrelation kernel. This is reasonable, since the vector *K* is computed without the use of any target data. The fact that the target correlation is not reduced when *K* is computed only over impostor samples suggests that the correlations in both populations are not equal and one cannot be predicted from the other. Nevertheless, a reduction in either of the class-conditional correlation coefficients can result in a reduction of ρ as given by (9).

Finally, Table 3 shows some three-way and four-way combination results. The first four results correspond to the combination of the three SVM systems. The three combinations shown that use the anticorrelation kernel on the V and S systems perform very similarly, resulting in a performance improvement of 18.9% on the SRE06 EER with respect to the baseline combination. We can see that using multiple anticorrelation on the S system with respect to the other two systems already in the combination (M and V_M) does not lead to further improvements (line $M + V_M + S_{M,V_M}$). This is, in fact, good news, since doing the multiple anticorrelation involves a significant amount of extra computation to obtain the *K* vector of system S with respect to the scores from V_M.

The last three lines in Table 3 show the results on some four-way score-level combinations. We see a large improvement of 19.2% when a successive anticorrelation procedure is used, where each new model is anticorrelated with the one previously added to the combination. The best three- and four-way combinations all include the V_M system. This was expected since the two-way gain from using anticorrelation on that system was the largest among all two-way combinations.

An overall observation from this table is that the proposed method performs better on SRE06 data than on SRE05 data, even though the combiner is trained on SRE05 data, making the SRE05 results slightly optimistic. We believe that this might be a consequence of a better statistical match

6. Letters X and Y are used in this section to indicate any two systems. Hence, $X, Y \in \{G, M, V, S\}$.

System	SRE05		SRE06		
	DCF	EER%	DCF	EER%	CorI / CorT
$G_{scores} \cup M$	0.219	5.710	0.189	4.079	-
$G + M$	0.226	5.750	0.201	4.019	0.51/0.79
$G + M_G$	**0.210**	**5.465**	**0.188**	**3.779**	0.26/0.76
$G + V$	0.203	5.383	0.191	3.419	0.74/0.88
$G + V_G$	**0.194**	**5.261**	**0.182**	**3.239**	0.35/0.86
$G + S$	0.219	5.587	0.232	4.499	0.16/0.45
$G + S_G$	**0.216**	**5.465**	**0.224**	**4.259**	0.11/0.44
$M \cup V$	0.188	5.383	0.164	3.239	-
$M \cup V_{scores}$	0.176	5.098	0.148	3.119	-
$M + V$	0.180	5.139	0.160	3.299	0.58/0.87
$M + V_M$	**0.161**	**4.812**	**0.140**	**2.759**	0.37/0.84
$M_V + V$	0.171	4.976	0.142	3.179	0.31/0.83
$M \cup S$	0.245	6.403	0.200	4.379	-
$M \cup S_{scores}$	0.225	**5.913**	**0.190**	4.439	-
$M + S$	0.224	6.158	0.194	4.319	0.22/0.55
$M + S_M$	0.221	5.995	0.191	**4.079**	0.14/0.52
$M_S + S$	**0.215**	5.995	0.191	**4.079**	0.15/0.53
$V \cup S$	0.183	5.057	0.152	3.179	-
$V + S$	0.163	4.609	0.146	3.239	0.19/0.50
$V + S_V$	0.161	4.812	0.145	3.119	0.13/0.49
$V_S + S$	**0.159**	**4.527**	**0.137**	**2.999**	0.15/0.49

Table 2: Results for two-way combinations of the systems in Table 1. Symbol "$\cup$" indicates feature concatenation. Hence, $M \cup S$ corresponds to feature-level combination, while $M \cup S_{scores}$ corresponds to feature+score combination of systems M and S. Symbol "$+$" indicates score-level combination. The last column shows the correlations between the pair of systems being combined, for the impostor (CorI) and the target (CorT) samples.

between SRE04 data (used to compute the *K* vectors) and SRE06 data than between SRE04 and SRE05 data.

Evidently, the behavior found in these experiments cannot be expected to generalize to all possible sets of features and tasks. For example, if much smaller sets of features were used and enough training data was available for both classes, feature-level combination might result in better performance than score-level combination (as seen in the simulated experiments). Nevertheless, the systems used here are representative of the kinds of systems used for speaker recognition on state-of-the-art systems, where very large feature vectors have been found to outperform smaller ones. Furthermore, the small amount of positive training examples is an inherent characteristic of the speaker recognition task. Finally, these characteristics are found in many other modern machine learning tasks, a notable example being classification of microarray expression data, where the

System	SRE05		SRE06	
	DCF	EER%	DCF	EER%
$M+V+S$	0.149	4.690	0.134	3.179
$M+V_M+S_M$	0.133	**4.364**	0.117	**2.579**
$M+V_M+S_V$	**0.132**	4.445	**0.115**	**2.579**
$M+V_M+S_{M,V_M}$	0.133	4.405	0.116	**2.579**
$G+M+V+S$	0.149	4.649	0.141	3.119
$G+M_G+V_G+S_G$	0.147	4.323	0.132	2.700
$G+M_G+V_M+S_V$	**0.137**	**4.160**	**0.120**	**2.519**

Table 3: Results for three-way and four-way combinations of the systems in Table 1.

number of features is in tens of thousands, and the number of samples for most studies is limited to tens or at most hundreds.

6.5 Interaction with Intersession Variability Compensation

The variability found across different recordings of the same speaker is commonly called *intersession variability* (ISV). This effect can be caused by a mismatch in channel conditions, emotional state, phonetic content, and so on, and it is one of the biggest sources of errors in speaker verification. Several methods have been developed to reduce the effect of intersession variability.

In the realm of SVMs, the most widely used ISV compensation (ISVC) method is nuisance attribute projection (NAP) (Solomonoff et al., 2004; Campbell, 2006). NAP consists of estimating the directions in the feature space that vary with the sessions and then projecting the samples on the complement of the space determined by those directions. The *noisy* directions are calculated as the first few eigenvectors of the within-speaker covariance matrix. This matrix is in turn estimated from held-out data for which several samples of each speaker are available. All speakers are pulled together and a single within-speaker covariance matrix is estimated. The number of directions to be eliminated from the feature vectors is determined empirically. Both NAP and the anticorrelation method presented here transform the features by eliminating certain directions. In the case of NAP these directions are the ones estimated to have information superfluous to the task of speaker verification. In the anticorrelation procedure, a single direction is eliminated: the one that maximizes the average class-conditional covariance between the two systems being combined.

In the case of UBM-GMM systems and systems like the supervector-SVM, which are based on the UBM-GMM models, a different type of ISVC based on the factor analysis method can be applied (Kenny and Dumouchel, 2004; Kenny et al., 2007). The method is based on the assumption that a supervector m corresponding to a certain sample can be decomposed into a speaker-dependent and a channel-dependent component. That is, $m = s + c$, where s is a speaker supervector and c a channel supervector. Furthermore, c is assumed to be given by ux, where u is a low-rank matrix and x is a normally distributed random vector. The components of vector x are called the *channel factors* and the columns of matrix u the *eigenchannels*. In order to estimate the matrix u, a database with several samples for each speaker (as the one required for NAP) is needed. In some models, s is further decomposed into different terms. Many different methods have been used to estimate and

System	SRE05		SRE06		SRE06
	DCF	EER%	DCF	EER %	CorI / CorT
M	0.230	6.525	0.195	4.139	-
M_V	0.257	7.667	0.217	4.919	-
V	0.171	4.935	0.145	2.819	-
V_M	0.177	5.302	0.144	2.879	-
$M+V$	0.144	4.568	0.120	2.639	0.47/0.87
$M+V_M$	0.144	4.690	0.117	2.579	0.40/0.86
M_V+V	0.142	4.649	0.119	2.639	0.36/0.85

Table 4: Results for individual systems and their combinations after ISVC.

compensate for the channel factors. The method used in this paper for the supervector system is described by Matrouf et al. (2007).

Table 4 shows a subset of results for two of the systems described in Section 6.2 when ISVC is applied. Factor analysis is used for the Supervector-SVM system, and NAP is used for the MLLR-SVM system. We can see that, for these systems, the anticorrelation kernel does not reduce the class-dependent correlations between pairs of systems enough to result in a gain in the combination performance. The fact that applying the anticorrelation kernel does not result in a significant reduction of the class-dependent correlations between the systems indicates that the *K* vectors computed from the held-out data are not a good estimation of the *K* vectors in the test data. If we compare the impostor correlation between the same pair of systems when no ISVC is applied (Table 2) versus the impostor correlation when ISVC is applied (Table 4) we see that ISVC'ed systems are much less correlated. Furthermore, the correlation when the anticorrelation method is applied results in similar values for ISVC'ed and non-ISVC'ed systems. This suggests that much of the correlation between the non-ISVC'ed systems is due to intersession variability effects. This intuition is confirmed by computing the projection of the *K* direction on the NAP directions, which shows that *K* is mostly in the direction of the first few NAP directions (when they are sorted by the size of the corresponding eigenvalue). Hence, when ISVC is applied to a system, the ISV effects are eliminated (or reduced), resulting on less-correlated systems that combine better with each other. For example, the performance for combination $M+V$ for non-ISVC'ed systems (3.299% from Table 2) is only 3.5% better than the best of the two individual performances (3.419% from Table 1), while the performance for that same combination but using ISVC'ed systems (2.639% from Table 4) is 6.4% better than the best of the two individual performances (2.819% from Table 4).

These observations explain the reduction in class-conditional correlation between pairs of systems when ISVC is applied to them, since a large part of the class-conditional correlation is due to the intersession variability. On the other hand, it does not explain why the correlation cannot be further reduced after the intersession variability noise has been eliminated. The reason for this is simply that the vector *K* estimated for each target model does not predict the direction of maximum impostor covariance on the test data. On the training data, we know that the covariance for the impostor cloud is necessarily pushed to zero when $\lambda = \infty$, but we do not observe the same behavior on the test data. This is because, as we saw in the simulations, the estimation of the direction *K*

gets noisier for larger feature vectors. Hence, only a very noticeable effect (like the intersession variability one) can be robustly estimated.

We believe that the observations presented in this section do not invalidate the usefulness of the method for the speaker verification task. As mentioned in Kenny et al. (2007), for ISVC to work, a well-balanced database is required where samples from several different recording conditions for each speaker are available. This kind of database is not easy to obtain. When such a database is not available, ISVC cannot be applied to the systems. In these cases, the anticorrelation method proposed here would be able to bring back some of the gain that ISVC would result in if the right database was available.

7. Conclusions

While speaker verification systems have seen large gains in performance from *ad hoc* combination of several component systems, a unified framework for joint development of a combined system that ensures system diversity has been lacking. The component systems are trained in isolation to maximize individual performance rather than the overall system being trained to maximize combined performance. In this work, we presented a simple model for the system combination problem and found the performance of the combined system to be a function of the performance of the individual systems being combined and a "correlation coefficient" obtained from the average class-condition covariance of the vector of scores. Based on this result we presented a technique for taking into account the characteristics of the scores from a set of fixed existing systems during the development of a new SVM system in order to improve the combined system performance. This is realized through a modification of the SVM optimization problem via the introduction of a regularization term involving the covariance between the scores of the previously existing systems and the input features to the SVM, explicitly encouraging diversity of the resulting system ensemble. The trade-off between the individual performance of the SVM system and the inter-system average class-conditional covariance is reflected in the optimization through the introduction of the Lagrange multiplier λ. The technique can be implemented cheaply through the use of a simple kernel function, which we call *anticorrelation* kernel.

We show the effectiveness of the anticorrelation technique in a series of simulated experiments and in speaker verification experiments on the 2005 and 2006 NIST SRE tasks using four component systems: a standard UBM-GMM system, a cepstral supervector system, an MLLR-based system, and a prosodic system. We show results using the proposed kernel on all possible combinations involving two systems (two-way combinations) and on some combinations involving three and four systems. We demonstrate a performance gain of around 19% for a four-way combination using the anticorrelation kernel with respect to the performance of the combination obtained without anticorrelation. When the same four speaker verification systems are compensated for intersession variability, the gains from the anticorrelation method disappear. Our analysis indicates that the reason for this is that much of the correlation between the systems is, in fact, due to the intersession variability. Once systems are compensated for this variability, the remaining correlation is too hard to estimate robustly. The anticorrelation method can then be seen as a replacement for intersession variability compensation methods when the right databases are not available for the estimation of the matrices needed for those methods.

The fact that, in our experiments, the combination performance improves monotonically as λ grows and ρ decreases indicates that the optimal trade-off between the performance of the individual

system and the value of ρ probably occurs at a negative value of ρ. Since we are using $\lambda\sigma_{SB}^2$ as our regularization term, σ_{SB} and, with it, ρ are forced toward zero and negative values will be unlikely to occur in our setup. Using a linear term $\lambda\rho$ in the objective function would allow ρ to become negative, perhaps leading to better combination results. Solving this optimization problem, though, requires the use of general-purpose convex optimization software, which would be too slow for our purposes, or the development of a solver specifically designed for it. This is a direction we plan to explore.

We note that the anticorrelation technique is general in that it can be applied to any binary classification task for which more than one system can be trained and at least one of them is an SVM. Many modern machine learning problems have these characteristics, among them microarray gene expression classification problems (Brown et al., 2000), biometric tasks (Roy and Bhattacharya, 2005; Heisele et al., 2003), and a variety of other classification tasks (Sebastiani, 2002). The proposed method has the potential to lead to significant gains on some of those tasks and many others depending on the nature of the features used, their dimension, the number of samples available for training, the absolute performance of the systems, and so on. Finally, since the implementation of the proposed method simply reduces to the use of a specific kernel function, any statistical procedure that can be kernelized (of which SVMs are simply one example) could potentially benefit from it.

Acknowledgments

We thank Robert M. Gray and Robert Tibshirani for helpful comments and discussions. We also thank our colleagues at SRI's Speech Technology and Research Laboratory. In particular, we thank Sachin Kajarekar, Andreas Stolcke, and Nicolas Scheffer for providing some of the features and systems used in these experiments. We are also grateful to the anonymous reviewers who provided valuable feedback that greatly improved the paper. This research was funded by NSF CNS-0652510 at Stanford University and through a development contract with Sandia National Laboratories by the National Geospatial-Intelligence Agency (NGA) under National Technology Alliance (NTA) Agreement Number NMA 401-02-9-2001 at SRI International. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the NSF, NGA, the United States Government, or Rosettex.

Appendix A. Proof of Upper Bound on EER (Equation 4)

We wish to prove that if a set of scores F is distributed such that $F \mid Y = y \sim \mathcal{N}(\mu_y, \sigma_y^2)$, for $y = a, b$ (that is, the class-conditional distributions are Gaussian), the EER obtained when the class is estimated as in (1) is upper bounded by $\phi(-\frac{1}{2}\frac{\delta\mu}{\sigma})$, where $\delta\mu = \mu_b - \mu_a$, and σ satisfies $\sigma \geq (\sigma_a + \sigma_b)/2$.

For the case in which $\sigma_a = \sigma_b = \sigma$, the threshold t^* corresponding to EER is given by $t_s^* = (\mu_a + \mu_b)/2$ (s here stands for *same* variance) since this is the value that results in the rate of false acceptances ($e_{b|a}$) being equal to the rate of false rejections ($e_{a|b}$). Replacing this threshold in (2) or (3), we get that $\mathrm{EER}_s = \phi(-\frac{1}{2}\frac{\delta\mu}{\sigma})$. The upper plot in Figure 8 illustrates this case.

When variances are not equal, the EER threshold is no longer at half the distance between the two means. Nevertheless, we can find the approximate location of the threshold by mapping the value of t_s^* to two new locations, t_{db} and t_{da} (d stands for *different* variance) such that in one case the rate of false rejections is equal to EER_s and in the other case the rate of false acceptances is equal to EER_s. The EER threshold for the unequal variance case, t_d^*, will then be somewhere between these two points, since being to the left or to the right of both of them would result in the rates of false rejections and false acceptances being different from each other.

The values t_{da} and t_{db} are determined such that $e_{b|a}(t_{da}) = \text{EER}_s$ and $e_{a|b}(t_{db}) = \text{EER}_s$, respectively (with $e_{a|b}$ and $e_{b|a}$ defined in (2) and (3)), and they are given by

$$\begin{aligned} t_{db} &= \sigma_b/\sigma(t_s^* - \mu_b) + \mu_b \\ t_{da} &= \sigma_a/\sigma(t_s^* - \mu_a) + \mu_a. \end{aligned}$$

If $t_{da} \leq t_{db}$ then there will be a threshold t_d^* between t_{da} and t_{db} for which $e_{a|b}(t_d^*) = e_{b|a}(t_d^*) = \text{EER}_d \leq \text{EER}_s$. This is the case illustrated in Figure 8. So, if we can find some σ such that $t_{da} \leq t_{db}$, we can replace this value in $\text{EER}_s = \phi(-\frac{1}{2}\frac{\delta\mu}{\sigma})$ to get the desired upper bound. Now,

$$t_{da} - t_{db} = \frac{\sigma_a}{\sigma}(t_s^* - \mu_a) + \mu_a - \frac{\sigma_b}{\sigma}(t_s^* - \mu_b) - \mu_b = \delta\mu\left(\frac{\sigma_b + \sigma_a}{2\sigma} - 1\right).$$

Since we are assuming that $\delta\mu > 0$, we see that $t_{da} \leq t_{db}$ if and only if $\sigma \geq (\sigma_a + \sigma_b)/2$. When $\sigma = (\sigma_a + \sigma_b)/2$ we get $t_{da} = t_{db}$, which implies that $\text{EER}_d = \text{EER}_s$. Hence, we can always compute EER_d as $\phi(-\frac{\delta\mu}{\sigma_a+\sigma_b})$. It is easy to prove that $\sigma = \sqrt{(\sigma_a^2 + \sigma_b^2)/2}$ (the square root of the average variance instead of the average of the standard deviations) also satisfies $\sigma \geq (\sigma_a + \sigma_b)/2$.

References

T. Anderson and R. Bahadur. Classification into two multivariate normal distributions with different covariance matrices. *The Annals of Mathematical Statistics*, 33(2):420–431, June 1962.

L. Breiman. Bagging predictors. In *Machine Learning*, pages 123–140, 1996.

G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: A survey and categorisation. *Journal of Information Fusion*, 6:5–20, 2005a.

G. Brown, J. L. Wyatt, and P. Tiño. Managing diversity in regression ensembles. *Journal of Machine Learning Research*, 6:1621–1650, 2005b.

M. Brown, W. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. Furey, M. Ares Jr., and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. In *Proceedings of the National Academy of Sciences*, volume 97, pages 262–267, 2000.

N. Brummer, L. Burget, J. Cernocky, O. Glembek, F. Grezl, M. Karafiat, D. van Leeuwen, P. Matejka, P. Schwarz, and A. Strasheim. Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006. *IEEE Transactions on Audio, Speech and Language Processing*, 15(7):2072–2084, September 2007.

N. Brummer and J. du Preez. Application independent evaluation of speaker detection. *Computer Speech and Language*, 2006.

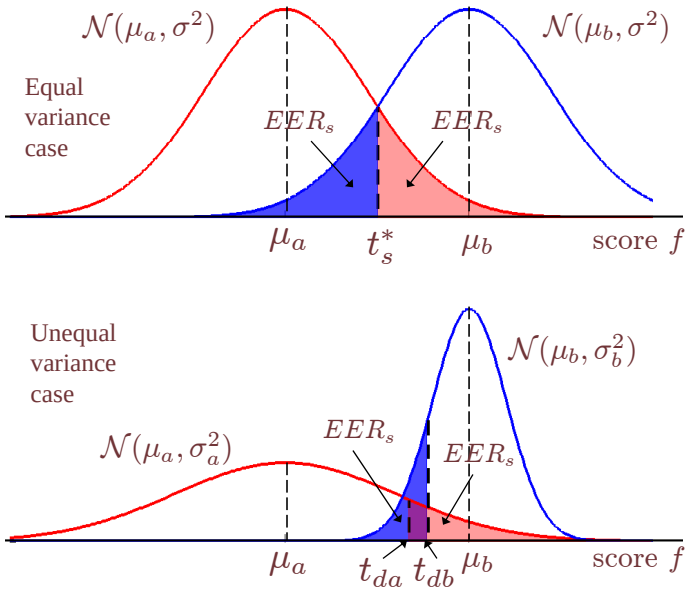

Figure 8: Proof of (4). The upper figure illustrates the computation of EER_s, the EER for the equal variance case. In the lower figure, the unequal variance case is considered. The x-axis values t_{da} and t_{db} are computed such that the area of the two shaded regions coincides with the area of the respective shaded regions in the upper plot. All four shaded areas are then equal to EER_s. The figure illustrates the case in which $t_{da} \leq t_{db}$. In this case, there will be a threshold t_d^* somewhere between t_{da} and t_{db} which corresponds to EER_d, the EER for the unequal variance distributions. From this, we conclude that, if $t_{da} \leq t_{db}$, then $\text{EER}_d \leq \text{EER}_s$.

W. Campbell. Compensating for mismatch in high-level speaker recognition. In *Proceedings of the Speaker and Language Recognition Workshop, Odyssey 2006*, Puerto Rico, USA, June 2006.

W. Campbell, D. Sturim, and D. Reynolds. Support vector machines using GMM supervectors for speaker verification. *IEEE Signal Processing Letters*, 13(5):308–311, May 2006.

N. Dehak, P. Kenny, and P. Dumouchel. Continuous prosodic features and formant modeling with joint factor analysis for speaker verification. In *Proceedings of the 10th European Conference on Speech Communication and Technology (Interspeech 07)*, Antwerp, August 2007.

L. Ferrer, M. Graciarena, A. Zymnis, and E. Shriberg. System combination using auxiliary information for speaker verification. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, April 2008a.

L. Ferrer, E. Shriberg, S. Kajarekar, and K. Sönmez. Parameterization of prosodic feature distributions for SVM modeling in speaker recognition. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Honolulu, April 2007.

L. Ferrer, E. Shriberg, S. Kajarekar, A. Stolcke, K. Sönmez, A. Venkataraman, and H. Bratt. The contribution of cepstral and stylistic features to SRI's 2005 NIST speaker recognition evaluation system. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 101–104, Toulouse, May 2006.

L. Ferrer, Kemal Sönmez, and E. Shriberg. An anticorrelation kernel for improved system combination in speaker verification. In *Proceedings of the Speaker and Language Recognition Workshop, Odyssey 2008*, Stellenbosch, South Africa, January 2008b.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

G. Fumera and F. Roli. A theoretical and experimental analysis of linear combiners for multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2001.

B. Heisele, P. Ho, J. Wu, and T. Poggio. Face recognition: Component-based versus global approaches. In *Computer Vision and Image Understanding*, volume 91(1), pages 6–12. Elsevier, 2003.

F. Huenupán, N. B. Yoma, C. Molina, and C. Garretón. Speaker verification with multiple classifier fusion using Bayes based confidence measure. In *Proceedings of the 10th European Conference on Speech Communication and Technology (Interspeech 07)*, Antwerp, August 2007.

P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel. Joint factor analysis versus eigenchannels in speaker recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4): 1435–1447, May 2007.

P. Kenny and P. Dumouchel. Experiments in speaker verification using factor analysis likelihood ratios. In *Proceedings of the Speaker and Language Recognition Workshop, Odyssey 2004*, Toledo, Spain, May 2004.

A. Kocsor, K. Kovács, and C. Szepesvári. Margin maximizing discriminant analysis. In *Proceedings of the 15th European Conference on Machine Learning*, Pisa, September 2004.

A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems*, volume 7, pages 231–238. MIT Press, 1995.

L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, 2004.

G. R. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

Y. Liu. *Negative Correlation Learning and Evolutionary Design of Neural Network Ensembles*. PhD thesis, University of New South Wales, New South Wales, Australia, 1999.

D. Matrouf, N. Scheffer, B. Fauve, and J.F. Bonastre. A straightforward and efficient implementation of the factor analysis model for speaker verification. In *Proceedings of the 10th European Conference on Speech Communication and Technology (Interspeech 07)*, Antwerp, August 2007.

N. C. Oza and K. Tumer. Input decimation ensembles: Decorrelation through dimensionality reduction. In *Lecture Notes in Computer Science*, volume 2096, pages 238–247. Springer, 2001.

P. Pavlidis, J. Cai, J. Weston, and W. S. Noble. Learning gene functional classifications from multiple data types. *Journal of Computational Biology*, 9:401–411, 2002.

D. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adami, Q. Jin, D. Klusacek, J. Abramson, R. Mihaescu, J. Godfrey, D. Jones, and B. Xiang. The SuperSID project: Exploiting high-level information for high-accuracy speaker recognition. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 4, pages 784–787, Hong Kong, April 2003.

D. Reynolds, W. Campbell, T. Gleason, C. Quillen, D. Sturim, and P. Torres-Carrasquillo. The 2004 MIT Lincoln Laboratory speaker recognition system. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Philadelphia, March 2005.

B. E. Rosen. Ensemble learning using decorrelated neural networks. *Special Issue on Combining Artificial Neural Networks: Ensemble Approaches*, 8:373–384, 1996.

K. Roy and P. Bhattacharya. Iris recognition with support vector machines. In *Advances in Biometrics*, volume 3832 of *Lecture Notes in Computer Science*, pages 486–492. Springer, 2005.

F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1): 1–47, 2002.

E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, and A. Stolcke. Modeling prosodic feature sequences for speaker recognition. *Speech Communication*, 46(3-4):455–472, 2005. Special Issue on Quantitative Prosody Modelling for Natural Speech Description and Generation.

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

A. Solomonoff, C. Quillen, and W. Campbell. Channel compensation for SVM speaker recognition. In *Proceedings of the Speaker and Language Recognition Workshop, Odyssey 2004*, Toledo, Spain, May 2004.

A. Stolcke, L. Ferrer, and S. Kajarekar. Improvements in MLLR-transform-based speaker recognition. In *Proceedings of the Speaker and Language Recognition Workshop, Odyssey 2006*, Puerto Rico, USA, June 2006.

A. Stolcke, S. S. Kajarekar, L. Ferrer, and E. Shriberg. Speaker recognition with session variability normalization based on MLLR adaptation transforms. *IEEE Transactions on Audio, Speech, and Language Processing*, September 2007.

K. Tumer and J. Ghosh. Analysis of decision boundaries in linearly combined neural classifiers. *Pattern Recognition*, 29:341–348, 1996.

N. Ueda and R. Nakano. Generalization error of ensemble estimators. In *Proceedings of International Conference on Neural Networks*, pages 90–95, 1996.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1999.

M. Zanda, G. Brown, G. Fumera, and F. Roli. Ensemble learning in linearly combined classifiers via negative correlation. In *Proceedings of the International Workshop on Multiple Classifier Systems*, Prague, May 2007.