An Anticorrelation Kernel for Improved System Combination in Speaker Verification

Luciana Ferrer[1,2] *Kemal Sönmez*[2] *Elizabeth Shriberg*[2]

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

Abstract

This paper presents a method for training SVM-based classification systems for combination with other existing classification systems designed for the same task. Ideally, a new system should be designed such that, when combined with the existing systems, the resulting performance is optimized. To achieve this goal, we include a regularization term in the SVM objective function that aims to reduce the within-class correlation between the resulting scores and the scores produced by one of the existing systems, introducing a trade-off between such correlation and the system's individual performance. That is, the SVM system "takes one for the team", falling somewhat short of its best possible performance in order to be more complementary to the existing system. We report results on the NIST 2005 and 2006 speaker recognition evaluations (SRE) using three component systems: a standard UBM-GMM system, an MLLR-based system, and a prosodic system, and show that the proposed technique results in performance gains of 16% in EER and 23% in DCF.

1. Introduction

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 [1, 2], 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 [3]. Most of these systems perform the combination of information sources at the score level ([4, 1, 2, 5, 6]): systems that model each type of feature using a certain model are independently developed and their scores are combined in the last stage 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 the combination is not guaranteed to give strictly better performance than every system 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 individual classifiers that are highly uncorrelated. This way, all classifiers contribute independent and therefore, hopefully, complementary information leading to a better final decision.

In this work, we consider the case of two available classifiers, one of them fixed and given (we can consider this system as a black box that simply produces a score value for each sample) and the other one a support vector machine (SVM) that we need to train. Our main goal is to modify the training criteria for the SVM so that the scores resulting from this system are as minimally correlated as possible with the scores from the black box system. The resulting systems should produce better combination performance than we would get if we trained the SVM independently to maximize the (soft) margin.

First, we motivate the approach by demonstrating the notion of anticorrelation optimization on artificially generated data. Then, we state the problem formally and develop the convex optimization problem that leads to the anticorrelation kernel. Finally, we demonstrate results on artificial data and on NIST 2005 and 2006 SRE tasks using combinations of three component systems and discuss some alternative approaches.

2. Anticorrelation Kernel

The problem described above can be formally stated as follows. Consider a binary classification task with classes $y \in \{+1, -1\}$, for which two separate classifiers B and S are available. We will restrict the system S to correspond to an SVM. System B, on the other hand, can be any classifier that produces a score for each sample. We will consider this system to be a black box from which we only have the scores that it produces.

The final classification decision will be made based on a combination of the outputs generated by both classifiers. Our goal is to work on system S, in order to improve the combination performance over the one obtained when system S is designed with no knowledge of system B.

2.1. Motivation

We base our strategy on the following intuition: Two systems that produce highly correlated scores cannot lead to a combination performance that is significantly better than that of the individual systems. Hence, we would like to train system S so that the correlation between the resulting scores and those from system B is small. But there is a problem with this idea: the correlation between both scores grows when the distance between the two classes is larger and this distance is exactly what we would like to maximize to achieve good performance. Hence, minimizing the correlation directly forces the performance of the system to degrade. On the other hand, if instead of minimizing the overall correlation of the systems we minimize the within-class correlation, we can reduce the correlation of the systems without *directly* affecting the distance between the classes (in practice, as we will see, reducing the correlation of the systems will of course have some cost on the performance of at least one of them).

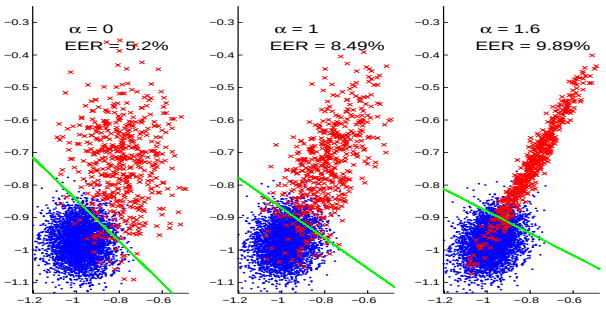

Figure 1: Scatter plot of the scores for two systems, for varying values of the within-class correlations, along with the EER% achieved when optimally combining the two scores with a linear classifier. Correlation for class y is given by $\alpha\rho_y$, where ρ_y is the correlation between the MLLR and the SNERF systems for class y. The green lines show the decision boundary for EER.

In [7], Tumer and Ghosh formalize this intuition by deriving an expression for the added error (the error beyond the Bayesian error) obtained when combining classifiers with a simple average operation under several, rather strong, assumptions (among them, that the output of the systems is monotonous around the decision boundaries, that errors from a certain classifier for the different classes are independent and identically distributed, and that all systems have the same error distribution). In the expression, the added error of the combined system equals that of the individual systems when the average within-class correlation between the errors committed by the systems is 1. On the other hand, if the average correlation is 0, the added error of the individual systems is reduced by 1/N, where N is the number of systems being combined. In the general case, the added error grows linearly with the average within-class correlation.

Figure 1 gives an example of the ideas mentioned above. The data in the plots were created assuming that each class corresponds to a jointly Gaussian distribution (red markers correspond to +1, blue markers correspond to -1). The mean, variance, and correlation between the systems for each class were estimated from real data (specifically, from the MLLR and SNERF systems, which will be introduced in Section 4.1). Then, artificial data were generated according to that model, with the correlation for class y given by $\alpha\rho_y$, where ρ_y is the correlation between the two systems for class y estimated from the data, and α takes values 0, 1, and $1/\max(\rho_{+1}, \rho_{-1}) - 0.05$ for the left, center, and right plots respectively. For each value of α, the optimal linear classifier (in the mean squared error sense) is indicated by the green line, and the resulting equal error rate (EER) is given. Since the marginal means and variances are kept unchanged for all plots, the individual system performance is also unchanged. We can then clearly see that by changing only the within-class correlation of the two systems, we can obtain large improvements in performance.

Evidently, the example given above is not realizable in practice. To reduce the correlation between the systems, at least one of the systems will have to be affected, most probably resulting in changed within-class means and variances. Since the means and variances obtained without trying to minimize the correlation are probably the ones that optimize the individual performance of the system, we would expect that intervening in the training of a system in order to minimize the correlation between that system and some other one will degrade its performance. The hope is that the degradation of the performance of the individual system will have a smaller effect on the performance of the combination than the reduction of the correlation, resulting in an overall performance gain.

2.2. Support Vector Machines

Consider a training set with m samples, $T = \{(x_i, y_i) \in \mathcal{R}^d \times \{-1, +1\}; i = 1, ..., m\}$, where x_i are the features and y_i the class corresponding to sample i. The goal is to find a function $f(x) = w^t x + b$, such that sign$(f(x))$ is the predicted class for feature vector x. The standard (primal) support vector machine (SVM) formulation for classification is given by [8]:

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

Minimizing the norm of the weight vector is equivalent to maximizing the margin between the samples and the hyperplane. The *slack* variables ϵ_i allow for some samples to be at a distance smaller than the margin 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 are only required to be able to compute the inner products between the transforms for each pair of samples, *i.e.*, we only need to compute $K(x_i, x_j) = \phi(x_i)^t \phi(x_j)$ for each $i, j = 1, \ldots, m$. This fact is what allows for complex transforms of the input features to be used, as long as the kernel function $K(x_i, x_j)$ can be easily computed. As we will see, one way of implementing the method proposed in this paper is by using the kernel trick.

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 [8, 9] 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 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.

2.3. Modified Support Vector Machines

Based on the ideas explained in Section 2.1, we would like to modify the SVM problem by adding a term $\lambda\rho^2$ in the objective function in (1), where λ is a tunable parameter and ρ is the within-class correlation between system S and system B. Given the scores $\{b_i; i = 1, ..., m\}$ from system B for the training set T, the within-class correlation between the scores produced by the SVM and these scores is given by

$$\rho = \frac{\text{cov}(B, S|Y)}{\sqrt{\text{var}(B|Y)\text{var}(S|Y)}} \quad (2)$$

where cov$(B, S|Y)$ is the within-class covariance between the scores for systems B and S, and var$(B|Y)$ and var$(S|Y)$ the

within-class variances.[1] These can be approximated by the within-class sample covariance and variances in the training set T. The within-class sample or empirical covariance cov_e can be calculated as

$$\text{cov}_e(B, S|Y) = \frac{1}{m}\sum_{i=1}^{m}(b_i - \bar{b}_{y_i})(s_i - \bar{s}_{y_i}) \tag{3}$$

where $\bar{b}_y$ and $\bar{s}_y$ are the sample means for each set of scores for class y. The value s_i is the output of the SVM, *i.e.*, $s_i = w^t x_i + b$. Replacing this into (3) we get

$$\text{cov}_e(B, S|Y) = w^t K \tag{4}$$

where

$$K = \frac{1}{m}\sum_{i=1}^{m}(b_i - \bar{b}_y)(x_i - \bar{x}_y) \tag{5}$$

where $\bar{x}_y$ is the vector of feature means for class y. K is simply the vector of within-class (sample) covariances between each input feature and the scores from system B. Similarly, we obtain $\text{var}_e(S|Y) = w^t M w$ where M is the within-class sample covariance matrix of the training set T. Calling $v = \text{var}_e(B|Y)$, the sample variance of the scores from system B, we can write ρ_e^2 (the empirical correlation) as

$$\rho_e^2 = \frac{w^t K K^t w}{v\ w^t M w} \tag{6}$$

Using the empirical correlation instead of the actual one, we can write the new objective function as $J(w, \epsilon) = \frac{1}{2}w^t w + \frac{1}{2}\lambda\frac{w^t K K^t w}{v\ w^t M w} + C\sum_i \epsilon_i$. This function is not convex which makes the problem much harder to solve. On the other hand, if instead of trying to minimize the correlation we try to minimize the covariance, we get

$$J(w, \epsilon) = \frac{1}{2}w^t w + \frac{\lambda}{2}w^t K K^t w + C\sum_i \epsilon_i \tag{7}$$

$$= \frac{1}{2}w^t A w + C\sum_i \epsilon_i \tag{8}$$

where $A = I + \lambda K K^t$ is a symmetric positive semidefinite matrix. The optimization problem obtained when using this objective function can be solved exactly since it is convex. Furthermore, a covariance equal to zero implies a correlation equal to zero. If by minimizing the square covariance we can force it to be zero, we have in fact achieved the minimization of the square correlation. In Section 4.4 we show that minimizing the within-class covariance in practice results in the within-class correlation being reduced (although not to zero). These facts justify the use of the covariance instead of the correlation in the objective function.

By doing a change of variable $\tilde{w} = Bw$, with $A = B^t B$ (*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), we can write the new optimization problem as

$$\begin{array}{lll} \text{minimize} & J(\tilde{w}, \epsilon) = \frac{1}{2}\tilde{w}^t\tilde{w} + C\sum_i \epsilon_i & \\ \text{subject to} & y_i(\tilde{w}^t z_i + b) \geq 1 - \epsilon_i & i = 0, ..., m \\ & \epsilon_i \geq 0 & i = 0, ..., m \end{array} \tag{9}$$

where

$$z_i = B^{-1}x_i \tag{10}$$

Since matrix A has a very particular structure, we can find an expression for its inverse using the matrix inversion lemma, by which $A^{-1} = I - \frac{\lambda}{1+\lambda K^t K}K K^t$. Hence, one way of implementing the proposed method is to define a kernel $K(x, y) = x^t B^{-1}(B^{-1})^t y = x^t A^{-1} y$ to be used by the SVM. It can be easily shown that 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

$$K(x, y) = x^t y - \frac{\lambda}{1 + \lambda K^t K}x^t K y^t K \tag{11}$$

The computation of this kernel requires only the calculation of three inner products, which makes the method computationally feasible. Another approach is to directly transform the features using (10). 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 prove that

$$B^{-1} = I - \frac{\alpha}{K^t K}K K^t \tag{12}$$

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 (10). This means that we can implement the proposed method by transforming the input features using the following expression:

$$z_i = x_i - \alpha\frac{K^t x_i}{K^t K}K \tag{13}$$

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. Hence, in our experiments, the kernel implementation is used.

2.4. 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 within-class covariance between the scores from system B and the scores from system S is given by Equation (4). For a fixed value of $\|w\| = v$, the w that maximizes the absolute value of $w^t K$ is given by $w = vK/\|K\|$. Hence, K gives the direction for the vector w for which the within-class covariance between the two systems is maximum. A w orthogonal to K would result in zero within-class covariance between the two systems. Our goal is then 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

[1] 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.

balance can be achieved by tuning the parameter λ. Given this interpretation for K we can see that 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 that would result in the maximum within-class covariance between the two systems.

We can interpret the kernel given by (11) 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 and y into the vector K from the linear kernel. The resulting value of the kernel will be small if x and y are both aligned with K. Since the SVM will make an effort to separate only points that give a high kernel value, this means 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 within-class covariance between the systems.

Finally, we note that if we choose to transform the features instead of using the kernel trick, the resulting features have a very simple interpretation. When $\lambda = \infty$, Equation (13) becomes $z_i = x_i - \frac{K^t x_i}{K^t K} K$, which is the expression for subtracting the component on K of x_i. If λ is not ∞ then only a part of the component is subtracted.

Note the similarity of expression (13) with the nuisance attribute projection (NAP) formula [10]. In both cases, we wish to eliminate certain directions from the feature vectors. In general, NAP has been used to eliminate directions estimated to have information superfluous to the task of speaker verification. Instead, in the anticorrelation procedure, a single direction is eliminated: the one that maximizes the correlation between the two systems being combined.

2.5. Possible Extensions

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. In this case, when $\lambda = \infty$, z_i is simply the projection of x_i on the complementary space to that spanned by the correlation vectors $K_1, \ldots, K_N$, between the features used in S and each of the systems $B_1, \ldots, B_N$. This method can also be used in cases in which the two classes cannot be assumed to have the same covariance structure. In this case, we could compute two vectors K_{+1} and K_{-1} corresponding to the correlation of system B with the features from system S for each class[2]. It is then possible to anticorrelate with both directions K_{+1} and K_{-1}, instead of with the overall K, potentially using different values of λ for both regularization terms. In this paper we focus on the simplest version where we wish to anticorrelate with a single system B, as described above. Furthermore, since we are applying the method to a speaker verification task where only a very limited number of samples of the positive class is available, we only estimate K_{-1} and use $K = K_{-1}$. Other tasks, though, might benefit from the use of the generalized method just described.

3. Experiments on Artificial Data

To test the proposed kernel on a simple task, we generated data for two classes with model $x = C\tilde{x} + m_y$, where the $\tilde{x}_i$ are generated independently with a normal distribution with zero mean and unit variance, C is a random matrix intended to create correlation between the features, and m_y is a vector of zeros for one class and a vector of ones for the other class. We then took half of the features and trained an SVM, which served as system B. The remaining features were used to train system S with varying values of λ. We created two separate sets, one for training and one for testing, each of them with 5,000 negative samples and 1,000 positive samples. The combination is performed using another SVM that is trained on the training set with the scores from the two SVM systems, B and S, for each value of λ.

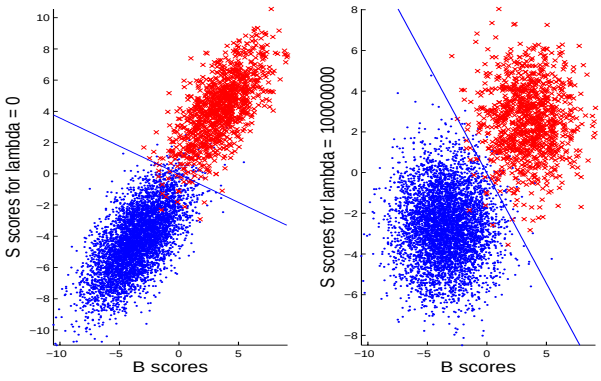

Figure 2: Scores from system B versus scores from system S for two values of λ

Figure 2 shows the scatter plot of scores (on the training data) for both systems with $\lambda = 0$ and $\lambda = 10^7$. We can see that for the large value of λ, the within-class covariances (and, hence, correlations) have been reduced to 0. We can also see that the separation of the two classes is better for the larger λ, which implies that the performance of the combination should be better in this case.

Figure 3 confirms this observation. In this figure we see the error rates for system S, system B and the combined system, and the correlation between system S and system B, as a function of the value of λ for the test data. The error for system B does not depend on λ. The error for system S grows with the value of λ since we are trading off poorer performance in exchange for lower correlation with system B. The figure shows that, in fact, we achieve lower correlations (reaching a value of zero) for higher values of λ. Finally, we see that the combination performance dramatically improves for higher values of λ, from 2.02% to 1.32%. This is a 35% relative improvement. Similar results were found by changing the random seed, the number of features, and the number of training and test samples.

4. 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, 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 the combination involving all three systems (three-way combination).

[2]Note that vector K, as defined in (5) can be obtained from those two vectors as $K = m_{+1}/mK_{+1} + m_{-1}/mK_{-1}$

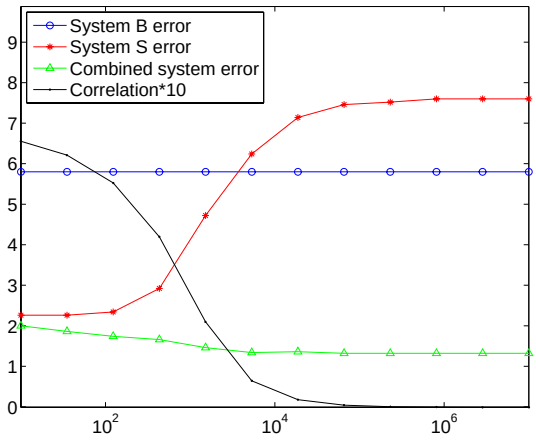

Figure 3: Error and correlation as a function of λ for an artificial problem

4.1. Individual System Descriptions

A brief description for each of the speaker verification systems used in the experiments follows.

UBM-GMM system: This system is a Gaussian mixture model (GMM) with universal background model (UBM), based on 13 MFCCs (without C_0) and first-, second-, and third-order difference features. The features are modeled by 2048 mixture components. For details, see [11].

MLLR-SVM system: The MLLR-SVM system [11, 12] uses the speaker adaptation transforms used in the speech recognition system as features for speaker verification. A total of four affine 39x40 transforms are 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 by SVMs trained to perform regression on the class labels using a linear kernel or the anticorrelation kernel proposed here.

SNERF system: This system models syllable-based prosodic features [13]. 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 [14]. 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 MLLR features.

4.2. 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 [1, 2, 3], 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 (5) 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 has to 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.

The performance measures used in this paper are the equal error rate (EER), and NIST's detection cost function (DCF), which 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. In this paper, DCF always refers to the minimum DCF achievable by the system on the data were the results are presented.

4.3. Databases

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 are taken from 2003 and 2004 NIST evaluations along with some FISHER data, resulting in a total of 2122 conversation sides. The SRE2005 task contains 26,270 trials, and the SRE2006 task contains 24,013 trials. The data used to obtain the correlation statistics (as described above) were drawn from data from the 2004 evaluation and comprised 2627 conversation sides.

4.4. Results

Table 1 shows the results on SRE05 and SRE06 data for the individual systems and two- and three-way combinations. Each block in this table corresponds to results obtained with the same set of systems with and without applying the anticorrelation kernel proposed here. In all cases the 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 experiments with the systems from this table. For the two SVM systems (MLLR and SNERFs) we show results obtained by training the target SVMs using the kernel in (11) with K computed using the scores corresponding to each of the other two systems. For example, M_G corresponds to a system that uses the MLLR fea-

tures and anticorrelation kernel with K given by the vector of covariances between the MLLR features and the scores from the GMM-UBM system. 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 M_S (MLLR features using anticorrelation kernel with respect to the SNERF system) for SRE06. 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, allows for other more robust directions to be used.

The next three blocks of results in Table 1 show the two-way combinations. The combiner used in all cases is a perceptron trained on SRE05 data. We can see that every time a combination is done between systems B and S_B, the performance is better than that for the combination of B and S. That is, applying the anticorrelation kernel to system S always gives a gain in the combination performance, even though in most cases system S_B has worse individual performance than system S. Furthermore, note that the 2-way combinations involving the SNERF system achieve results as good as or better than the combination of the two other systems, even though the performance of the SNERF system is approximately twice as bad as either of those two systems. This behavior can be predicted from the fact that the SNERF system is originally much less correlated to the MLLR and the GMM-UBM systems than those two systems are to each other.

Finally, the last block in the table shows two three-way combination results. The first one combines the three systems used in this paper, without using the proposed method. The second one combines the GMM with both the MLLR and SNERF systems with an anticorrelation kernel with respect to the GMM-UBM system. The performance gain here is larger than for any two-way combination. For SRE06, the gain is 16% for EER and 23% for DCF.

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 this might be a consequence of a better statistical match between SRE04 data (used to compute the K vectors) and SRE06 data than between those data and SRE05 data.

The last column in Table 1 shows the within-class 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 drastically reduced when the proposed method is used, even though it does not reach a zero value as we observed in the simulated experiments. This could mean that the amount of data used for the computation of K (2627 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.

Figure 4 shows a plot of false rejection versus false acceptance rates (obtained by varying the decision threshold) for the three systems without anticorrelation and the two three-way combinations from Table 1. We can see a uniform gain for all operating points in the curve when using the MLLR and SNERF

System	SRE05 EER / DCF	SRE06 EER / DCF	SRE06 CorI / CorT
G	7.52 / 0.306	6.58 / 0.336	-
M	7.93 / 0.291	7.07 / 0.305	-
M_G	8.86 / 0.322	7.82 / 0.307	-
M_S	7.89 / 0.296	6.36 / 0.284	-
S	14.97 / 0.564	14.35 / 0.617	-
S_G	16.14 / 0.594	15.64 / 0.624	-
S_M	17.52 / 0.646	16.02 / 0.631	-
G + M	6.43 / 0.239	5.39 / 0.267	0.56 / 0.80
G + M_G	5.83 / 0.208	5.07 / 0.230	0.27 / 0.74
G + S	5.74 / 0.233	5.61 / 0.290	0.22 / 0.47
G + S_G	5.70 / 0.230	5.02 / 0.264	0.10 / 0.44
M + S	6.63 / 0.252	6.09 / 0.280	0.30 / 0.58
M + S_M	6.59 / 0.248	5.50 / 0.255	0.11 / 0.50
M_S + S	6.43 / 0.236	5.39 / 0.249	0.19 / 0.53
G + M + S	5.54 / 0.203	5.29 / 0.253	-
G + S_G + M_G	4.94 / 0.177	4.42 / 0.195	-

Table 1: Performance results for individual systems and their combination. G = GMM-UBM system, M = MLLR system, S = SNERF system. The subindex corresponds to the system whose scores are used to compute the anticorrelation kernel. The last column shows the within-class correlations between the pair of systems being combined, for the impostor (CorI) and the target (CorT) samples.

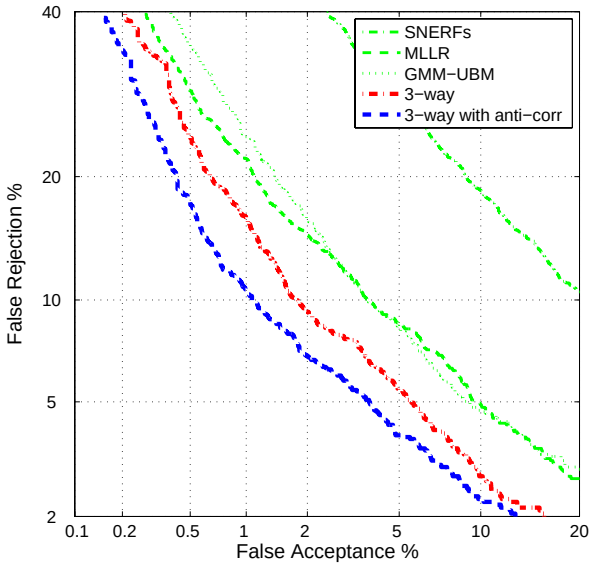

Figure 4: False rejection versus false acceptances for the individual systems and the 3-way combinations corresponding to the last two lines of Table 1.

systems with anticorrelation with respect to the GMM system.

When both systems being combined are SVM systems, the method presented in this paper can be compared with a simple feature-level combination where the feature vectors from both systems are concatenated and an SVM is trained with the resulting vectors. A further refinement of this consists on weight-

ing the vector components, assigning weight w to the features from one of the original systems and weight $1 - w$ to the other features. This allows us to compensate for different lengths in the original vectors and/or to bias the training procedure to make more use of the features from the better-performing system. We tried this for the MLLR/SNERF pair and tuned the weights given to each system on SRE05 data. The best result (DCF=0.286 and EER=6.25%) was obtained by giving a large weight (0.95) to the MLLR features. This result is in fact worse than the score-level combination of line $M + S$ in Table 1. Furthermore, 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.

Finally, another method that needs to be considered is one 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. This method results in a performance equal to that of system B alone. This is easily understood if we consider the speaker verification setup. Since only a very limited set of samples is available to train the speaker model, the same samples have to be used to train both models. Furthermore, in our experiments, the same negative samples are used to train both systems. This results in system B producing highly optimistic scores on these samples. When these scores are used as an extra feature for system S, the weight corresponding to it is likely to take value 1 (since that feature alone can perfectly classify the samples) while all other features are ignored. This results in system S producing scores that are identical to the scores produced by B.

5. 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 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 have presented a technique for taking into account the characteristics of the scores from a fixed existing system during the development of a complementary SVM system in order to maximize the combined system performance. This is realized through a modification of the SVM optimization via the introduction of a regularization term that is informed by the covariance characteristics between the existing fixed detection system and the input features to the SVM. The trade-off between the individual performance of the SVM system and the inter-system within-class correlation is reflected in the optimization through the introduction of the Lagrange multiplier λ. However, the technique does not seem to require tuning since λ can be taken effectively as infinity and safely considered optimal in all our experiments.

We have shown the effectiveness of the anticorrelation technique in speaker verification experiments on the 2005 and 2006 NIST SRE tasks using three component systems: a standard UBM-GMM 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 the combination involving all three systems (three-way combination). We demonstrate performance gains of 16% in EER and 23% in DCF for the three-way combination.

Finally, we note that the anticorrelation technique is general in that it can be applied to any multi-system statistical detection task in which one or more of the subsystems is trained using SVMs. The technique is applicable to combinations of many systems through progressive application of the anticorrelation kernel to the different systems or using the method mentioned in Section 2.4. Furthermore, 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.

6. Acknowledgments

We thank our colleagues at SRI's Speech Technology and Research Laboratory for useful discussions. This research was funded through a development contract with Sandia National Laboratories and by the National Geospatial-Intelligence Agency (NGA) under National Technology Alliance (NTA) Agreement Number NMA 401-02-9-2001. 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 NGA, the United States Government, or Rosettex.

7. References

[1] 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 *Proc. ICASSP*, Toulouse, May 2006, vol. 1, pp. 101–104.

[2] 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*, vol. 15, no. 7, pp. 2072–2084, Sept. 2007.

[3] D. Reynolds, W. Campbell, T. Gleason, C. Quillen, D. Sturim, and P. Torres-Carrasquillo, "The 2004 MIT Lincoln Laboratory speaker recognition system," in *Proc. ICASSP*, Philadelphia, Mar. 2005.

[4] 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 *Proc. ICASSP*, Hong Kong, Apr. 2003, vol. 4, pp. 784–787.

[5] F. Huenupán, N. B. Yoma, C. Molina, and C. Garretón, "Speaker verification with multiple classifier fusion using Bayes based confidence measure," in *Proc. Interspeech*, Antwerp, Aug. 2007.

[6] N. Dehak, P. Kenny, and P. Dumouchel, "Continuous prosodic features and formant modeling with joint factor analysis for speaker verification," in *Proc. Interspeech*, Antwerp, Aug. 2007.

[7] K. Tumer and J. Ghosh, "Error correlation and error reduction in ensemble classifiers," *Connection Science*, vol. 8, no. 3-4, pp. 385–403, 1996.

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

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

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

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

[12] A. Stolcke, L. Ferrer, and S. Kajarekar, "Improvements in MLLR-transform-based speaker recognition," in *Proc. Odyssey-06*, Puerto Rico, USA, June 2006.

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

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