SVM Speaker Verification using an Incomplete Cholesky Decomposition Sequence Kernel

*Jérôme Louradour *, Khalid Daoudi * and Francis Bach ***

*IRIT, CNRS UMR 5505, Université Paul Sabatier, Toulouse, France
**Centre de Morphologie Mathématique, Ecole des Mines de Paris, Fontainebleau, France
{louradou,daoudi}@irit.fr francis.bach@mines.org

Abstract

The Generalized Linear discriminant Sequence (GLDS) kernel has been showing to provide very good performance in SVM speaker verification in NIST SRE evaluations. The GLDS kernel is based on an explicit mapping of each sequence to a single vector in a feature space using polynomial expansions. Because of practical limitations, these expansions have to be of degree less or equal to 3. In this paper, we generalize the GLDS kernel to allow not only any polynomial degree but also any expansion (possibly infinite dimensional) that defines a Mercer kernel (such as the RBF kernel). To do so, we use low-rank decompositions of the Gram matrix to express the feature space kernel in terms of input space data only. We present experiments on the Biosecure project data. The results show that our new sequence kernel outperforms the GLDS one as well as the one developed in our recent work.

1. Introduction

The technique of Support Vector Machines (SVM) is an interesting alternative to Gaussian Mixture Models (GMMs) for speaker verification systems using acoustic features, as they are well suited to separate complex regions in binary classification problems, through an optimal nonlinear decision boundary. A challenge however in applying SVM to monitor conversations in a communication network, such as in NIST SRE evaluations, is to deal with the huge amount of data available. Thus, in order to exploit a rich database involving various types of low quality cell phones with a SVM training algorithm, the frame-based approach such as the one in [1] needs to be adapted to make a tractable training and testing procedure. A solution could be to use clustering methods to reduce the size of the training corpus as was done in [2]. On the other hand, the problem in speaker verification is to classify sequences of vectors. It is then more natural to conceive kernels that measure similarity between sequences and use them in a SVM architecture.

The use of sequence kernels in SVM speaker verification has gained considerable attention in recent years. In [3, 4, 5] for instance, sequence kernels based on generative probabilistic models have been used. However, the sequence kernel that has shown the best results so far in NIST SRE evaluations is the GLDS kernel [6]. The latter consists basically of an explicit mapping of each sequence to a single vector in a feature space using polynomial expansions. Then, a SVM with a linear kernel is used in this feature space.

The GLDS kernel has however both practical and theoretical limitations. The former is due to the fact that only polynomial expansions of degree less or equal to 3 can be used in practice. The latter is due to the fact that it does not (readily) generalize to infinite expansions such as the radial basis one. The purpose of this paper is to overcome these two limitations. We first start by defining a class of sequence kernels by allowing the expansion in GLDS to be any expansion that defines a Mercer kernel. We then provide a finite-dimensional form to the sequence kernels defined this way. This form can still be intractable in speaker verification applications. We then use low rank matrix decompositions to achieve tractable sequence kernels.

2. Overview of the GLDS kernel

The original form of the GLDS kernel [6] involves a polynomial expansion ϕ_p, with monomials (between each combination of vector components) up to a given degree p. For example, if $p = 2$ and $\mathrm{x} = [x_1, x_2]^\top$ is a 2-dimensional input vector, $\phi_p(x) = [x_1, x_2, x_1^2, x_1 x_2, x_2^2]^\top$.

The GLDS kernel between two sequences of vectors $X = \{\mathrm{x}_t\}_{t=1\ldots T_X}$ and $Y = \{\mathrm{y}_t\}_{t=1\ldots T_Y}$ is given as a rescaled dot product between average expansions:

$$K_{GLDS}(X,Y) = \frac{1}{T_X}\sum_{t=1}^{T_X} \phi_p(\mathrm{x}_t)^\top \; \mathbf{M}_p^{-1} \; \frac{1}{T_Y}\sum_{s=1}^{T_Y} \phi_p(\mathrm{y}_s) \tag{1}$$

where $\mathbf{M}_p$ is the second moment matrix of polynomial expansions ϕ_p estimated on some background population, or its diagonal approximation for more efficiency.

Conceived in this way, the GLDS kernel is difficult to tune, because the size of the explicit polynomial expansion ϕ_p becomes intractable for polynomial expansions with maximal degree p higher than 3. Indeed, let d be the dimension of the input space, the dimension of the expansion is $D = \frac{(d+p)!}{d!p!}$. In practice d is about 25, and D becomes too large when $p > 3$ (*e.g.* $D = 23,751$ when $d = 25$ and $p = 4$). That is why in practice GLDS SVM based systems use an expansion with monomials up to degree 3.

An interesting problem then is to find a tractable way to compute or approximate (1) for any p. A more general problem is how to to provide a finite-dimensional form of (1) for any expansion ϕ including infinite ones, so as to really exploit the "kernel trick". By this way, Radial Basis

Functions (RBF) expansion could also be used. This is the purpose of the next section.

3. A rich class of kernels

Let's consider the class of sequence kernels of the form:

$$\begin{aligned}\hat{K}(X,Y) &= \tfrac{1}{T_X}\textstyle\sum_{t=1}^{T_X}\phi(\mathrm{x}_t)^\top \quad \mathbf{M}^{-1} \quad \tfrac{1}{T_Y}\textstyle\sum_{s=1}^{T_Y}\phi(\mathrm{y}_s) \\ &= \overline{\phi}(X)^\top \quad \mathbf{M}^{-1} \quad \overline{\phi}(Y)\end{aligned} \tag{2}$$

where

- ϕ is a vector expansion of size $D \leq +\infty$ defining a Mercer kernel k:

  $$k(\mathrm{x},\mathrm{y}) = \phi(\mathrm{x})^\top\phi(\mathrm{y}) \tag{3}$$

- $\mathbf{M} = E(\phi\phi^\top)$ is the second moment matrix of expansions ϕ estimated on a set of background population $B = \{\mathrm{b}_1,\ldots,\mathrm{b}_n\}$ (of size n). $\mathbf{M}$ can be expressed as a matrix product

  $$\mathbf{M} = \frac{1}{n}\Phi_B\Phi_B^\top \tag{4}$$

  where $\Phi_B = [\phi(\mathrm{b}_1),\ldots,\phi(\mathrm{b}_n)]$ is the $D \times n$ matrix of background vector expansions.

Note that $\hat{k}(\mathrm{x},\mathrm{y}) = \phi(\mathrm{x})^\top\mathbf{M}^{-1}\phi(\mathrm{y})$ is also a kernel satisfying the Mercer condition, with a rescaling process in the feature space defined by the expansion ϕ. The sequence kernel can be written with this re-weighted kernel as $\hat{K}(X,Y) = \frac{1}{T_X T_Y}\sum_t\sum_s\hat{k}(\mathrm{x}_t,\mathrm{y}_s)$.

Note also that the GLDS expansion ϕ_p does not lead exactly to the standard polynomial kernel $k(x,y) = (c + \mathrm{x}\cdot\mathrm{y})^p$ (each monomial would have to be normalized with the appropriate coefficients).

Note finally that the kernel $\hat{K}$ is invariant to sequence permutation. It is thus a kernel between sets of vectors. We use however the terminology "sequence kernel" for simplicity.

3.1. Expressing $\hat{K}$ in a dual form

In this section, we show how to express the re-weighted vector kernel $\hat{k}$ as a function of the standard vector kernel k, and of the set of background vectors $B = \{\mathrm{b}_1,\ldots,\mathrm{b}_n\}$ considered for the rescaling operated by the matrix $\mathbf{M}^{-1}$.

Let's consider the thin Singular-Value Decomposition (SVD) of background expansions Φ_B:

$$\Phi_B = USV^\top \tag{5}$$

where U and V are orthogonal matrices of sizes $D \times r$ and $n \times r$ respectively, $r \leq \min(n, D)$ being the rank of Φ_B. Then

$$\begin{aligned}\mathbf{M} &= \tfrac{1}{n}USV^\top\, VSU^\top \\ &= U\left(\tfrac{1}{n}S^2\right)U^\top\end{aligned} \tag{6}$$

Note that in the general case $\mathbf{M}$ is not guaranteed to be invertible and has to be regularized, by replacing for instance $\mathbf{M}$ by $\mathbf{M} = E(\phi\phi^\top) + \frac{1}{n}\varepsilon I$. This regularization is needed for statistical reasons in cases where the dimension of the feature space D is larger than the number of data points n [7]. We refer to [8] for the theoretical development using such regularization.

To invert (6), we use the fact that all combinations of rational operations do not change eigenvectors U and apply only on singular values included in $\frac{1}{n}S^2$. We can thus consider the pseudo-inversion [9]

$$\begin{aligned}\mathbf{M}^{-1} &= nUS^{-2}U^\top \\ &= n\Phi_B VS^{-4}V^\top\Phi_B^\top\end{aligned} \tag{7}$$

The kernel Gram matrix on background data, defined by $\mathbf{K}_{i,j} = k(\mathrm{b}_i,\mathrm{b}_j)$, can be written $\mathbf{K} = \Phi_B^\top\Phi_B$. Using (5), it has an explicit singular value decomposition $\mathbf{K} = VS^2V^\top$. Considering the pseudo-inverse $\mathbf{K}^{-2} = VS^{-4}V^\top$, the kernel $\hat{k}$ can be written as:

$$\begin{aligned}\hat{k}(x,y) &= n \quad \phi(x)^\top\Phi_B \quad \mathbf{K}^{-2} \quad \Phi_B^\top\phi(y) \\ &= n \quad \Psi_B(x)^\top \quad \mathbf{K}^{-2} \quad \Psi_B(y)\end{aligned} \tag{8}$$

where we define the vector mapping of size n, using the vector kernel (3), as:

$$\Psi_B(\mathrm{x}) = [k(\mathrm{b}_1,\mathrm{x}),\ldots,k(\mathrm{b}_n,\mathrm{x})]^\top \tag{9}$$

Note that Ψ_B is exactly the empirical kernel map defined in [7].

By linearity in the feature space, we can finally write $\hat{K}$ in a finite-dimensional form as

$$\begin{aligned}\hat{K}(X,Y) &= \tfrac{1}{T_X T_Y}\textstyle\sum_{t=1}^{T_X}\sum_{s=1}^{T_Y}\hat{k}(\mathrm{x}_t,\mathrm{y}_s) \\ &= n\;\overline{\Psi}_B(X)^\top\;\mathbf{K}^{-2}\;\overline{\Psi}_B(Y)\end{aligned} \tag{10}$$

where we define the sequence map of size n

$$\overline{\Psi}_B(X) = \left[\begin{array}{c}\frac{1}{T_X}\sum_{t=1}^{T_X}k(\mathrm{b}_1,\mathrm{x}_t) \\ \ldots \\ \frac{1}{T_X}\sum_{t=1}^{T_X}k(\mathrm{b}_n,\mathrm{x}_t)\end{array}\right] \tag{11}$$

In practice, the number of background vectors available for speech application is very high. In the case of monitoring conversations, the size n of background data available can be enormous. The computation of $\hat{K}$ would thus be intractable since its complexity is $O\left(n(T_X + T_Y + n)\right)$. In the next section, we use a low-rank matrix decomposition to provide an approximate but tractable form for (10).

4. Incomplete Cholesky Decomposition of the Gram Matrix

Current methods of reducing training data for kernel-based methods correspond to low-rank approximations of the gram matrix [10, 11]. The goal of these methods is to pick up a subset $C \subset B$ that would allow an approximation of the gram matrix $\mathbf{K}_{i,j} = k\left(\mathrm{x}_i,\mathrm{x}_j\right)_{(\mathrm{x}_i,\mathrm{x}_j)\in B^2}$ with a lower rank matrix, so as to rewrite kernel formulas with lower complexity.

An appealing technique is the Incomplete Cholesky Decomposition (ICD). The algorithm, described in [12], has a relatively low complexity $O(m^2n)$, if m is the desired size of the set C. Besides, it does not require to keep in memory the entire gram matrix $\mathbf{K}$ at any time.

Given a gram matrix $\mathbf{K}$ of size $n \times n$ (the actual rank of $\mathbf{K}$ may be smaller than n), the ICD of $\mathbf{K}$ is a $n \times m$ matrix $\mathbf{G}_m$, such that $\mathbf{K}$ can be approximated by $\mathbf{G}_m {\mathbf{G}_m}^\top$. $\mathbf{G}_m$, with rank $m < n$, is spanned by the columns of $\mathbf{K}$ indexed by a sequence $I = \{i_1, \dots, i_m\} \subset \{1, \dots, n\}$. By doing so, we can consider that the ICD provides a codebook $C = \{\mathrm{b}_{i_1}, \dots, \mathrm{b}_{i_m}\} \subset B$. In the following, we show how to express our sequence kernel with a tractable form involving C instead of B.

It can be shown [13] that $\mathbf{G}_m$ can be written:

$$\mathbf{G}_m = \mathbf{K}(:, I)\mathbf{K}(I, I)^{-1/2} \tag{12}$$

where $\mathbf{K}(:, I)$ means all columns of $\mathbf{K}$ indexed by I. With the same notation, $\mathbf{K}(I, I)$ is a $m \times m$ gram matrix with $\{\mathrm{b}_{i_1}, \dots, \mathrm{b}_{i_m}\}$ as entries.

The fact that Φ_B and ${\mathbf{G}_m}^\top$ have the same square ($\mathbf{K} = \Phi_B^\top \Phi_B \approx \mathbf{G}_m {\mathbf{G}_m}^\top$) implies that there exists a $D \times m$ orthogonal matrix U such that we can consider the incomplete decomposition (instead of (5)):

$$\Phi_B = U {\mathbf{G}_m}^\top \tag{13}$$

The second moment matrix $\mathbf{M}$ can thus be replaced by $\mathbf{M} = \frac{1}{n} U {\mathbf{G}_m}^\top \mathbf{G}_m U^\top$. This approximation amounts to regularizing the second moment matrix. Given the previous decomposition, we can invert

$$\begin{aligned} \mathbf{M}^{-1} = &\quad n\ U\ ({\mathbf{G}_m}^\top \mathbf{G}_m)^{-1}\ U^\top \\ = &\quad U\ \ K(I, I)^{1/2}\ \mathbf{R}^{-1}\ K(I, I)^{1/2}\ \ U^\top \end{aligned} \tag{14}$$

where we define using (12) the $m \times m$ matrix:

$$\mathbf{R} = \frac{1}{n}\mathbf{K}(:, I)^\top \mathbf{K}(:, I) \tag{15}$$

The ICD guarantees that $\mathrm{rank}({\mathbf{G}_m}^\top \mathbf{G}_m) = \mathrm{rank}(\mathbf{G}_m) = m$, which in turn guarantees that ${\mathbf{G}_m}^\top \mathbf{G}_m$ and $\mathbf{R}$ are invertible.

We can also derive from (12) and (13) that

$$\Phi_B^\top = \mathbf{G}_m U^\top = \mathbf{K}(:, I)\mathbf{K}(I, I)^{-1/2} U^\top.$$

If we assume that every expansion $\phi(\mathrm{x})$ belongs to the convex hull of the background expansions included in Φ_B, then we can show [8] that

$$\phi(\mathrm{x})^\top = \Psi_C(\mathrm{x})^\top \mathbf{K}(I, I)^{-1/2} U^\top$$

where $\Psi_C(\mathrm{x})$ is the (reduced) map involving the m codebook vectors $\{\mathrm{b}_{i_1}, \dots, \mathrm{b}_{i_m}\}$ extracted from the ICD:

$$\Psi_C(\mathrm{x}) = [k(\mathrm{b}_{i_1}, \mathrm{x}), \dots, k(\mathrm{b}_{i_m}, \mathrm{x})]^\top \tag{16}$$

Replacing in (2) the new expressions of $\mathbf{M}^{-1}$ and $\phi(X)$ (resp. $\phi(Y)$) leads to the new form of our sequence kernel:

$$\hat{K}_{ICDS}(X, Y) = \overline{\Psi}_C(X)^\top\ \mathbf{R}^{-1}\ \overline{\Psi}_C(Y) \tag{17}$$

where we consider the sequence map

$$\overline{\Psi}_C(X) = \begin{bmatrix} \frac{1}{T_X} \sum_{t=1}^{T_X} k(\mathrm{b}_{i_1}, \mathrm{x}_t) \\ \dots \\ \frac{1}{T_X} \sum_{t=1}^{T_X} k(\mathrm{b}_{i_m}, \mathrm{x}_t) \end{bmatrix} \tag{18}$$

The computational complexity of $\hat{K}_{ICDS}(X, Y)$ is $O(m(T_X + T_Y + m))$. In practice the value of m can be chosen to be much lower than n, which leads in turn to an efficient kernel computation. Moreover, diagonal approximations of $\mathbf{R}$ can be considered to make the computation highly efficient (we do not consider such approximations in this paper). Note finally that computation of $\mathbf{R}^{-1}$ can be done off-line.

The sequence kernel $\hat{K}_{ICDS}$ given by (17) has a similar form to the RKHS Sequence kernel given in our previous work [14], where we had adopted the same procedure as Campbell in [6] to conceive a kernel between two sequences. This procedure consists in training a discriminant model (with outputs 0/1) on a sequence (in a Reproducing Kernel Hilbert Space generated by k) and testing on the other. After some approximation a symmetric kernel, that satisfies the Mercer Conditions, is obtained.

The kernel arising from our last approach has the expression:

$$K_{RKHS}(X, Y) = \overline{\Psi}_C(X)^\top\ \mathbf{K}_C^{-2}\ \overline{\Psi}_C(Y) \tag{19}$$

where $C = \{c_1, \dots, c_m\}$ is a set of codebook vectors obtained by a vector quantization of the background set B. $\mathbf{K}_C$ is the Gram matrix on C: $\mathbf{K}_{C\,i,j} = K(c_i, c_j)$.

5. Experiments

5.1. Corpora and front-end

Our experiments used female data from the NIST 2004 Speaker Recognition Evaluation, in accordance with the development protocol defined by the Biosecure project [15]. In this scenario, we consider 113 background speakers for tuning the system, and more than 7000 tests involving 181 target speakers and 368 testing sequences. All sequences, including about 2 minutes of speech, come from the NIST SRE 2004 evaluation database in core conditions (1side-1side).

To extract acoustic vectors from a speech sequence, 12 MFCC and their first order time derivatives are extracted on a 16ms window, at a 10ms frame rate. The derivative of the energy logarithm is also added. Then, a speech activity detector discards silence frames, using an unsupervised bi-Gaussian model [16]. Finally, the 25-dimensional input vectors are warped [17] over 3 sec windows.

5.2. System implementation

The first step is to run the ICD on the Gram matrix of the background population. In our case, it would be computationally expensive to run this iterative algorithm on a huge amount n of data, as we need to memorize a matrix $\mathbf{G}_m$ of size $n \times m$. Our experiments showed that if we finally pick up about $m \sim 5000$ codebook vectors, there is no point in considering all background data available. We have roughly the same performance when considering $20,000$ background vectors or $200,000$ background vectors. We thus fix $m = 5000$ and run the ICD on the Gram matrix of $20,000$ background vectors picked up randomly in the background corpus. These vectors have to be representative of observed speech features, as the set of background vectors used to estimate $\mathbf{M}_p$ for the GLDS kernel.

Once the codebook for the mapping (18) is chosen, we can compute off-line the normalisation matrix $\mathbf{R}$ defined by (15). In a SVM speaker verification scheme, we have to train several target speaker models using a common set of background sequences considered as impostors, whose characteristics can be computed off-line and kept in memory. To save computations for training target speaker models when some sequences are given to the system, we decide to pre-compute the sequence kernel between all pairs of impostors, and to keep in memory all rescaled maps of impostors, defined by $\mathbf{R}^{-1}\overline{\Psi}_C$. By doing so, when a target speaker sequence is given to the system for training, we only need to compute its map $\overline{\Psi}_C$, and then a dot product between this map and all background rescaled maps, in order to obtain all kernel values to train a SVM.

The testing procedure can be made efficient with a similar trick. If we note (T_i) the training sequences (impostors + given target *sp*), the discriminative function (defined with some Lagrangian coefficients (α_i)) can be encompassed into a single m-dimensional vector $\hat{\omega}_{sp}$ (as was done in [6]):

$$\begin{aligned} f(\cdot) &= \textstyle\sum_i \alpha_i \hat{K}(T_i,\cdot)+\beta \\ &= \overline{\Psi}_C(\cdot)^\top \underbrace{\sum_i \alpha_i \mathbf{R}^{-1}\overline{\Psi}_C(T_i)}_{\hat{\omega}_{sp}} +\beta \end{aligned} \qquad (20)$$

5.3. Choice of the kernel parameters

In this section, we discuss how to choose the parameters of the kernel k chosen to define $\hat{K}$ (degree for a polynomial, width of a gaussian function).

5.3.1. Polynomial kernels

Considering a polynomial kernel $k_p(\mathrm{x},\mathrm{y}) = (c+\mathrm{x}\cdot\mathrm{y})^p$, we can see comparing results when $c=0$ (Fig. 1) and $c=1$ (Fig. 2) that it is better to take a non-zero c. This means that it is better to take into account all monomials with a degree equal or lower than p, as was done with the GLDS kernel (when $c=0$ only monomials with degree p are taken into account).

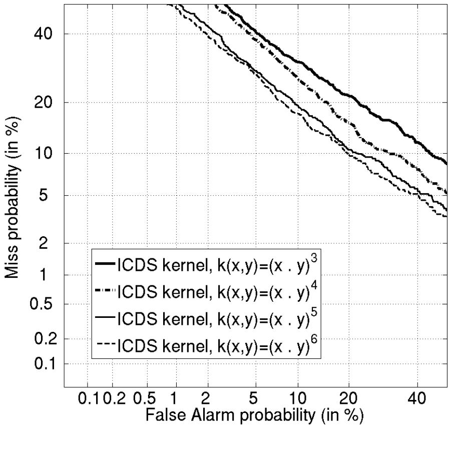

Figure 1: Performance according to the degree of the polynomial vector kernel of the form $(x\cdot y)^p$

Moreover, Figs.1 and 2 show that we have better performance using a degree higher than 3. This suggests that the GLDS kernel would perform better if it could consider monomials with degrees higher than 3 in the expansion ϕ_p. Unfortunately, such an extension is not tractable in our application.

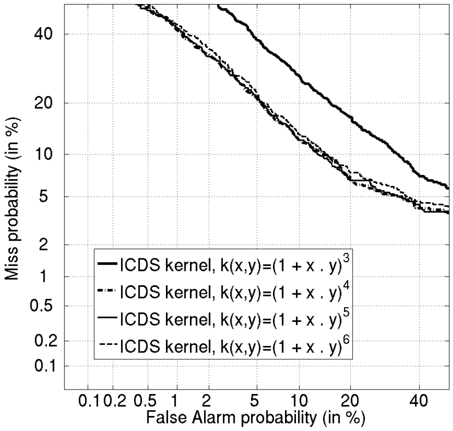

Figure 2: Performance according to the degree of the polynomial vector kernel of the form $(1+x\cdot y)^p$

5.3.2. RBF kernels

Considering a RBF kernel $k_{rbf}(\mathrm{x},\mathrm{y}) = e^{-\gamma\|\mathrm{x}-\mathrm{y}\|^2}$, [7] recommends to choose the parameter γ in the order of $\gamma_0 = 1/2d\sigma^2$, where σ^2 is the mean of the variance of each component of input vectors in $\Re^d$. With the front-end processing used in our experiments, it corresponds to $\gamma_0 \approx 0.3$. Our experiments confirm this recommendation.

Indeed, if γ is too high, the vector kernel will fit too much to the data: $\mathbf{K}$ will be close to an identity matrix (maximal rank), and our sequence mapping defined in (17) will amount to counting how many vectors from a sequence lie in a narrow neighbourhood of each respective codebook vector. On the contrary, if γ is too low, the rank of $\mathbf{K}$ will be low (with eigenvalues decreasing rapidly) and we will only consider a (too) small number of features. Experiments in Fig. 3 show that if γ lies in a reasonable range around γ_0, then performances are

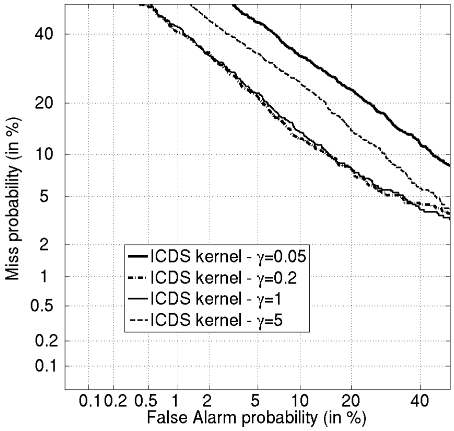

Figure 3: Performance according to the spreading of the RBF kernel

roughly the same (*resp.* are degraded when γ reaches some obsolete values).

5.4. Comparison with GLDS

The best performances of our new ICD Sequence kernel (in this experiment) were obtained using an RBF kernel with $\gamma = 0.2$. In Fig. 4, we compare the performance of this ICD Sequence kernel to

- The GLDS approach [6] with a polynomial expansion of degree 3 and a diagonal approximation of the second moment matrix $\mathbf{M}_p$.
- The GLDS approach with a polynomial expansion of degree 3 and a full matrix $\mathbf{M}_p$, so as to compare systems with comparable complexity.
- Our previous approach [14], which is referred to as a RKHSS kernel

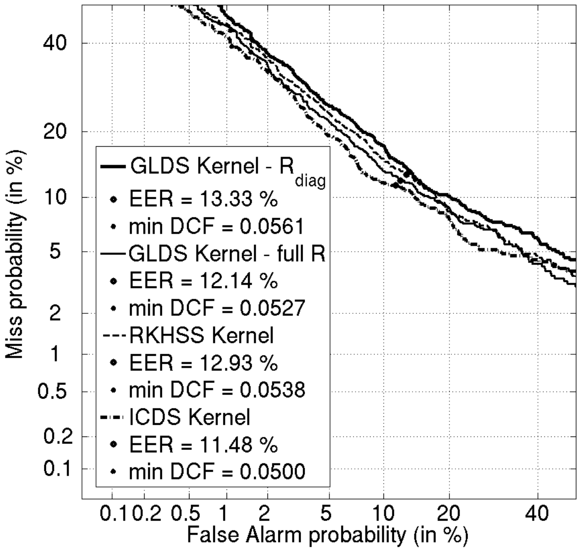

Figure 4: Comparison of the GLDS, RKHSS and ICDS kernels

The results show that our new system is outperforming all the others at all operating points. Recent experiments show that this is still de case with other front-end processing exploiting short-term spectral information.

6. Conclusion

Using low-rank decomposition of the Gram matrix, we presented a new framework to generalize the GLDS kernel to any feature space expansion that defines a Mercer kernel. The sequence kernel we obtained, the ICD kernel, leads to performances also better than the GLDS ones and also than our previous approach. Moreover, many extensions are possible. For instance, it is interesting to consider a covariance matrix $\mathbf{M}$ instead of the second order moment one because the kernel would then correspond to a Malahanobis distance in the feature space. The theoretical and experimental development of such an extension and others can be found in [8].

7. References

[1] M. Schmidt and H. Gish, “Speaker identification via support vector machines,” in *Proc. ICASSP*, 1996.

[2] Z. Lei, Y. Yang, and Z. Wu, “Mixture of support vector machines for text-independent speaker recognition,” in *Proc. Interspeech*, 2005.

[3] J. Kharroubi, D. Petrovska-Delacretaz, and G. Chollet, “Combining gmm's with suport vector machines for text-independent speaker verification,” in *Proc. Eurospeech*, 2001.

[4] P. Ho and P Moreno, “SVM kernel adaptation in speaker classification and verification,” in *Proc. ICSLP*, 2004.

[5] V. Wan, *Speaker Verification using Support Vector Machines*, Ph.D. thesis, University of Sheffield, 2003.

[6] W.M. Campbell, J. Campbell, D. Reynolds, E. Singer, and P. Torres-Carrasquillo, “Support vector machines for speaker and language recognition,” *Computer Speech and Language*, 2005.

[7] B. Schölkopf, S. Mika, C. J.C.Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola, “Input space versus feature space,” *IEEE Trans. Neural Networks Proc.*, vol. 100, no. 5, pp. 1000–1017, 1999.

[8] J. Louradour, K. Daoudi, and F.R. Bach, “SVM speaker verification using an incomplete cholesky decomposition sequence kernel : extended version,” IRIT research report, www.irit.fr/~jerome.louradour/papers/, 2005.

[9] G.H. Golub and C.F. Van Loan, *Matrix Computation*, The John Hopkins Univ. Press, 1996.

[10] A.J. Smola and B. Schölkopf, “Sparse greedy matrix approximation for machine learning,” in *Proc. ICML*, 2000.

[11] S. Fine and K. Scheinberg, “Efficient SVM training using low-rank kernel representations,” *Journal of Learning Machine Research*, vol. 2, 2001.

[12] F.R. Bach and M.I. Jordan, “Kernel independent component analysis,” *Journal of Learning Machine Research*, vol. 3, 2002.

[13] F.R. Bach and M.I. Jordan, “Predictive low-rank decomposition for kernel methods,” in *Proc. ICML*, 2005.

[14] J. Louradour and K. Daoudi, “Conceiving a new sequence kernel and applying it to SVM speaker verification ,” in *Proc. Interspeech*, 2005.

[15] “Biosecure network of excellence : Biometrics for secure authentification,” http://www.biosecure.info, 2005.

[16] I. Magrin-Chagnolleau, G. Gravier, and R. Blouet, “Overview of the 2000-2001 elisa consortium research activities,” in *Proc. A Speaker Odyssey*, 2001.

[17] J. Pelecanos and S. Sridharan, “Feature warping for robust speaker verification,” in *Proc. Speaker Odyssey*, 2001.