14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

UNSUPERVISED SPEAKER INDEXING USING ONE-CLASS SUPPORT VECTOR MACHINES

Belkacem Fergani[1,2]*, Manuel Davy*[2]*, and Amrane Houacine*[1]

[1] LCPTS/Electronics and Computer Sciences, USTHB
B.P 32, El Alia, Bab Ezzouar, Algiers, Algeria
bfergani@msn.com, a_houacine@lycos.com
[2] LAGIS/CNRS, BP48, 59651 Villeneuve d'Ascq Cedex, France
Manuel.Davy@ec-lille.fr

ABSTRACT

This paper addresses the unsupervised speaker change detection problem, which is a key issue in any audio indexing process. Here, we derive a new approach based on the Kernel Change Detection algorithm introduced recently by Desobry et al. This new algorithm does not require explicit modeling of the data, and is able to deal with large dimensional acoustic feature vectors. Several experiments using RT'03 NIST data show the efficiency of the algorithm. Comparisons to the well known GLR-BIC algorithm are presented, for various parameter settings.

1. INTRODUCTION

The automated indexing of audio databases is an active research area, which raises many challenging problems. Speech-related audio indexing requires, as sub-tasks, speech/non-speech discrimination, speech recognition and unsupervised speaker segmentation. The latter task consists of labeling an audio record into segments, each containing one and only one speaker at a given time[1].

This problem has been studied in several previous works (see, e.g., [1, 2] and references therein), and standard techniques are based on the analysis of acoustic feature vectors denoted $\mathbf{x}(n)$ (generally, MFCCs), living in the space $\mathcal{X}$, extracted framewise[2] from the signal $x(n)$. Given the time series of acoustic features $\mathbf{x}(n)$, $n = 1, 2, \ldots$, most techniques apply two sliding windows apart from the current analysis time n. The analysis window located before n defines the *immediate past set* $\mathsf{X}_\mathrm{p}(n) = \{\mathbf{x}(n - m_\mathrm{p}), \ldots, \mathbf{x}(n-1)\}$ of m_p acoustic features, whereas the other window contains m_f features vectors, forming the *immediate future set* $\mathsf{X}_\mathrm{f}(n) = \{\mathbf{x}(n+1), \ldots, \mathbf{x}(n+m_\mathrm{f})\}$. In standard systems, speaker segmentation comes down to comparing the vectors in $\mathsf{X}_\mathrm{p}(n)$ and $\mathsf{X}_\mathrm{f}(n)$. This is performed using the Generalized Likelihood Ratio (GLR) test, either directly [2] or indirectly, such as in the Bayesian Information Criterion (BIC) approach [3]. GLR-based tests require a probabilistic data model to be given, generally a single Gaussian, or a mixture of several Gaussians.

Subsequent to change detection, speaker indexing requires to group signal segments (located in between detected changes) into single speaker segments. This can be performed by learning and maintaining a set of speaker models (integrated approach [4, 5]) or step-by-step, by applying a clustering technique [3], or by a fusion of both techniques [1, 6]

[1] In real records, it may happen that several speakers can be heard at the same time, but is a reasonable and often used simplifying assumption that only one speaker is present at a time.

[2] Here, for notational simplicity, it is assumed that one acoustic feature vector is extracted at each time n.

In this paper, we propose to apply the new Kernel Change Detection (KCD) algorithm introduced in [7] to the problem of speaker change detection and clustering tasks known as speaker segmentation process in an indexing framework. In this context, our approach is original as, to our knowledge, there is no equivalent SVM based method reported in the literature. A slightly similar work is reported in [8] where, however, the algorithm presented uses a supervised strategy.

Different from the above approaches, this paper presents a kernel unsupervised approach aimed at comparing the sets $\mathsf{X}_\mathrm{p}(n)$ and $\mathsf{X}_\mathrm{f}(n)$ at each time n, by defining a dissimilarity measure between these sets. In that respect, KCD is similar to GLR-based approaches. However, KCD being based on two one-class Support Vector Machines (SVM), the complexity of the model fitted to the data is controlled, while being able to represent any data configuration, even with large dimensional, heterogeneous acoustic features.

In Section 2, we recall the principle of the KCD algorithm. In Section 3, we present our new KCD-based speaker segmentation algorithm in terms of the acoustic features used, as well as the abrupt changes detection algorithm, and the segments grouping strategy (clustering). In Section 4, we present experimental results for the RT03S database [9]. Our results are compared to those obtained with the GLR-BIC approach, thanks to the NIST scoring procedure. Section 5 presents conclusions and future work directions.

2. THE KERNEL CHANGE DETECTION ALGORITHM

The Kernel Change Detection (KCD) algorithm, introduced in [7], consists of estimating the *level sets* of the underlying distributions of $\mathsf{X}_\mathrm{p}(n)$ and $\mathsf{X}_\mathrm{f}(n)$ at each time n. In the following, we focus on any of these two sets, denoted X for simplicity. Assume the feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$ in X have been generated i.i.d. from a given probability density function (pdf) $\mathsf{p}(\mathbf{x})$, i.e., $\mathsf{p}(\mathsf{X}) = \prod_{i=1}^{m} \mathsf{p}(\mathbf{x}_i)$ (this is the assumption made in, e.g., the BIC approach). The level set S^λ of $\mathsf{p}(\mathbf{x})$ is the set of points in the acoustic features space $\mathcal{X}$ such that $\mathsf{p}(\mathbf{x}) \geq \lambda$, where λ is some positive constant. Whenever $\mathcal{X} = \mathbb{R}^d$ and $\mathsf{p}(\mathbf{x})$ is the d-dimensional multivariate Gaussian, S^λ is an ellipsoid in a d-dimensional space. In the more general case, the level set may have any (generally smooth) shape. KCD applies one-class support vector machines in order to estimate a level set from the feature vectors, without assuming Gaussianity.

2.1 One-class SVM

Let us define a kernel function $\mathsf{k}(\mathbf{x}_1, \mathbf{x}_2)$ over $\mathcal{X}$ to the reals. In the following, the kernel can be though of as Gaussian

$$\mathsf{k}(\mathbf{x}_1, \mathbf{x}_2) = \exp -\frac{1}{2\sigma^2} \|\mathbf{x}_1 - \mathbf{x}_2\|_{\mathcal{X}}^2 \qquad (1)$$

where $\|\cdot\|^2_{\mathcal{X}}$ is a norm in $\mathcal{X}$. This kernel being definite positive, it induces a Reproducing Kernel Hilbert Space (RKHS) denoted $\mathcal{H}$ [10]. The RKHS $\mathcal{H}$ is a linear vector space of functions from $\mathcal{X}$ to $\mathbb{R}$ endowed with a dot product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ and the corresponding norm $\|\cdot\|_{\mathcal{H}}$. Moreover, $\mathcal{H}$ is complete for the norm[3] $\|\cdot\|_{\mathcal{H}}$ and the *reproducing kernel property* holds: for any $\mathbf{x}\in\mathcal{X}$ and any function $\mathsf{f}(\cdot)\in\mathcal{H}$, the function $\mathsf{k}\big(\mathbf{x},\cdot\big)$ belongs to $\mathcal{H}$ and $\langle \mathsf{f}(\cdot), \mathsf{k}\big(\mathbf{x},\cdot\big)\rangle_{\mathcal{H}} = \mathsf{f}(\mathbf{x})$. In other words, computing the possibly non-linear function $\mathsf{f}(\cdot)$ at any point can be performed by computing a (linear) dot product in $\mathcal{H}$ (this is the so-called *kernel trick*). Given the Gaussian kernel over $\mathcal{X}$, and a set of feature vectors X, one-class SVM estimates the level set as

$$\mathsf{S}^{\lambda} \;=\; \{\mathbf{x}\in\mathcal{X} | \mathsf{f}^{\lambda}(\mathbf{x}) + b \geq 0\} \tag{2}$$

The level set estimation problem becomes that of estimating a function in $\mathcal{H}$ close to the true, unknown level set. $\mathcal{H}$ being given (through the choice of $\mathsf{k}\big(\cdot,\cdot\big)$), the function $\mathsf{f}^{\lambda}(\cdot)$ and b are found by minimizing the regularized risk

$$\mathsf{R}^{\mathrm{reg}}\left[\mathsf{f}(\cdot)+b\right] \;=\; C\mathsf{R}^{\mathrm{emp}}_{\mathsf{X}}\left[\mathsf{f}(\cdot)+b\right] + \frac{1}{2}\|\mathsf{f}(\cdot)\|^2_{\mathcal{H}} \tag{3}$$

where the *empirical risk* $\mathsf{R}^{\mathrm{emp}}_{\mathsf{X}}\left[\mathsf{f}(\cdot)+b\right]$ indicates how well the function $\mathsf{f}(\cdot)+b$ fits the feature vectors in the set X, and $\|\mathsf{f}(\cdot)\|^2_{\mathcal{H}}$ penalizes too complex functions, that is, level sets with too complicated shapes. The positive parameter C is user-defined, and it trades off between the goodness of fit and the complexity of $\mathsf{f}(\cdot)+b$. In practice, the empirical risk is computed as the average cost incurred by errors, penalizing functions $\mathsf{f}(\cdot)+b$ that are negative at some training vectors:

$$\mathsf{R}^{\mathrm{emp}}_{\mathsf{X}}\left[\mathsf{f}(\cdot)+b\right] \;=\; \frac{1}{m}\sum_{i=1}^{m}\max\big(0, \mathsf{f}(\mathbf{x}_i)+b\big) \tag{4}$$

Thanks to Eq. (3), finding the level set comes down to solving an optimization problem in $\mathcal{H}$. It can be shown [10] that the function that minimizes the regularized risk can be written for any $\mathbf{x}$ as

$$\mathsf{f}^{\lambda}(\mathbf{x}) + b \;=\; \sum_{i=1}^{m}\alpha_i \mathsf{k}\big(\mathbf{x},\mathbf{x}_i\big) + b \tag{5}$$

where the weights α_i are actually the Lagrange multiplier introduced to solve the minimization problem of Eq. (3). The corresponding dual optimization problem, which yields the α_i's, is quadratic with linear constraints (easy to solve). This formulation yields, however, a difficulty: the parameter C happens to be uniquely related to the level λ and the relation between C and λ is complex and may not be written in closed-form. In practice, C has to be tuned by trial and errors, and its relation to the desired λ is never assessed. This problem can be overcome by using the ν one-class SVM [10], which replaces the parameter C and the empirical risk by an alternate definition and which may be written as the optimization problem, where the optimal α_i's are to be plugged into Eq. (5):

$$\begin{aligned}
&\text{Minimize} \quad \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j\mathsf{k}\big(\mathbf{x}_i,\mathbf{x}_j\big) \text{ w.r.t. } \{\alpha_1,\ldots,\alpha_m\}\\
&\text{with} \quad 0\leq\alpha_i\leq\frac{1}{\nu m} \text{ for } i=1,\ldots,m\\
&\text{and} \quad \sum_{i=1}^{m}\alpha_i = 1
\end{aligned} \tag{6}$$

[3] A space is complete for some norm if all its Cauchy sequences converge into it in terms of this norm.

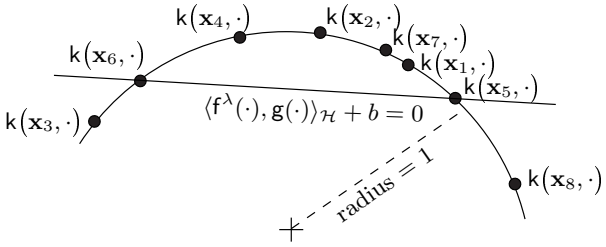

Figure 1: Geometrical interpretation of one-class SVM in $\mathcal{H}$. The mapped training vectors $\mathsf{k}\big(\mathbf{x}_i,\cdot\big)$, $i=1,\ldots,m$ are all located on a hypersphere with radius one. The function $\mathsf{f}^{\lambda}(\cdot)$ and b define a hyperplane with equation $\langle \mathsf{f}^{\lambda}(\cdot), \mathsf{g}(\cdot)\rangle_{\mathcal{H}} + b = 0$. Most of the data are located on the side of the hyperplane that is away from the hypersphere origin (the corresponding weights α_i are zero ($i\in\{1,2,4,7\}$), whereas the outliers $\mathsf{k}\big(\mathbf{x}_3,\cdot\big)$ and $\mathsf{k}\big(\mathbf{x}_8,\cdot\big)$ are located on the other side and are such that $\alpha_3 = \alpha_8 = 1/\nu m$. The points located on the intersection of the hyperplane and the hypersphere verify $0<\alpha_i<1/\nu m$ ($i\in\{5,6\}$).

In the above optimization problem, the parameter ν tunes the fraction of feature vectors in X that are located *outside* the level set S^{λ} (these are called *outliers*). More precisely, it can be shown that, asymptotically, ν *is* the fraction of outliers [10], which means that $1-\nu$ is the probability mass enclosed inside the level set S^{λ}, yielding a relation between ν and λ. In practice, tuning ν is even more intuitive that tuning λ. For example, $\nu = 0.2$ means that at most (and asymptotically exactly) 20% of the feature vectors in X are outliers. In the following, we denote the SVM level sets estimated from $\mathsf{X}_{\mathrm{p}}(n)$ and $\mathsf{X}_{\mathrm{f}}(n)$ with $\mathsf{S}^{\nu}_{\mathrm{p}}(n)$ and $\mathsf{S}^{\nu}_{\mathrm{f}}(n)$. Two important remarks are that 1) most of the weights α_i are zero, the nonzero weights correspond to the outliers which are also called *Support Vectors* (SVs) and 2) when X is one of the sliding windows $\mathsf{X}_{\mathrm{p}}(n)$ or $\mathsf{X}_{\mathrm{f}}(n)$, the α_i's can be updated easily for consecutive times n and $n+1$, as, e.g., $\mathsf{X}_{\mathrm{p}}(n)$ and $\mathsf{X}_{\mathrm{p}}(n+1)$ differ by only two feature vectors – see [11] for details.

The one-class SVM admits a simple geometrical interpretation in $\mathcal{H}$: First, the feature vectors in $\mathcal{X}$ are *mapped* to $\mathcal{H}$ by $\mathbf{x}\;\rightarrow\;\mathsf{k}\big(\mathbf{x},\cdot\big)$. Second, the mapped training vectors all have norm one in $\mathcal{H}$ when using the Gaussian kernel, because $\|\mathsf{k}\big(\mathbf{x},\cdot\big)\|^2_{\mathcal{H}} = \langle \mathsf{k}\big(\mathbf{x},\cdot\big), \mathsf{k}\big(\mathbf{x},\cdot\big)\rangle_{\mathcal{H}} = \mathsf{k}\big(\mathbf{x},\mathbf{x}\big) = 1$ (using the reproducing kernel property), they are thus located on a hypersphere with radius one. Third, solving Eq. (6) can be seen as finding, in $\mathcal{H}$, the hyperplane orthogonal to $\mathsf{f}(\cdot)$ such that it is located as far as possible from the hypersphere origin, separating it from the training data $\mathsf{k}\big(\mathbf{x}_i,\cdot\big)$, Fig. 1. From estimated level sets, KCD defines a dissimilarity measure between the two sets of features $\mathsf{X}_{\mathrm{p}}(n)$ and $\mathsf{X}_{\mathrm{f}}(n)$, by comparing the corresponding level sets $\mathsf{S}^{\nu}_{\mathrm{p}}(n)$ and $\mathsf{S}^{\nu}_{\mathrm{f}}(n)$.

2.2 Kernel-based dissimilarity measure

The KCD dissimilarity measure is built on the assumption that the sets $\mathsf{X}_{\mathrm{p}}(n)$ and $\mathsf{X}_{\mathrm{f}}(n)$ are similar if and only if the estimated level sets $\mathsf{S}^{\nu}_{\mathrm{p}}(n)$ and $\mathsf{S}^{\nu}_{\mathrm{f}}(n)$ are similar according to some dissimilarity measure. In $\mathcal{X}$, the shape of $\mathsf{S}^{\nu}_{\mathrm{p}}(n)$ and $\mathsf{S}^{\nu}_{\mathrm{f}}(n)$ may be complex and winding – with possibly several non-connected parts – making the definition of a dissimilarity measure difficult. Fortunately, we can use the geometrical interpretation of one-class SVM in $\mathcal{H}$: the level set is included into the intersection of the hypersphere with the hyperplane (a "hypercircle"), see Fig. 1. Thus, comparing the level sets

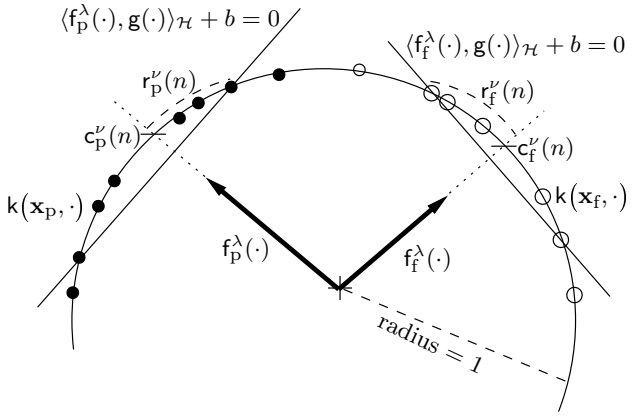

Figure 2: Geometrical interpretation of KCD. The situation plotted corresponds to an abrupt change, because the hyperplane parameterized by $\mathsf{f}_\mathrm{p}^\lambda(\cdot)$ (corresponding to the immediate past set – filled circles) and $\mathsf{f}_\mathrm{f}^\lambda(\cdot)$ (corresponding to the immediate future set – non-filled circles) are well separated, and the distance $\mathsf{d}\big(\mathsf{c}_\mathrm{p}^\nu(n), \mathsf{c}_\mathrm{f}^\nu(n)\big)$ is large compared to the radii $\mathsf{r}_\mathrm{p}^\nu(n)$ and $\mathsf{r}_\mathrm{f}^\nu(n)$, measured along the hypersphere.

$\mathsf{S}_\mathrm{p}^\nu(n)$ and $\mathsf{S}_\mathrm{f}^\nu(n)$ in $\mathcal{X}$ can be performed in $\mathcal{H}$, by comparing the corresponding hypercircles with centers denoted $\mathsf{c}_\mathrm{p}^\nu(n)$ and $\mathsf{c}_\mathrm{f}^\nu(n)$ and radii $\mathsf{r}_\mathrm{p}^\nu(n)$ and $\mathsf{r}_\mathrm{f}^\nu(n)$ see Fig. 2. The radii can be evaluated as the distance between the circle centers and any point onto the hypercircle, such as one of support vectors with $0 < \alpha_i < 1/\nu m$, see Fig. 1. The KCD dissimilarity measure is

$$\mathsf{D}^\nu(n) \;=\; \frac{\mathsf{d}\big(\mathsf{c}_\mathrm{p}^\nu(n), \mathsf{c}_\mathrm{f}^\nu(n)\big)}{\mathsf{r}_\mathrm{p}^\nu(n) + \mathsf{r}_\mathrm{f}^\nu(n)} \tag{7}$$

In the computation of radii as well as the numerator of Eq. (7), the distance used is the *arc distance*, computed along the hypersphere, and which can be computed only in terms dot products in $\mathcal{H}$, that is, kernel functions in $\mathcal{X}$. In practice, $\mathsf{D}^\nu(n)$ is computed easily from the α_i's of each level sets $\mathsf{S}_\mathrm{p}^\nu(n)$ and $\mathsf{S}_\mathrm{f}^\nu(n)$, see [7] for the closed-form expressions. Note that several dissimilarity measures could be easily defined in much the same way [12]. When applied to relevant acoustic features, this dissimilarity enables the segmentation of the time series $\mathbf{x}(n)$, as shown in [7] for music signals, and in Section 4 for speakers.

3. KERNEL SPEAKER CHANGE DETECTION

This section details the full speaker change detection algorithm, specific choices for the acoustic feature vectors, parameters tuning, and comparisons with other techniques.

3.1 Kernel speaker change detection algorithm

The speaker change detection algorithm is as follows. At each time n, repeat the following three steps:

1. **Acoustic features extraction**, see Section 3.4. This can be performed in the same time as the sets $\mathsf{X}_\mathrm{p}(n)$ and $\mathsf{X}_\mathrm{f}(n)$ are formed.
2. **One-class SVM computations**, by solving Problem (6) for both sets — the lower complexity algorithm presented in [11] can also be used, it simply updates the weights α_i's using the fact that, e.g., $\mathsf{X}_\mathrm{p}(n-1)$ and $\mathsf{X}_\mathrm{p}(n)$ are made of the same vectors, up to the newest and oldest ones.
3. **Dissimilarity measure computation**, using Eq. (7), which results in the index $\mathsf{D}^\nu(n)$. As in any other segmentation technique, changes are detected whenever $\mathsf{D}^\nu(n)$ peaks over some threshold, which can be fixed for all times, or by an index search technique.

3.2 Kernel speaker clustering

Once the speaker changes are detected, the next important step is clustering (see [13] and references therein for an overview). It is an iterative agglomerative method that consists of labeling segments of speech, which boundaries have been detected by the previous algorithm, with speaker labels. At each iteration, the algorithm groups the two closest clusters, according to a chosen dissimilarity measure. Here, the metric used to compare the segments is the dissimilarity measure defined in Eq. (7).

The output of hierarchical classification is generally represented by a dendrogram which illustrates the consecutive groupings of clusters. For all the experiments described in section 4, and for both GLR-BIC and KCD algorithms, we used complete linkage as an agglomerative rule because it gave the best results. To form the final set of clusters (the estimated number of speakers), a dendrogram pruning method is needed so as to produce a partition composed of all grouped segments. Several techniques exist in the literature (see [14] and references therein). Here, we chose the simple though efficient technique which consists of cutting the dendrogram at a given height.

3.3 Discussion

An important question about the kernel technique presented here concerns its comparison to standard approaches based on likelihood ratios. Actually, as pointed out by Vapnik [15], it is a much easier task to learn a level set than a full pdf, making the use of one-class SVM based techniques more suited to change detection[4]. Moreover, recent results show that the dissimilarity measure in Eq. (7) admits an interpretation in terms of a modified generalized likelihood ratio test, thanks to the exponential family of distributions [16].

Finally, the one-class SVM methodology enables to implement learning in the RKHS, where the complexity of the learning task scales as m, whatever the dimension $d_\mathcal{X}$ of $\mathcal{X}$ (that is, whatever the dimension of the acoustic features!). Thus, a level set can be learnt from few data, compared to $d_\mathcal{X}$.

3.4 Features selection

The latter property is actually a key advantage of our kernel based approach. In practice, this enables the use of large dimensional acoustic feature vectors. Examples are MFCCs (standard choice), MFCC + LPC and any heterogenous combination of acoustic features vectors (possibly large dimensional). Redundancy is not critical with kernel methods, unlike in the GLR-BIC approach. The results presented in Section 4 show the interest of using many acoustic features, within an inference framework that is insensitive to the dimension of the data.

4. EXPERIMENTS AND RESULTS

In order to evaluate the efficiency of our algorithm, several experiments have been conducted. They are reported in the following subsections.

4.1 Database

All the results reported here were carried out using the NIST-RT'03S data [9]. These data are provided by the NIST evaluation campaign for speaker diarization on American broadcast news. They are divided in two corpora: one of them is used for training the system and tuning parameters (development corpora) and another called (evaluation corpora) is devoted to validation.

[4]See [7] for a thorough discussion about the advantages of the KCD over standard GLR approaches.

Here, we actually used the dry-run files from the RT'02 broadcast news evaluation [6] as development corpora. This development set is composed of six broadcast news records of about 10 min each. The evaluation corpus is composed of three 30 min talk-shows recorded in 2001 from various American channels.

4.2 Experiments on dry-run files

In order to tune the KCD parameters for both speaker change detection and clustering tasks we used the six dry-run files. The experiments are made on each of the six files and the results reported are the average on the whole dry-run files. The evaluation metric used is the NIST evaluation metric called Diarization Error Rate **DER**, (see [6] for details). This metric is clearly described in the NIST-RT'03S evaluation plan [9].

Here, the acoustic parametrization is based on 16 Mel Frequency Cepstral Coefficients. This first parametrization is motivated by its wide use in speaker indexing, in particular for GLR-BIC based methods [1, 2] and [6].

4.2.1 Tuning the KCD parameters

In order to select the best parameters with respect to the DER, several experiments have been conducted, whose results are reported in Fig. 3 and Tab.'s 1-2. We recall that the parameters to be tuned are : $m = m_{\mathrm{p}}{=}m_{\mathrm{f}}$ (length of the immediate past set $\mathsf{X}_{\mathrm{p}}(n)$ and of the immediate future $\mathsf{X}_{\mathrm{f}}(n)$ set, assumed equal), in seconds. In practice, the sets $\mathsf{X}_{\mathrm{p}}(n)$ and $\mathsf{X}_{\mathrm{f}}(n)$ are formed at each time n, and the time step between two adjacent such sets (i.e., between time n and $n+1$) is denoted Δ_n, and is expressed in seconds. Both m and Δ_n are used in the GLR-BIC and in the KCD algorithms, and have the same value for both algorithms, enabling fair comparison. KCD uses the following additional parameters: the kernel parameter σ and the parameter ν which control the fraction of outliers.

Fig. 3 shows the evolution of the DER when the kernel parameter σ increases. The minimum DER error is found at $\sigma = 0.51$, where it equals 10.73%. For this experiment, we choose $m = m_{\mathrm{p}}{=}m_{\mathrm{f}} = 1.5$s and $\Delta_n = 0.2$s, which are the suitable values tested for GLR-BIC methods. The parameter $\nu = 0.1$, meaning that up to 10% of the data are considered as outliers.

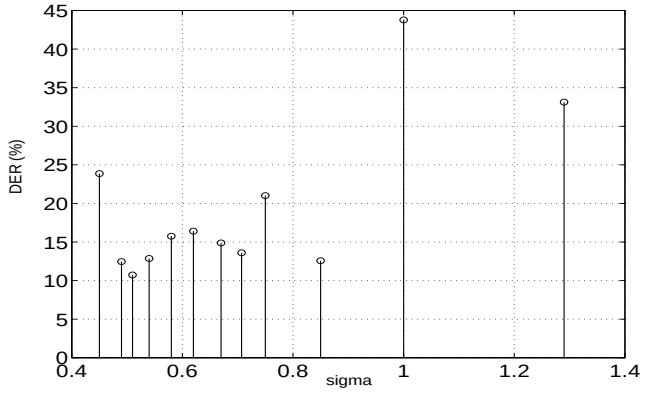

Figure 3: Evolution of the DER for the KCD algorithm when the kernel width parameter σ kernel increases.

While the above parameters (σ and ν) are specific to the KCD algorithm, the following two tables summarize the evolution of the sliding window's length and step increment which are both common parameters for GLR-BIC and KCD based methods. Tables 2 and 3 permit to note that the values of $m = 1.5$s and $\Delta_n = 0.2$s are suitable values.

Table 1: Evolution of DER for the KCD algorithm when m increases for ν=0.1s, σ=0.51 and $\Delta_n = 0.2$s.

m (s)	0.5	0.7	0.9	1.5	2.5	3.5
DER (%)	14.70	13.83	19.21	10.73	16.65	25.36

Table 2: Evolution of the DER for the KCD algorithm as Δ_n increases, for $m = 1.5$s, $\nu = 0.1$ and $\sigma = 0.51$.

Δ_n (s)	0.1	0.2	0.3	0.4	0.5	0.6
DER (%)	24.42	10.73	11.93	15.27	12.85	22.68

As can be seen, the choice of the algorithm parameters must be done carefully, by using some training data. Experiments similar to those presented above may be conducted in order to obtain the optimal parameters. More principled strategies, such as cross-validation, may also be used.

4.2.2 Evaluation on various acoustic feature sets

In this subsection, we evaluate the impact of various acoustic parametrization (described in Table 3) on the algorithm performance. For each configuration, we select the parameters optimized in such way to ensure a minimal DER.

The minimized DER obtained and the corresponding selected parameters with theses acoustic parameterizations, for both GLR-BIC and KCD, are presented in Tab.'s 4–5.

Table 3: Acoustic parameterizations tested with the KCD algorithm.

Configuration	Acoustic vector composition
C_0	16 MFCCs
C_1	16 MFCCs and 10 LPCCs
C_2	C_1 and 10 Linear Reflection coefficients
C_3	C_2 and 10 Filter bank coefficients
C_4	16 MFCCs and 16 ΔMFCCs
C_5	C_4 and 16 $\Delta\Delta$MFCCs

The number of speakers may be estimated by cutting the dendrogram at the height that yields the desired number of speakers. Estimating accurately the number is one of the major goals of Speaker Segmentation. It should be mentionned here that our speaker estimation method is not a fully automated, as explained in [6], but we assume various numbers of speaker, and we evaluate the corresponding DER.

Table 4: Comparison (KCD/GLR-BIC) on various parametrization kinds described in Table 3.

Config	DERmin (%)		estim. # of Spkrs	
	KCD	RVG-BIC	KCD	RVG-BIC
C_0	10.73	26.38	17	9
C_1	8.37	21.18	17	9
C_2	7.95	15.08	17	11
C_3	10.90	15.26	9	9
C_4	11.44	20.91	13	11
C_5	8.63	14.30	19	11

Table 5: Selected parameters on various parametrizations

Configuration	Selected parameters
C_0	$m = 1.5$s, $\sigma = 0.51$, $\Delta_n = 0.2$s.
C_1	$m = 2.0$s, $\sigma = 1$, $\Delta_n = 0.3$s
C_2	$m = 1.5$s, $\sigma = 1$, $\Delta_n = 0.3$s
C_3	$m = 2.5$s, $\sigma = 0.707$, $\Delta_n = 0.2$s
C_4	$m = 0.7$s, $\sigma = 0.707$, $\Delta_n = 0.2$s
C_5	$m = 0.9$s, $\sigma = 0.85$, $\Delta_n = 0.3$s

Table 6: Results obtained on the evaluation corpora. The KCD algorithm parameters are $m = 1.5$s, $\nu = 0.1$, $\sigma = 1$ and $\Delta_n = 0.3$s. The parametrization selected is C_2. Files are (a) 20010228.2100-2200-MNB-NBW, (b) 20010217.1000-1030-VOA-ENG and (c) 20010220.2000-2100-PRI-TWD.

File	GLR-BIC	KCD	Estimated # of Speakers
(a)	25.60	14.34	23
(b)	20.17	12.28	25
(c)	22.69	14.27	22

As also explained in [6], we can estimate an optimal number of speaker as the value that minimizes the diarization error. In this context, the parametrization selected should be C_2 corresponding to a DER of 7.95%.

4.3 Validation on evaluation files

This section presents the results obtained on the three evaluation files for both our KCD algorithm and the BIC-GLR based method. Based on the previous experiments, the parameters selected for KCD are $m = 1.5$s, $\nu = 0.1$, $\sigma = 1$ and $\Delta_n = 0.3$s. The parametrization selected is C_2. Table 6 shows that our method outperforms the GLR-BIC based approach. The KCD best result averaged over the three files is DER=13.63%. The same parameter values for m and Δ_n are selected for GLR-BIC and KCD. In fact, the results for KCD displayed in Table 6 should be compared to those reported in [6] (page 22, Table 7), where state-of-the art results are given for a number of other methods and where the method-specific parameters are fine optimized, independently of KCD.

5. CONCLUSIONS AND FUTURE WORK

The results reported above show that our new KCD algorithm opens a promising research direction towards better speaker segmentation algorithms. It is shown through several experiments that our method outperforms the standard GLR-BIC method. In this context it is important to note that our results must be compared to those published in [6] where state-of-the art results are given for a number of other methods and where the method-specific parameters are fine optimized, independently of KCD. In particular, our algorithm being insensitive to the dimension of the acoustic features, we can envisage the use of more informative, larger dimensional acoustic features. It is also shown that our algorithm is less sensitive to the acoustic feature redundancy than the standard method. Further work will be devoted to a better adequation of the acoustic features to the KCD approach and to some prior processing to be implemented before performing speaker segmentation (as other acoustic macro-class segmentation methods do).

REFERENCES

[1] D. Moraru, S. Meignier, C. Fredouille, L. Besacier, and J.-F. Bonastre, "The ELISA consortium approaches in broadcast news speaker segmentation during the NIST 2003 rich transcription evaluation", in *IEEE ICASSP'04*, Montreal, Canada, 2004.

[2] S. Kwon and S. Narayanan, "Unsupervised speaker indexing using generic models", *IEEE Trans. Speech Audio Proc.*, vol. 13, no. 5, pp. 1004–1013, Sept. 2005.

[3] P. Delacourt and C. Wellekens, "Distbic: a speaker-based segmentation for audio data indexing", *Speech Communication*, vol. 32, no. 1, pp. 111–126, Sept. 2000.

[4] S. Meignier et al., "Evolutive hmm for speaker tracking systems.", in *IEEE ICASSP'00*, Istambul, Turkey, 2000, pp. 1177–1180.

[5] S. Meignier, *Indexation en locuteurs de documents sonores: Segmentation d'un document et Appariement d'une collection*, Ph.D. thesis, Universite d'Avignon et des pays de vaucluse, Laboratoire Informatique d'Avignon, 2002.

[6] S. Meignier et al., "Step-by-step and integrated approaches in broadcast news speaker diarization", *Computer Speech and Language* , pp. 1–28, 2005, in Press.

[7] F. Desobry, M. Davy, and C. Doncarli, "An online kernel change detection algorithm", *IEEE Trans. Sig. Proc.*, vol. 53, no. 5, Aug. 2005.

[8] D. Srikrishna Satish V. kartik and C. Chndra Sekhar, "Speaker change detection using support vector machines", in *NOLISP'05*, Barcelona, Spain, 2005.

[9] NIST RT03S, "The rich transcription spring 2003 (rt-03s) evaluation plan http://www.nist.gov/speech/tests/rt/rt2003/spring/docs/rt-03-spring-eval-plan-v4.pdf,", 2003.

[10] B. Schölkopf and A. Smola, *Learning with Kernels*, MIT Press, Cambridge, USA, 2002.

[11] M. Davy, F. Desobry, A. Gretton, and C. Doncarli, "An online support vector machine for abnormal events detection", *Signal Processing*, 2006, to appear.

[12] F. Desobry and M. Davy, "Dissimilarity measures in feature space", in *IEEE ICASSP'04*, Montreal, Canada, 2004.

[13] D. Liu and F. Kubala, "Online speaker clustering", in *IEEE ICASSP'04*, Montreal, Canada, 2004.

[14] S. Meignier, J.-F. Bonastre, and Ivan Magrin-Chagnolleau, "Speaker utterances tying among speaker segmented audio documents using hierarchical classification: towards speaker indexing of audio databases", in *ICSLP 2002*, Denver, Colorado, United States of America, September 2002, 2002, vol. 1, pp. 573–576.

[15] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, USA, 1998.

[16] S. Canu and A. Smola, "Kernel methods and the exponential family", in *ESANN'05*, Brugge, Belgium, May 2005.