POLITECNICO DI TORINO
Repository ISTITUZIONALE

Large scale training of Pairwise Support Vector Machines for speaker recognition

Original
Large scale training of Pairwise Support Vector Machines for speaker recognition / Cumani, Sandro; Laface, Pietro. - In: IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING. - ISSN 2329-9290. - STAMPA. - 22:11(2014), pp. 1590-1600. [10.1109/TASLP.2014.2341914]

Availability:
This version is available at: 11583/2555345 since:

Publisher:
IEEE - INST ELECTRICAL ELECTRONICS ENGINEERS INC

Published
DOI:10.1109/TASLP.2014.2341914

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

12 October 2024

Large scale training of Pairwise Support Vector Machines for speaker recognition

Sandro Cumani and Pietro Laface

Abstract

State–of–the–art systems for text–independent speaker recognition use as their features a compact representation of a speaker utterance, known as "i–vector". We recently presented an efficient approach for training a Pairwise Support Vector Machine (PSVM) with a suitable kernel for i–vector pairs for a quite large speaker recognition task. Rather than estimating an SVM model per speaker, according to the "one versus all" discriminative paradigm, the PSVM approach classifies a trial, consisting of a pair of i–vectors, as belonging or not to the same speaker class. Training a PSVM with large amount of data, however, is a memory and computational expensive task, because the number of training pairs grows quadratically with the number of training i–vectors. This paper demonstrates that a very small subset of the training pairs is necessary to train the original PSVM model, and proposes two approaches that allow discarding most of the training pairs that are not essential, without harming the accuracy of the model. This allows dramatically reducing the memory and computational resources needed for training, which becomes feasible with large datasets including many speakers. We have assessed these approaches on the extended core conditions of the NIST 2012 Speaker Recognition Evaluation. Our results show that the accuracy of the PSVM trained with a sufficient number of speakers is 10-30% better compared to the one obtained by a PLDA model, depending on the testing conditions. Since the PSVM accuracy increases with the training set size, but PSVM training does not scale well for large numbers of speakers, our selection techniques become relevant for training accurate discriminative classifiers.

Index Terms

Speaker recognition, i–vector, PLDA, Support Vectors, Pairwise Support Vector Machines

I. Introduction

Speaker recognition, like face recognition, is a peculiar classification task in which the number of classes is usually large (possibly an open set), and the number of samples per class is often small. Thus, in most state–of–the–art systems, speaker recognition is formulated as the solution of a classification problem where a pair of patterns - each representing a unknown speaker utterance - is scored to verify the hypothesis that the two utterances belong to the same speaker. Probabilistic Linear Discriminant Analysis (PLDA) [1], [2], in combination with i–vectors [3], a compact representation of a Gaussian Mixture Model (GMM) supervector [4], is the current reference for this speaker recognition paradigm. A successful alternative to the generative PLDA model has been recently presented in [5], [6]: a new discriminative SVM model, where a single pairwise SVM (PSVM) is trained to classify a trial – composed of two i–vectors – as belonging to the "same speaker", or to the "different speaker" class. This is in contrast with the usual "one-versus-all" framework, where an SVM model is created for each enrolled speaker, using as samples of the impostor class the i–vectors of a background cohort of speakers. The PSVM approach avoids the major weakness of "one-versus-all" SVM training, namely the scarcity of available samples for the target speaker, which can easily reduce to just one.

Although our pairwise SVM training implementation is extremely efficient compared to a naïve approach, it is nevertheless expensive in terms of computational resources, which grow quadratically with the number of the training i–vectors. In [7] we introduced a simple and effective technique for discarding the i–vector pairs that are not essential for training a pairwise SVM. The selection of the i–vector pairs was obtained by exploiting the scores of a PLDA model on the training data. This work revises and extends this approach, introducing an additional

The authors are with the Dipartimento di Automatica e Informatica, Politecnico di Torino, 10143 Torino, Italy (e-mail: sandro.cumani@polito.it, pietro.laface@polito.it).

Computational resources for this work were provided by HPC@POLITO (http://www.hpc.polito.it).

This paper is an extended and revised version of paper [7] presented at ICASSP 2104.

technique. Using our selection procedures, PSVM training with datasets including a large number of speakers becomes feasible in terms of memory and computation costs. The success of these techniques depends on the possibility of appropriately selecting a small subset of the training pairs. We theoretically prove that the number of training pairs necessary for obtaining an accurate PSVM model is indeed a very small fraction of the number of training set pairs, and moreover that it grows only linearly with the number of speakers. We also experimentally show that simple strategies allow selecting a small subset of pairs containing most of the essential patterns.

In order to illustrate our approaches, it is worth recalling that the solution of the SVM optimization problem, in its dual formulation [8], [9], shows that the separation hyperplane is a function only of a subset of training patterns, the so-called Support Vectors (SVs). Since it is known that the number of SVs increases linearly with the number of training data [10], several solutions have been proposed in the past trying to detect and discard from the training set the patterns that do not belong to the SV set. Our approaches follow this data selection framework. Among the numerous data selection techniques that have been proposed for binary SVMs, the ones that best fit our problem are presented in [11], [12], [13], but are computationally very expensive. In [11] a cross–training approach is proposed where the training data are split into non–overlapping subsets, which are used for training independent SVMs. The training patterns close to the average margin hyperplane are selected for training the final SVM. This approach is interesting because the training procedure can be performed in parallel on each subset, but it has several drawbacks. Not only it is difficult to select meaningful non–overlapping subsets of i–vector pairs, but also this technique remains expensive for a large speaker set, and does not offer any guarantee that the average margin hyperplane is similar to the optimal hyperplane. Hierarchical parallel training is proposed in the cascade SVM approach of [12], which is, however, even more expensive than the former because all the training patterns have to be scored by each SVM in the tree, and also because the procedure is iterative. In [13], a pre–processing step is required that creates a nearest neighbor structure of the training instances is proposed. Again, this approach is extremely slow for large datasets, as also experimentally shown in [14].

All these approaches try to reduce the number of SVs for a generic SVM binary problem, but the problem that we address is characterized by a small number of utterances, pronounced by a large number of speakers, and by two highly unbalanced classes. The number of pairs for the "same speaker" class is $\sum_{i=1}^{N} r_i^2$, where N and r_i are the number of speakers and the number of utterances of speaker i, respectively. The total number of pairs is, instead, $\left(\sum_{i=1}^{N} r_i\right)^2 = n^2$, i.e., it grows quadratically with the number of i–vectors n. Thus, also the number of "different speaker" pairs grows quadratically.

The main contribution of this work is the proof that the number of SVs of a PSVM increases linearly with respect to the number of training i–vectors, rather than quadratically as the number of i–vector pairs. By appropriately selecting a small subset of the complete set of pairs, we are able to keep most of the SVs, obtaining an accurate model while saving memory and computation time.

Our first selection strategy relies on the observation that the time and the amount of memory required for training a generative PLDA model grow only linearly with the number of the i–vectors. Thus, we first train a PLDA model with all the i–vectors, then the scores of all the training pairs are computed using this PLDA model, and all pairs with a score lower than a threshold are removed from the training list of the PSVM. The rationale for this approach is that a good correlation exists between the PLDA and the PSVM scores. Thus, "different speaker" i–vector pairs with large negative PLDA scores would lie in the correct class region, and far from the margin hyperplane. These pairs can be discarded without modifying the optimal solution of the PSVM because with high probability they are not SVs.

We compared this PLDA–based strategy for selecting the subset of i–vector pairs that include with high probability the SVs of the complete training set with the simplest and fastest approach based on random selection of the pairs [15]. In particular, we keep all the "same speaker" pairs and a random selection of the "different speaker" pairs. Since a PSVM model trained with a small set of pairs selected according to this technique performed surprisingly well, we replaced the PLDA model used for pair selection with a PSVM trained with the randomly selected pairs. This strategy, thus, is again based on two models, the first one is an approximate PSVM trained with random pairs, the final model is trained with the pairs selected according to the scores of the approximate PSVM. This technique is a simplified, yet effective, version of an approach proposed for SVMs several years ago in [16], and stemming back to the work devoted to speeding–up k–Nearest Neighbor classification of [17].

Several variations of the random selection approach have been proposed in the SVM literature, e.g., [18], [19], [20] and many others referred in [21], [14]. All these techniques aim at finding a solution that converges to the

optimal hyperplane. However, since they usually require expensive iterations to finally obtain the reduced set of training patterns, they are not suited to a very large set of pairs. The Simpler Core Vector Machines (SCVM) is an interesting approach [22], [23], which is similar to the Cutting Plane algorithm used in SVM^{Perf} [24]. Its drawback is that for very large datasets, the size of the core-set remains huge. Indeed, SCVM has been tested only for a few millions of patterns. As illustrated in Section III, we use the Optimized Cutting Plane Algorithm (OCAS) [25], [26] as a primal solver for training a PSVM. Thus our strategy, based on data pre–selection and OCAS, can be seen as an optimization and speedup of the SCVM technique.

Since the reduced set of training i–vectors is very small compared to the full set, our selection strategies are extremely efficient, allowing PSVM training with billions of pairs. The achieved reduction of the memory and computational complexity of PSVM training, however, does not affect the classification accuracy, as shown by our results on the extended core conditions of the NIST 2012 Speaker Recognition Evaluation (SRE).

The outline of the paper is as follows: Section II formulates the score of an i–vector pair, which is the key computation both in SVM training and testing, as a second order Taylor approximation of a symmetric score function. Section III recalls the PSVM classifier, and its efficient training procedure. In Section IV we detail our selection strategies. Section V presents the experimental results, comparing the performance of the PLDA and of the PSVM models trained with selected subsets of pairs. Our conclusions are drawn in Section VI.

II. Taylor approximation of an i–vector pair score

We recall our interpretation in [5] of the PSVM score as a second order Taylor series approximation to any analytic symmetric scoring function $s(\mathbf{\Phi})$ of the i–vector pair $\mathbf{\Phi} = (\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$. This formulation is here recalled for introducing the SVM hyper–parameters that must be estimated, and for illustrating the changes necessary for training a PSVM with a reduced dataset.

The Taylor expansion for s, around a point $\hat{\mathbf{\Phi}}$, is:

$$s(\mathbf{\Phi}) = \sum_{k=0}^{+\infty} \frac{\left(\left(\mathbf{\Phi} - \hat{\mathbf{\Phi}}\right) \cdot \boldsymbol{\nabla}\right)^k s|_{\hat{\mathbf{\Phi}}}}{k!} \ . \tag{1}$$

Without loss of generality (see Section III-E of [5]), we consider the second order Taylor expansion for $s(\mathbf{\Phi})$ around $\hat{\mathbf{\Phi}} = \mathbf{0}$, which is:

$$\hat{s}(\mathbf{\Phi}) = s(\hat{\mathbf{\Phi}}) + (\mathbf{\Phi} \cdot \boldsymbol{\nabla} s|_{\hat{\mathbf{\Phi}}}) + \mathbf{\Phi}^T (\mathbf{H}(s)|_{\hat{\mathbf{\Phi}}}) \mathbf{\Phi} \ , \tag{2}$$

where $\boldsymbol{\nabla}$ is the vector of differential operators

$$\boldsymbol{\nabla} = \left(\frac{\partial}{\partial \Phi_1}, \ldots, \frac{\partial}{\partial \Phi_d} \right) \ ,$$

d is the dimension of the i–vector pair, and $\mathbf{H}(s)$ is half the Hessian of function $s(\mathbf{\Phi})$.

Defining [5]:

$$s(\hat{\mathbf{\Phi}}) = k \ , \quad \boldsymbol{\nabla} s|_{\hat{\mathbf{\Phi}}} = [\, \mathbf{c} \quad \mathbf{c} \,] \ , \quad \mathbf{H}(s)|_{\hat{\mathbf{\Phi}}} = \begin{bmatrix} \mathbf{\Gamma} & \mathbf{\Lambda} \\ \mathbf{\Lambda} & \mathbf{\Gamma} \end{bmatrix} \ , \tag{3}$$

with a symmetric $\mathbf{\Lambda}$, we obtain the quadratic function of the i–vector pair:

$$\begin{aligned} \hat{s}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) \quad = \quad & \boldsymbol{\phi}_1^T \mathbf{\Lambda} \boldsymbol{\phi}_2 + \boldsymbol{\phi}_2^T \mathbf{\Lambda} \boldsymbol{\phi}_1 + \boldsymbol{\phi}_1^T \mathbf{\Gamma} \boldsymbol{\phi}_1 + \boldsymbol{\phi}_2^T \mathbf{\Gamma} \boldsymbol{\phi}_2 \\ & + (\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2)^T \mathbf{c} + k \ . \end{aligned} \tag{4}$$

It is worth noting that the structure imposed by (3) arises naturally from the symmetry of the score function $s(\mathbf{\Phi})$, from the symmetry of the expansion point $\hat{\mathbf{\Phi}}$, and from the symmetry of the Hessian. Equation (4) can be also interpreted as a linear function of the hyper–parameter set $\Theta = \{\mathbf{\Lambda}, \mathbf{\Gamma}, \mathbf{c}, k\}$ in an expanded space of i–vector pairs. In particular, $\hat{s}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ can be written as the dot–product of a vector of weights $\mathbf{w}$ (the model hyper–parameters) and an expanded vector $\varphi(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ representing the i–vector pair:

$$\hat{s}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \mathbf{w}^T \varphi(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) \ . \tag{5}$$

The parameters and the expanded i–vector pair are:

$$\mathbf{w} = \begin{bmatrix} \mathrm{vec}(\mathbf{\Lambda}) \\ \mathrm{vec}(\mathbf{\Gamma}) \\ \mathbf{c} \\ k \end{bmatrix}, \text{ and } \varphi(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \begin{bmatrix} \mathrm{vec}(\boldsymbol{\phi}_1\boldsymbol{\phi}_2^T + \boldsymbol{\phi}_2\boldsymbol{\phi}_1^T) \\ \mathrm{vec}(\boldsymbol{\phi}_1\boldsymbol{\phi}_1^T + \boldsymbol{\phi}_2\boldsymbol{\phi}_2^T) \\ \boldsymbol{\phi}_1 + \boldsymbol{\phi}_2 \\ 1 \end{bmatrix}, \tag{6}$$

where $\mathrm{vec}(\cdot)$ is the operator that stacks the columns of a matrix into a vector.

III. Pairwise SVM training

Our formulation is equivalent to a second degree inhomogeneous polynomial kernel SVM [5], which could be trained by using a dual solver [18], [27], but this approach has severe limitations for large training sets. Caching the complete kernel matrix is impractical even for relatively small datasets because it would require $O(n^4)$ memory, where n is the number of i–vectors. Also ineffective are the alternatives of keeping in memory the complete dataset of mapped features ($O(n^2d^2)$, where d is the i–vector dimension) or expanding the features on–line, with a computational complexity $O(n^2d^2)$ for each iteration. Considering, for example, that number of training i–vectors in NIST 2010 SRE dataset is approximately n=20000, and that a popular setting for the i–vector dimension d=400, a standard SVM dual solver for estimating the hyperplane $\mathbf{w}$ would be impractical. Pairwise SVM training using polynomial kernel SVMs has been also proposed in [28] for medium–size training sets. It has been shown in [5], [28] that the kernel formulation depends only on i–vector dot–products, the kernel, thus, can be efficiently computed by caching the i–vector Gram–matrix, however, the memory cost remains huge, growing as $O(n^2)$.

We use a primal solver because it has been shown in [5] that it is possible to efficiently evaluate the loss function and its gradient with respect to $\mathbf{w}$ over the set of all training pairs, in $O(n^2d) + O(nd^2)$ time, without the need of expanding the i–vectors. An analysis of large-scale SVM training algorithms suited to speaker recognition tasks [29] allowed us to select, among the primal solvers, the Optimized Cutting Plane Algorithm (OCAS) approach proposed in [26], [30], which offers a general framework for solving convex unconstrained regularized risk minimization problems. Although the memory cost remains $O(n^2)$ even for a primal solver, the i–vector pair selection strategies, illustrated in the next section, allow memory complexity to be dramatically reduced, making PSVM training feasible even with very large scale training sets, whereas a dual solver cannot take advantage of the pair selection techniques, as highlighted in Section IV.

A. PSVM training by means of a primal solver

In its primal form, the SVM optimization can be seen as the solution of the unconstrained convex regularized risk minimization problem:

$$E(\mathbf{w}) = \arg\min_{\mathbf{w}} \frac{1}{2}\lambda \|\mathbf{w}\|^2 + \frac{1}{p}\sum_{i=1}^{p} \ell(\mathbf{w}, \mathbf{x_i}, \zeta_i) \tag{7}$$

where, in the PSVM framework, p is the number of training pairs, $\mathbf{x}_i \in \mathcal{X}$ denotes a training pattern consisting of a pair of (d–dimensional) i–vectors with associated label $\zeta_i \in \{-1, +1\}$, and λ is a regularization factor. The second term in this expression is the empirical risk evaluated on the training set, whereas the first term — the squared $L2$ norm of the separating hyperplane $\mathbf{w}$ — is a regularization contribution, which is related to the generalization capability of the model [31]. The regularization factor λ allows tuning the trade–off between the margin and the empirical risk. The latter is the sum of so–called hinge (L1) loss function:

$$\ell_{L1}(\mathbf{w}, \mathbf{x_i}, \zeta_i) = \max(0, 1 - \zeta_i \mathbf{w}^T \mathbf{x}_i) \ . \tag{8}$$

It is worth noting that all possible i–vector pairs are considered, including the pairs $(\boldsymbol{\phi}_i, \boldsymbol{\phi}_i)$, though they should not lie inside the margin, and the symmetric pairs $(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$ and $(\boldsymbol{\phi}_j, \boldsymbol{\phi}_i)$. This allows the score symmetry constraints in (3) to be implicitly satisfied.

Using OCAS, the SVM parameters $\mathbf{w}$ are optimized by evaluating the loss function and the sub–gradients of its objective function (7). These evaluations require, in principle, a sum over all the expanded i–vector pairs in the training set. Since their number is n^2, which can easily reach the order of hundred of millions for typical training sets, these evaluations would not be effective or even feasible because their complexity would be $O(n^2d^2)$. However,

both the objective function and its gradient can efficiently be computed from (4). In particular, let $\mathbf{D} = [\boldsymbol{\phi}_1 \boldsymbol{\phi}_2 \ldots \boldsymbol{\phi}_n]$ be a $d \times n$ matrix including n i–vectors, and let $\mathbf{S}_{\theta i,j} = \mathbf{S}_\theta(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$ denote the score matrix for all possible pairs related to component θ of $\mathbf{w}$, where $\theta \in \{\boldsymbol{\Lambda}, \boldsymbol{\Gamma}, \mathbf{c}, k\}$. From (5) and (4) the score matrices can be evaluated as:

$$\begin{aligned} S_{\boldsymbol{\Lambda}}(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) &= \boldsymbol{\phi}_i^T \boldsymbol{\Lambda} \boldsymbol{\phi}_j + \boldsymbol{\phi}_j^T \boldsymbol{\Lambda} \boldsymbol{\phi}_i \Rightarrow \mathbf{S}_{\boldsymbol{\Lambda}} = 2\ \mathbf{D}^T \boldsymbol{\Lambda}\, \mathbf{D} \\ S_{\boldsymbol{\Gamma}}(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) &= \boldsymbol{\phi}_i^T \boldsymbol{\Gamma} \boldsymbol{\phi}_i + \boldsymbol{\phi}_j^T \boldsymbol{\Gamma} \boldsymbol{\phi}_j \Rightarrow \mathbf{S}_{\boldsymbol{\Gamma}} = \tilde{\mathbf{S}}_{\boldsymbol{\Gamma}} + \tilde{\mathbf{S}}_{\boldsymbol{\Gamma}}^T \\ S_c(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) &= \mathbf{c}^T (\boldsymbol{\phi}_i + \boldsymbol{\phi}_j) \qquad \Rightarrow \mathbf{S}_c = \tilde{\mathbf{S}}_c + \tilde{\mathbf{S}}_c^T \\ S_k(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) &= k \qquad\qquad\qquad\quad \Rightarrow \mathbf{S}_k = k \cdot \mathbf{1}\ , \end{aligned} \tag{9}$$

where

$$\tilde{\mathbf{S}}_{\boldsymbol{\Gamma}} = [\underbrace{d_{\boldsymbol{\Gamma}} \ldots d_{\boldsymbol{\Gamma}}}_{n}] \qquad \tilde{\mathbf{S}}_c = [\underbrace{d_c \ldots d_c}_{n}]\ , \tag{10}$$

and

$$d_{\boldsymbol{\Gamma}} = \mathrm{diag}(\mathbf{D}^T \boldsymbol{\Gamma} \mathbf{D}) \qquad d_c = \mathbf{D}^T \mathbf{c}\ . \tag{11}$$

The $\mathrm{diag}(\cdot)$ operator returns the diagonal of a matrix as a column vector, and $\mathbf{1}$ is an $n \times n$ matrix of ones. No explicit expansion of i–vectors is therefore necessary for evaluating the scores.

Denoting by $\mathbf{S} = \mathbf{S}_{\boldsymbol{\Lambda}} + \mathbf{S}_{\boldsymbol{\Gamma}} + \mathbf{S}_{\mathbf{c}} + \mathbf{S}_k$ the sum of the partial score matrices, the SVM loss function can be obtained as:

$$\begin{aligned} \mathcal{L}_{L1}(\mathbf{D}, \mathbf{Z}) &= \sum_{i,j} \max[0, 1 - \zeta_{i,j} \mathbf{w}^T \varphi(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)] \\ &= \langle \mathbf{1}, \max[\mathbf{0}, \mathbf{1} - (\mathbf{Z} \circ \mathbf{S})] \rangle\ , \end{aligned} \tag{12}$$

where $\mathbf{1}$ and $\mathbf{0}$ are $n \times n$ matrices of ones and of zeros, respectively, $\mathbf{Z}$ is the $n \times n$ matrix of the labels $\zeta_{i,j}$ for each i–vector pair $(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$, $\circ$ is the element-wise matrix multiplication operator, and $\langle \cdot, \cdot \rangle$ denotes the inner product of its arguments.

In order to compute the objective function gradient, let $g_{i,j}$ be the derivative of the hinge loss function with respect to the score $s_{i,j} = \mathbf{w}^T \varphi(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$:

$$g_{i,j} = \begin{cases} 0 & \text{if } \zeta_{i,j} s_{i,j} \geq 1 \\ -\zeta_{i,j} & \text{otherwise}\ , \end{cases} \tag{13}$$

and let $\mathbf{G}$ be the matrix of the elements $g_{i,j}$. Recalling that $\mathbf{G}$ is symmetric, the terms of the sub–gradient of the loss function can be rewritten in terms of dot–products and element–wise matrix products as:

$$\nabla \mathcal{L}_{\mathbf{w}} = \begin{bmatrix} 2\ \mathrm{vec}\left(\mathbf{D}\mathbf{G}\mathbf{D}^T\right) \\ 2\ \mathrm{vec}\left(\ [\mathbf{D} \circ (\mathbf{1_A}\ \mathbf{G})]\, \mathbf{D}^T\right) \\ 2\ \ [\mathbf{D} \circ (\mathbf{1_A}\ \mathbf{G})]\ \ \mathbf{1_B} \\ \mathbf{1_B}^T\ \mathbf{G}\ \mathbf{1_B} \end{bmatrix}\ , \tag{14}$$

where $\mathbf{1_A}$ is a $d \times n$ matrix of ones, and $\mathbf{1_B}$ is an n–dimensional column vector of ones. Again, no explicit expansion of i–vectors is necessary for this evaluation.

Due to the small size of the i–vectors, the dataset of training utterances can easily be loaded in main memory. The evaluation of loss functions and gradients in OCAS, thus, requires matrix–by–matrix multiplications of large matrices ($n \times n$), which can also be kept in main memory if n is not too large. For a very large training set, these computations can be performed through block decomposition of the matrices, but this procedure would be cumbersome and slow.

B. PSVM training with a reduced set of i–vector pairs

The selection strategies illustrated in the next section solve these memory issues, and allow training a pairwise SVM even with millions of i–vectors of a very large set of speakers, i.e, billions of i–vector pairs. Using a reduced training set, the computation of the scores, loss function, and gradients is modified as follows:

- the partial scores in (9), and the loss function (12), are computed only for the selected pairs,
- the elements of matrix $\mathbf{G}$, corresponding to discarded pairs are set to 0 in (14). Since their number is very large, a sparse representation of matrix $\mathbf{G}$ is used.

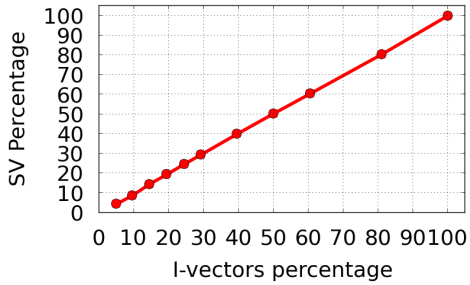


Fig. 1: Percentage of SVs as a function of the percentage of the training i–vectors.

IV. I–VECTOR PAIR SELECTION

Although a dual solver is not used by our approach, we will make use of the definition of Support Vector to motivate our selection strategies. The main theoretical contribution of this work is given in the Appendix, where we demonstrate that the number of SVs is not simply bounded by the number of i–vector pairs, but by a linear function of the number of the "same speaker" pairs. Since these latter are a very small fraction of the complete set of the pairs, it is possible to devise appropriate selection strategies that dramatically reduce the set of i–vector pairs to consider for training a PSVM.

A. Number of support vectors

In Appendix we give a proof that, for a generic binary SVM, the maximum number of bounded SVs, i.e., of patterns lying inside the margin, cannot be greater than two times the number of training patterns of the less populated class. This proof has not practical interest for training a generic SVM with a balanced number of patterns per class, because it trivially states that the number of SVs is less than the training patterns. However, our proof is critical for training a PSVM for speaker recognition, which is characterized by two highly unbalanced classes. Since the training set typically includes a quite large number of speakers, each providing a small number of utterances, the less populated class is the "same speaker" class, which grows linearly with the number of speakers rather than quadratically The number of unbounded SVs, i.e., of SVs lying on the margin, is less relevant because we also demonstrate that it with the number of i–vector pairs. Thus, the bounded SVs of a PSVM are a very small fraction of the total number of i–vector pairs. can be bounded to $d + 1$, where d is the dimension of the patterns.

Figure 1 plots the number of SVs obtained by training a PSVM with the i–vectors of an increasing number of speakers, as percentages relative to the PSVM trained with the complete set of speakers. These i–vectors are selected from the training dataset used for the NIST 2012 evaluation described in Section V. The complete set includes n = 48568 i–vectors, corresponding to approximately 2.3 billion "different speaker", and T = 979540 "same speaker" i–vector pairs. The plot confirms our theoretical claim that the number of SVs grows linearly with the number of i–vectors (n) rather than with the number of the training set pairs (n^2).

Computing the PSVM score s of each i–vector pair of the complete training set, we obtained the distribution of the number of the "same speaker" and "different speaker" i–vectors in the regions defined by the optimal hyperplane. The distribution is given in Table I, where the number of SVs in each region, and for each class, is shown in bold. It is worth noting that the total number of SVs is a small fraction of the total number of i–vector pairs, and that although the number of "different speaker" pairs is much greater than the number of "same speaker" pairs, this is not the case for the number of the corresponding SVs, which is, insted, comparable.

On the basis of this evidence we devised two simple strategies for detecting and discarding a huge set of i–vectors that would not contribute to the training of the PSVM because they have few chances to be SVs.

B. PLDA–based i–vector pair selection

The first selection strategy is based on two working hypotheses. The first one is that the scores obtained by a generative PLDA model and the scores obtained by a PSVM on the same data used for training are correlated. Indeed, the correlation coefficient between the PLDA and the PSVM scores for the "same speaker" and "different

TABLE I: Distribution of the number of "same speaker" and "different speaker" i–vector pairs in the regions defined by the optimal hyperplane margins. The number of SVs is shown in bold.

i–vector pairs	Score			
	$s < -1$	$-1 \leq s \leq 0$	$0 < s \leq 1$	$s > 1$
"same speaker"	**4381**	**165633**	**296234**	513292
"different speaker"	2.3G	**461947**	**9114**	**1248**

(a) (b)

Fig. 2: (a) Percentage of i–vector pairs that are included in the PSVM training set, and percentage of lost support vectors, as a function of the PLDA score threshold. (b) Zoom of a region of (a)

speaker" pairs of the above mentioned training set is 0.83 and 0.69, respectively. The correlation is clearly not complete, otherwise PSVM would not perform differently than PLDA. The second hypothesis is that we can a–priori identify and discard most of the i–vector pairs that do not contribute to the set of the PSVM SVs. Figure 2 shows the percentage of i–vector pairs that would be selected for PSVM training, and the percentage of SVs that would be discarded, as a function of a threshold on the PLDA scores. By keeping a small fraction of the complete training set, the filtered training set includes almost all the SVs of the complete training set. This implies that the PSVM solution obtained with the small subset of selected training pairs will need much less memory and processing time, but will be nevertheless near optimal.

In the PLDA–based selection approach, thus, the complete set of training pairs is classified using a PLDA model trained on the same data, and the K–best scoring pairs are selected. This is an expensive $O(n^2)$ computational step, but it is performed only once, and can be efficiently performed by using a double–ended priority queue algorithm [32], which requires a single sweep of the PLDA scores, in $O(n^2 \log K)$. Since we know that the number of SVs grows linearly with the number T of "same speaker" pairs, the parameter K is chosen to be a multiple k of their number, $K = kT$. By keeping the K–best scoring pairs, we eliminate the pairs with large negative PLDA scores, which with high probability would not contribute to the set of the SVs. The "same speaker" pairs with large positive PLDA scores could be discarded as well, but they are kept, neglecting the value of their PLDA scores, because their number grows linearly, and is much lower than the number of the "different speaker" pairs.

Since PLDA is currently the state–of–the–art technique in text–independent speaker recognition, it was natural to train a PLDA model, not only for exploiting the PLDA scores for the selection of the pairs, but also to obtain a reference for the performance of the PSVM models. The selection of the i–vector pairs, however, can also been performed by means of a different classifier, or just by random selection [15].

TABLE II: Performance of gender independent PLDA, and PSVM models trained with the complete and with the filtered training set, on NIST 2012 extended core evaluation. Last row gives the % performance improvement of the FSVM models with respect to PLDA.

Model	Condition 1 interview without added noise			Condition 2 phone call without added noise			Condition 3 interview with added noise			Condition 4 phone call with added noise			Condition 5 phone call from a noisy environment		
	EER	DCF08	Cprim	EER	DCF08	Cprim	EER	DCF08	Cprim	EER	DCF08	Cprim	EER	DCF08	Cprim
PLDA	3.76	0.160	0.323	2.52	0.124	0.336	2.94	0.108	0.239	5.43	0.231	0.476	2.99	0.147	0.380
PSVM	2.71	0.108	0.259	2.35	0.116	0.300	2.18	0.082	0.200	4.39	0.193	0.430	2.94	0.135	0.341
FSVM	2.66	0.108	0.260	2.37	0.117	0.302	2.13	0.082	0.200	4.40	0.191	0.430	2.94	0.136	0.342
% impr.	29.3	32.5	19.5	5.9	5.6	10.1	27.5	24.1	16.3	19.0	17.3	9.7	1.7	7.5	10.0

C. PSVM–based i–vector pair selection

We compared the performance of the reference PLDA model, trained with the complete training set, with the performance of a set of PSVMs trained with an increasing number of i–vectors, either selected from the PLDA scores or randomly. Random selection is the simplest and fastest selection approach, because it does not incur the overhead of the computation of the complete set of i–vector pair scores. As will be shown in Figure 4 of Section V-B, the results of the PSVM models trained with small sets of random selected i–vector pairs were surprisingly good. Thus, we decided to further explore the random selection technique, leading to a two–step strategy, where the first step consists in random selection, which allows training an approximate PSVM. The second step exploits the scores of this PSVM for selecting the reduced set of training pairs. Thus, the approximate PSVM takes the role that PLDA has in the PLDA–based strategy. Again the main assumption for the success of this approach is that there is good correlation between the scores of the approximate PSVM and the scores of the PSVM trained with the full set.

It is worth noting that an efficient approach for training the original PSVM by means of a dual solver can be devised, because the kernel of two i–vector pairs, $(\mathbf{a}, \mathbf{b})$ and $(\mathbf{c}, \mathbf{d})$, can be efficiently computed by caching the Gram matrix, i.e., the cross–trial dot–products $\mathbf{a}^T\mathbf{c}$ and $\mathbf{b}^T\mathbf{d}$ [5], [28]. This approach, however, is not more efficient than our primal–solver approach, and requires the computation and storage of the dot–products between every i–vector $\mathbf{a}$ and all the other i–vectors in the training set. An i–vector pair selection strategy cannot help reducing the memory requirements for a dual solver. Even if a trial $(\mathbf{a}, \mathbf{b})$ were selected for removal, the memory needed for storing the Gram matrix would not change, unless the i–vector $\mathbf{a}$, and thus all its pairs $(\mathbf{a}, \cdot)$, were completely removed from the training set. On the other hand, pair selection is effective for a primal solver because it allows saving the computational and memory costs for all the pairs that are discarded.

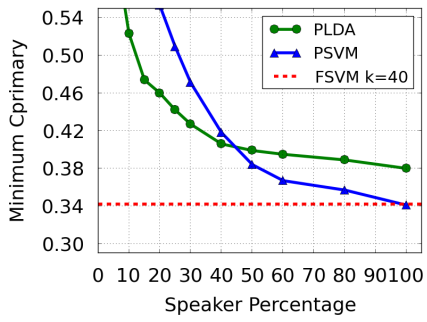
V. Experimental results

This section presents the results of a set of experiments performed on the NIST 2012 SRE extended core set [33]. Additional results comparing the standard PSVM and PLDA models also on the NIST 2010 SRE tasks, and using different i–vectors, can be found in [5], [6], [34]. In the set of experiments illustrated in this section, every utterance was processed after Voice Activity Detection, extracting, every 10 ms, 19 Mel frequency cepstral coefficients and the frame log-energy on a 25 ms sliding Hamming window. This 20–dimensional feature vector was subjected to short time mean and variance normalization using a 3s sliding window, and a 45-dimensional feature vector was obtained by stacking 18 cepstral (c_1-c_{18}), 19 delta (Δc_0-Δc_{18}) and 8 double–delta ($\Delta\Delta c_0$-$\Delta\Delta c_7$) parameters. We trained a gender–independent i–vector extractor, based on a 1024–component diagonal covariance gender–independent UBM, and on a gender-independent $\mathbf{T}$ matrix, estimated with data from NIST SRE 2004–2010, and additionally with the Switchboard II, Phases 2 and 3, and Switchboard Cellular, Parts 1 and 2 datasets, for a total of 66140 utterances. The i-vector dimension was fixed to $d = 400$. The PLDA and SVM models were trained using the NIST SRE 2004–2010 datasets, for a total of 48568 utterances of 3271 speakers. The complete enrollment dataset of the NIST 2102 was not used because it would become too large for training the PSVM reference system.

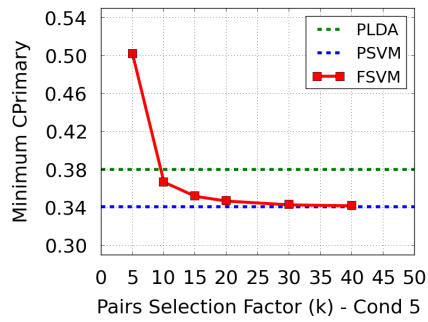
We trained PLDA models with full–rank channel factors, using 200 dimensions for the speaker factors. The i–vectors of the PLDA models were whitened and L_2 normalized. Within Class Covariance Normalization (WCCN) [35] was applied to the i–vectors for the PSVM. The WCCN transformations and the PLDA models have been trained using the previously mentioned NIST datasets. The SVM regularization coefficient λ was estimated from the data following the default method of SVM^{Light} (see Section VI-D of [5]). These systems were tested on the extended core NIST 2012 evaluation tasks [33]. The scores were not normalized.

A. PLDA–based i–vector selection

Table II summarizes the performance of three models, in terms of percent Equal Error Rate, minimum Decision Cost Function, and minimum C_{primary} with unknown non–target speakers only, defined for the NIST 2008 and 2012 evaluations [33], respectively. The results are shown for the reference PLDA model, for the PSVM model trained with the complete set, and for a pairwise SVM (FSVM) trained with a "PLDA–filtered" set of $40T$ pairs, where $T = 979540$ is the number of "same speaker" pairs. The FSVM model performs similarly to the PSVM, and better than the PLDA model, in all conditions, using only 1.7% of the training pairs.



(a) $C_{primary}$ of PLDA and PSVM as a function of the % of speakers included in the training set.



(b) $C_{primary}$ of FSVM models as a function of k, the multiple of the number of "same speaker" pairs.

Fig. 3: Performance comparison of PLDA and of pairwise SVM models on Condition 5 of NIST 2012 evaluation.

Figure 3a plots the minimum $C_{primary}$ obtained on NIST SRE 2012 Condition 5 by PLDA and by a set of PSVM models trained with utterances provided by an increasing number of speakers, as a function of the total number of speakers (reported as a percentage in the x–axis). The generative models of PLDA, trained with the utterances of a small subsets of the speakers, are better than the PSVM models, whereas the PSVM models prevail when the training set includes enough speakers (more than 50% of the speakers, in our experiment). The results of these PSVM models, trained with a reduced set of speakers, show that discriminative training is able to outperform PLDA models, but it requires a sufficient variety of speakers to avoid overfitting. The horizontal dashed line in the figure shows that a FSVM trained with the PLDA–selected $40T$–best scoring pairs is able to reach the performance of a PSVM trained with the complete training set, but using 60 times less memory.

Figure 3b shows the behavior of minimum $C_{primary}$ of six FSVMs trained with an increasing number of i–vectors, filtered from the complete set according to the PLDA–based approach. The number of the selected i–vectors in these sets is a multiple of the "same speaker" pairs. Using $k = 10$, i.e., just 10 times the number of "same speaker" pairs, our FSVM model outperforms PLDA. Increasing the value of factor k, the corresponding FSVM rapidly reaches the accuracy of the standard PSVM.

B. PSVM–based i–vector selection

We compared the PLDA–based selection with a straightforward uniform random selection of the same number of pairs, i.e., given the same factor k. The results obtained using this approach for all the NIST SRE 2012 conditions are shown in Figure 4, which plots the value of the parameter $C_{primary}$ as a function of factor k. Figure 5 completes the information related to Condition 5, with additional plots of the %EER, minimum DCF08, and minimum DCF10 costs [33]. The RSVM technique gives bad results if the number of filtered pairs is so low that it does not properly approximate the i–vector pair distribution, but it is surprisingly effective for k = 40, obtaining results similar to PLDA in most conditions.

The RSVM–based i–vector selection technique (labeled as RFSVM in the figures) is a natural extension of the purely random selection. It uses kT random selected pairs to train an initial RSVM, which is then exploited for scoring all training pairs, and for selecting the kT–best ones for training the final model. It is worth noting that we do not iterate, as other techniques suggest, because scoring all training pairs is expensive, and because the results are already accurate just after these two steps. The effectiveness of this approach is evident in Figure 4: just setting k = 5, i.e., by keeping only 0.2% of the training data, a RFSVM reaches the performance of the PSVM trained with the complete training set, with a 2% maximum relative variation, and using 480 times less memory. Moreover, the selection of a small number of patterns allows much faster training: in particular, for this dataset, the average time per iteration of the FSVM approach with $k = 40T$ was approximately 20s on a Xeon E5-2670 using 16 of its 32 cores, a big reduction compared to the average 330s required by the PSVM system. The reduction is even more significant considering that the full PSVM was trained with optimized BLAS libraries, whereas less optimized sparse matrix routines were used for the FSVM systems. All these results are obtained by PSVM and

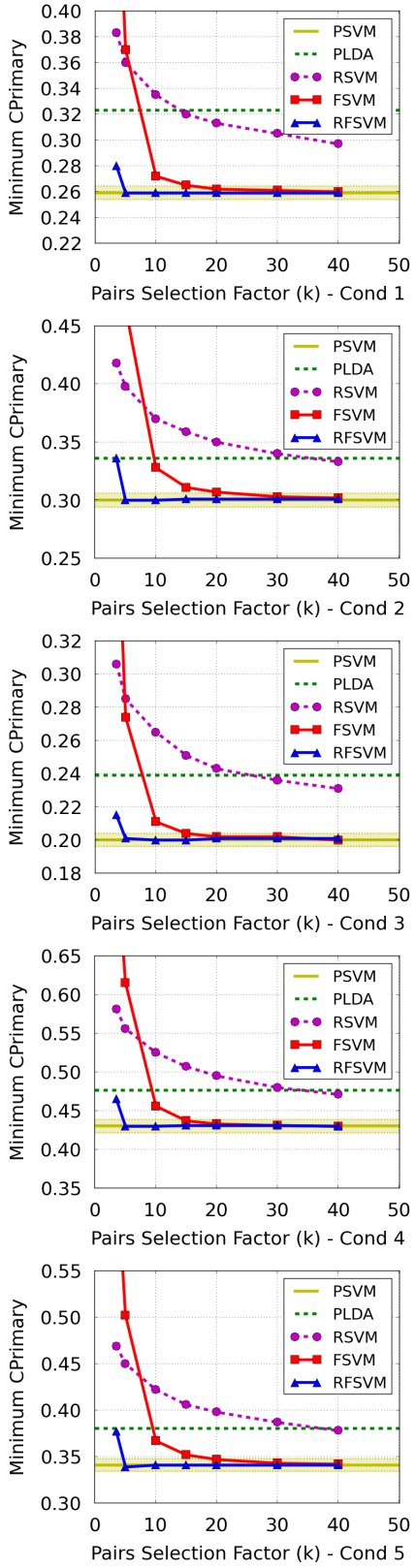
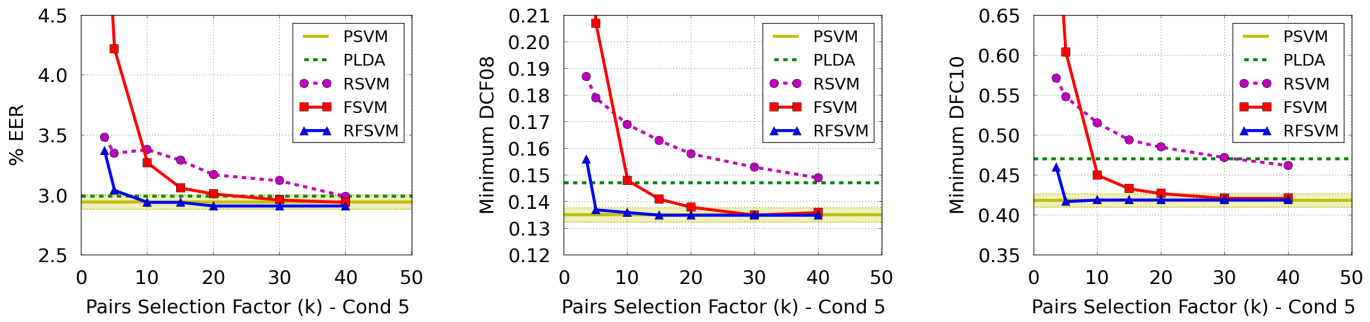


Fig. 4: Performance comparison of five classification models: PLDA, random i–vector selection SVM (RSVM), PLDA–filtered selection SVM (FSVM), RSVM–filtered selection (RFSVM), standard Pairwise SVM (PSVM). Minimum $C_{primary}$ for the five conditions defined by NIST 2012 evaluation. The shadowed region in each graph represents the 2% relative variation of the PSVM performance.



(a) % EER (b) Minumum DCF08 (c) Minumum DCF10

Fig. 5: % EER, minumum DCF08, and minimum DCF10 for Condition 5 of NIST SRE 2012. The shadowed region in each graph represents the 2% relative variation of the PSVM performance.

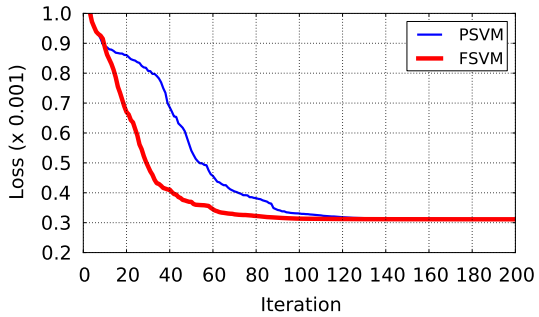


Fig. 6: Loss function of the PSVM and FSVM models as a function of the iteration number.

FSVM models trained with 200 iterations, corresponding approximately to 1 and 18 training time hours for FSVM and PSVM, respectively. This number of iterations is sufficiently large for estimating the asymptotic behavior of the two approaches. The number of iterations, however, can be reduced by means of different stopping criteria. Theoretical bounds on the number of iterations required to achieve a given precision are provided in [30], [26] for the BMRM and OCAS solvers. In practice, we observed that the FSVM convergence is much faster compared to the standard PSVM, as shown in Figure 6, which plots the loss function computed on the training set, and the minumum $C_{primary}$ computed on the test set, as a function of the number of iterations performed in training. The RFSVM loss functions are not plotted because, by changing the number of iterations of the first RSVM step, the selected pairs may change, making the RFSVM loss functions not comparable. The FSVM approach requires approximately a half of the iterations of PSVM to get the same performance, as shown in Figure 7.

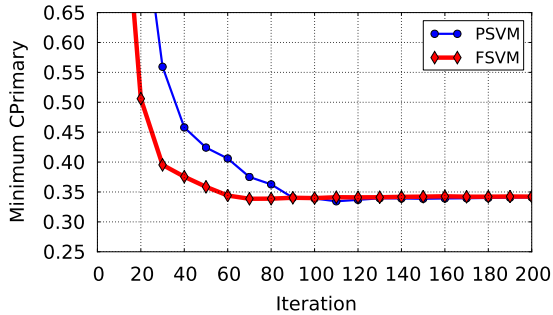


Fig. 7: PSVM and FSVM minimum $C_{primary}$ for NIST 2012 Condition 5 as a function of the number of iterations performed in training.

VI. Conclusions

We addressed the computational and memory issues raised by the quadratic increase of the i–vector pairs in Pairwise SVM training. We demonstrated that the number of support vectors is bounded by a linear function of the number of "same speaker" pairs, thus it does not increase quadratically, but linearly with the number of speakers. Two simple and effective approaches for discarding the i–vector pairs that are not essential in training a pairwise SVM have been presented. The first one exploits the correlation between the PSVM and the PLDA scores on the training data, whereas the second one uses the scores of an approximate PSVM trained with a reduced set of random selected pairs. The selection based on the approximate PSVM has shown to be more effective than PLDA–based selection when the number of the reduced training set is a small multiple of the "same speaker" pairs. Using these i–vector pairs selection approaches, and a primal solver, PSVM training with very large set of speakers becomes feasible. This is important because we have shown that FSVM benefits from the use of additional data of different speakers, and that the FSVM models trained with a large enough training set can perform better than PLDA models.

Appendix

Proposition 1: In a binary SVM, the maximum number of bounded Support Vectors cannot be greater than two times the number of training patterns of the less populated class.

Proof: For a given a training set $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ with $x_i \in \mathbb{R}^d$ and $y_i \in \{1, +1\}$, the primal SVM optimization problem is formulated as [31]:

$$\begin{aligned} &\underset{\mathbf{w},\mathbf{b},\boldsymbol{\xi}_i>0}{\operatorname{argmin}} \tfrac{1}{2}\|\mathbf{w}\|^2 + C\textstyle\sum_{i=1}^n \xi_i \\ \textit{subject to:}\quad & y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n\ . \end{aligned} \tag{15}$$

The corresponding Lagrange functional is:

$$\begin{aligned} L =& \frac{1}{2}\|\mathbf{w}\|^2 - \sum_i \alpha_i \left[y_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) - 1 + \xi_i\right] \\ & - \sum_i \mu_i \xi_i + C\sum_{i=1}^n \xi_i\ , \end{aligned} \tag{16}$$

where α_i, and μ_i are the Lagrange multipliers. Recalling the Karush–Kuhn–Tucker conditions for this problem:

$$\mathbf{w} - \sum_i \alpha_i y_i \mathbf{x}_i = \mathbf{0}\ , \tag{17}$$

$$-\sum_i \alpha_i y_i = 0\ , \tag{18}$$

$$C - \alpha_i - \mu_i = 0\ , \tag{19}$$

$$y_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) - 1 + \xi_i \geq 0\ , \tag{20}$$

$$\xi_i \geq 0\ , \tag{21}$$

$$\alpha_i \geq 0\ , \tag{22}$$

$$\mu_i \geq 0\ , \tag{23}$$

$$\alpha_i\left[y_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) - 1 + \xi_i\right] = 0\ , \tag{24}$$

$$\mu_i \xi_i = 0\ , \tag{25}$$

which provide a subset of necessary and sufficient conditions for optimality, (18) can be rewritten as:

$$\sum_{i|y_i=-1} \alpha_i = \sum_{i|y_i=+1} \alpha_i\ . \tag{26}$$

Let $\mathcal{B}_+$ denote the set of bounded SVs for class $y_i = +1$, and let $\mathcal{A}_+$ denote all the training data set belonging to class $y_i = +1$. Let the corresponding notation for class $y_i = -1$ be $\mathcal{B}_-$ and $\mathcal{A}_-$, respectively. Without loss of generality, we focus on the number of bounded SVs for class $y_i = -1$, assuming that it is the most populated class,

i.e. $\mathcal{A}_+ \leq \mathcal{A}_-$.
Recalling that the Lagrange multiplier for a bounded support vector i is $\alpha_i = C$, we get:

$$C\,|\mathcal{B}_-| = \sum_{i\in\mathcal{B}_-} C \;= \sum_{i\in\mathcal{B}_-} \alpha_i \tag{27}$$

$$\leq \sum_{i\in\mathcal{A}_-} \alpha_i = \sum_{i\in\mathcal{A}_+} \alpha_i \tag{28}$$

$$\leq \sum_{i\in\mathcal{A}_+} C \;=\; C\,|\mathcal{A}_+| \tag{29}$$

where (28) follows from (26).
Therefore, $|\mathcal{B}_-| \leq |\mathcal{A}_+|$, and since $|\mathcal{B}_+| \leq |\mathcal{A}_+|$ we get:

$$|\mathcal{B}_-| + |\mathcal{B}_+| \leq 2\,|\mathcal{A}_+| \;\; . \tag{30}$$

■

Corollary 1: Assuming that the average number of utterances per speaker is bounded, the number of bounded SVs for the biased PSVM approach grows at most linearly with the number of speakers.

Proposition 2: Let $\alpha_U^t = \left\{\alpha_i^t | 0 < \alpha_i^t < C\right\}$ be the set of Lagrange multipliers associated to unbounded SVs, of dimension d, corresponding to a solution $\mathcal{S}^t = (\boldsymbol{\alpha}^t, \mathbf{w}^t, \boldsymbol{\xi}^t, \boldsymbol{\mu}^t, b^t)$ satisfying the KKT conditions (17–25). Then, if the number of unbounded SVs, $\left|\alpha_U^t\right|$, is greater than $d+1$, there exists a solution $\mathcal{S}^{t+1}$, satisfying the KKT conditions, such that $\left|\alpha_U^{t+1}\right| < \left|\alpha_U^t\right|$.

In the literature it is possible to find the proof that the number of the "essential" SVs, i.e., the ones that appear in all possible expansion of the optimal hyperplane is not greater than d (see Theorem 10.4 in [36]), In [37] it is shown that linear dependent support vectors can be recognized and eliminated while leaving the solution otherwise unchanged. While these contributions can be exploited for reducing the number of SVs at testing time, however they do not guarantee that the selected patterns include all the support vectors necessary to estimate the original hyperplane. In particular, these approaches allow the hyperplane to be represent as a linear combination of patterns, but they do not ensure that the combination weights satisfy the KKT conditions. In [38] a proof that the number of irreducible support vectors cannot exceed the dimension of the feature space plus 1 is given, based on the properties of the Hessian matrix of the dual problem. In this work we give a different proof which develops and completes the considerations on the non–uniqueness of the dual solution presented in Section 4.4 of [31], and illustrates an iterative procedure to decrease the number of unbounded support vectors of an SVM solution whenever such number is larger than $d+1$.

Proof: Let $\mathbf{Y} = (y_i, \ldots, y_n)^T$ be the array of labels, and $\boldsymbol{\Phi} = [y_1\mathbf{x}_1, \ldots, y_N\mathbf{x}_N]$ the matrix consisting of the training feature vectors $\mathbf{x}$, each multiplied by its corresponding label. We assume that the number of samples n is greater than the feature dimension d, i.e., $n \geq d+1$, otherwise the proof would trivially be concluded because the number of unbounded SVs would not be greater than $d+1$.

Let $N(\boldsymbol{\Phi}) = \{\mathbf{v} | \Phi\mathbf{v} = 0\}$ denote the nullspace of $\boldsymbol{\Phi}$, and let also $H = \{\mathbf{v} \in N(\boldsymbol{\Phi}) | \mathbf{v}^T\mathbf{Y} = 0\}$, so that for any vector $\mathbf{v} = [v_1, \ldots, v_n]^T \in H$, $\sum_i v_i y_i \mathbf{x}_i = 0$, and $\sum_i v_i y_i = 0$ hold. Since last expression is formally equivalent to KKT condition (18), it follows that any dual feasible SVM solution $\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t + \mathbf{v}$ satisfies KKT condition (18). Moreover, if one sets $\mathbf{w}^{t+1} = \mathbf{w}^t$, KKT condition (17) is satisfied by the dual solution $\boldsymbol{\alpha}^{t+1}$. Thus, the dual solutions $\boldsymbol{\alpha}^t$ and $\boldsymbol{\alpha}^t + \mathbf{v}$ correspond to the same primal solution. In the following, we look for a feasible dual solution $\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t + \mathbf{v}$, obeying the KTT constraints, and allowing the number of unbounded SVs to be reduced.

The rank of matrix $\boldsymbol{\Phi}$ is $\rho_{\boldsymbol{\Phi}} \leq d$. Since, by the rank-nullity theorem of linear algebra [39], the dimension of the nullspace of $\boldsymbol{\Phi}$ is $\dim\left[N\left(\boldsymbol{\Phi}\right)\right] = n - \rho_{\boldsymbol{\Phi}}$, then $\dim(H) \geq n - d - 1$, due to the additional constraint $\mathbf{v}^T\mathbf{Y} = 0$.

Let U be the subspace of $\mathbb{R}^n$ spanned by the subset of the canonical base of $\mathbb{R}^n$, $U = \operatorname{span}\{\mathbf{e}_i | 0 < \alpha_i < C\}$, where:

$$\mathbf{e}_i = [\underbrace{0\ldots0}_{i-1}\;\; 1\;\; \underbrace{0\ldots0}_{n-i}]\;,\quad \forall i\;.$$

The dimension of U is, thus, the number of unbounded SVs. Subspace U can be alternatively defined as $U = \left\{\mathbf{u} \in \mathbb{R}^n | (\alpha_i^t = 0 \vee \alpha_i^t = C) \implies u_i = 0\;, \forall i\right\}$, i.e., the subspace of all vectors $\mathbf{u} \in \mathbb{R}^n$, with all their components

set to zero, except for the ones corresponding to unbounded SVs. According to this definition, two vectors $\boldsymbol{\alpha}^t$ and $\boldsymbol{\alpha}^t + \mathbf{u}$ differ only for the values associated to the index of the unbounded SVs. If $\dim(U) \leq \rho_{\boldsymbol{\Phi}} + 1$ the number of unbounded SVs is also not greater than $d+1$, otherwise, since $\dim(H) + \dim(U) > n$, then $\dim(H \cap U) \geq 1$, thus a vector $\mathbf{u} \in H \cap U$ exists such that $\mathbf{u} \neq \mathbf{0}$.

Let $S^{t+1} = (\boldsymbol{\alpha}^{t+1}, \mathbf{w}^{t+1}, \boldsymbol{\xi}^{t+1}, \boldsymbol{\mu}^{t+1}, b^{t+1})$ be a new solution, with $\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t + r\mathbf{u}$, with $r \in \mathbb{R}$. Factor r has a simple geometrical interpretation: $\boldsymbol{\alpha}^{t+1}$ is at distance r from point $\boldsymbol{\alpha}^t$ along the direction defined by vector $\mathbf{u}$. We will show that an appropriate choice of factor r leads to a new equivalent solution, which has a number of unbounded SVs reduced at least by one with respect to S^t, and does not violate any KKT condition.

Since $r\mathbf{u} \in H$, as long as we set $\mathbf{w}^{t+1} = \mathbf{w}^t$, KKT conditions (17) and (18) are not violated regardless of the choice of r. The KKT conditions define a set of box constraints $0 \leq \alpha_i \leq C$, but since $\mathbf{u} \in U$, the Lagrange multipliers α_i^{t+1} not associated with to unbounded SVs in solution S^t do not change regardless of the choice of r, thus the corresponding constraints are satisfied also for solution S^{t+1}. In order to avoid violating the KKT conditions for the unbounded SVs, $\boldsymbol{\alpha}^t$ can be moved along the direction defined by vector $\mathbf{u}$ only until it reaches one of the box constraints, i.e., the associated SV becomes bounded ($\alpha_i^{t+1} = C$), or it does no more belong to the set of SVs $\alpha_i^{t+1} = 0$. Thus, moving $\boldsymbol{\alpha}^t$ to one of that boundaries allows the number of unbounded SVs to be reduced at least by one.

In order to find an appropriate factor r, let us define, for each element i of $\mathbf{u}$ such that $u_i \neq 0$, $r_i = \underset{r}{\operatorname{argmax}} |r|$ subject to $0 \leq \alpha_i + ru_i \leq C$. It is worth noting that, since $\mathbf{u} \neq \mathbf{0}$, at least one element u_i of $\mathbf{u}$ is different from 0, and that $|r_i| > 0$ because $0 < \alpha_i < C$. The maximum $|r|$ is attained on the boundary defined by the constraint $0 \leq \alpha_i + ru_i \leq C$, i.e., for $|r_i| = \min\left(\left|\frac{\alpha_i}{u_i}\right|, \left|\frac{C-\alpha_i}{u_i}\right|\right)$. Moreover, for any r_i' such that $|r_i'| < |r_i|$, $0 \leq \alpha_i + r_i' u_i \leq C$. By selecting the unbounded Lagrange multiplier nearest to one of the box constraints, $j = \underset{i}{\operatorname{argmin}} \{|r_i|\}$, we assure that also the Lagrange multipliers of all other vectors satisfy $0 \leq \alpha_i + r_j u_i \leq C$, because either $|r_j| \leq |r_i|$ or $u_i = 0$. For the selected factor r_j, either $\alpha_j^{t+1} = \alpha_j^t + r_j u_j = C$ or $\alpha_j^{t+1} = 0$ will hold, thus the corresponding SV will no more belong to the unbounded set of SVs, whereas all the other Lagrange multipliers will satisfy KKT condition (22).

Thus, a new SVM solution can be obtained by setting $\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t + r_j\mathbf{u}$. We already observed that the new hyperplane does not change: $\mathbf{w}^{t+1} = \boldsymbol{\Phi}\alpha^t + r\boldsymbol{\Phi}\mathbf{u} = \mathbf{w}^t$. Let the new bias be $b^{t+1} = b^t$ and let $\boldsymbol{\xi}^{t+1} = \boldsymbol{\xi}^t$. Setting $\mu_i^{t+1} = C - \alpha_i^{t+1}$ for every i, the new solution satisfies all the KKT conditions:

- condition (17) is verified because $\mathbf{w}^{t+1} = \mathbf{w}^t$,
- condition (18) holds because $\mathbf{u} \in H$ and $(\boldsymbol{\alpha}^t)^T\mathbf{1} = 0$, thus $(\boldsymbol{\alpha}^{t+1})^T\mathbf{Y} = (\boldsymbol{\alpha}^t)^T\mathbf{Y} + r\mathbf{u}^T\mathbf{Y} = 0$,
- condition (19) follows from the definition of μ_i^{t+1},
- conditions (20), (21) follow from $\mathbf{w}^{t+1} = \mathbf{w}^t$, $b^{t+1} = b^t$ and $\boldsymbol{\xi}^{t+1} = \boldsymbol{\xi}^t$ constant,
- conditions (22), (23) are verified because $\mathbf{u} \in U$, so that $\alpha_i^{t+1} = \alpha_i^t$ for all i associated with $\alpha_i^t = 0$ or $\alpha_i^t = C$, and from to the selection of r, which guarantees that $0 \leq \alpha_i^{t+1} \leq C$ if $0 < \alpha_i^t < C$.
- Showing that condition (24) holds, requires recalling that $\mathbf{w}^{t+1} = \mathbf{w}^t$, and that the bias b does not change. Let us consider separately the unbounded SVs from the others. For the latter $\alpha_i^{t+1} = \alpha_i^t$ and $\xi_i^{t+1} = \xi_i^t$, thus, the condition is satisfied. The condition is satisfied also for the unbounded SVs, because $\xi_i^t = 0$, and $y_i(\mathbf{w}_i^{t+1})^T\mathbf{x}_i - 1 = y_i(\mathbf{w}_i^t)^T\mathbf{x}_i - 1 = 0$, and because we let $\xi_i^{t+1} = \xi_i^t = 0$.
- In order to verify condition (25), it is worth noting again that $\boldsymbol{\xi}^{t+1} = \boldsymbol{\xi}^t$. Then, for the original unbounded SVs, $\xi_i^t = 0$, so that $\mu_i^{t+1}\xi_i^{t+1} = 0$. For all the other vectors, the Lagrange multipliers do not change ($\alpha_i^{t+1} = \alpha_i^t$), therefore $\mu_i^{t+1} = \mu_i^t$, so that $\mu_i^{t+1}\xi_i^{t+1} = \mu_i^t\xi^t$.

Summarizing, an appropriate choice of r allows the number of unbounded SVs in the new solution to be reduced at least by one: the unbounded SV associated to α_j is either removed from the set of SVs, or it is transformed into a bounded SV, and the Lagrange multipliers of the other unbounded SVs are updated, whereas the Lagrange multipliers of all the other vectors keep their values. ■

We finally give the corollary that comes from our two propositions.

Corollary 2: Let define $A_{min} = \min(A_+, A_-)$, then, for a finite dimensional kernel K, there exists a SVM solution whose number of SVs does not exceed $2|A_{min}| + d + 1$.

REFERENCES

[1] S. Cumani and P. Laface, "Training pairwise Support Vector Machines with large scale datasets," in *ICASSP 2014*, 2014.

[2] S. J. D. Prince and J. H. Elder, "Probabilistic Linear Discriminant Analysis for inferences about identity," in *Proceedings of 11th International Conference on Computer Vision*, 2007, pp. 1–8.

[3] P. Kenny, "Bayesian speaker verification with Heavy–Tailed Priors," in *Keynote presentation, Odyssey 2010, The Speaker and Language Recognition Workshop*, 2010, Available at http://www.crim.ca/perso/patrick.kenny/kenny_Odyssey2010.pdf.

[4] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front–end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[5] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian Mixture Models," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 31–44, 2000.

[6] S. Cumani, N. Brümmer, L. Burget, P. Laface, O. Plchot, and V. Vasilakakis, "Pairwise Discriminative Speaker Verification in the I-Vector Space," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1217–1227, 2013.

[7] S. Cumani, N. Brümmer, L. Burget, and P. Laface, "Fast Discriminative Speaker Verification in the i–vector Space," in *Proceedings of ICASSP 2011*, 2011, pp. 4852–4855.

[8] V. N. Vapnik, *The nature of statistical learning theory*, Springer-Verlag, 1995.

[9] Corinna Cortes and Vladimir Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[10] I. Steinwart, "Sparseness of Support Vector Machines," *Journal of Machine Learning Research*, vol. 4, pp. 1071–1105, 2003.

[11] G.H. Bakir, L. Bottou, and J. Weston, "Breaking SVM complexity with cross training," in *Proc. of NIPS 2005*, 2005.

[12] H.P. Graf, E. Cosatto, L. Bottou, I. Dourdanovic, and V. Vapnik, "Parallel support vector machines:The cascade SVM," in *Advances in neural information processing systems*, vol. 17, pp. 521–528. MIT Press, 2005.

[13] N. Panda, E.Y. Chang, and G. Wu, "Concept Boundary Detection for Speeding Up SVMs," in *Proceedings of ICML 2006*, 2006, pp. 681–688.

[14] X.Liu, J.F. Beltran, N. Mohanchandra, and G.T. Toussaint, "On Speeding Up Support Vector Machines: Proximity Graphs Versus Random Sampling for Pre-Selection Condensation," *World Academy of Science, Engineering and Technology*, vol. 73, pp. 905–912, 2013.

[15] J. Wang, P. Neskovic, and L.N. Cooper, "Selecting Data for Fast Support Vector Machines Training," in *Trends in Neural Computation*, vol. 35, pp. 61–84. Springer Berlin Heidelberg, 2007.

[16] V. N. Vapnik, *Estimation of Dependences Based on Empirical Data*, Springer-Verlag, 1982.

[17] P.E. Hart, "The Condensed Nearest Neighbor Rule," *IIEEE Transactions on Information Theory*, vol. 18, no. 3, pp. 515–516, 1968.

[18] T. Joachims, "Making large–scale Support Vector Machine learning practical," in *Advances in Kernel Methods – Support Vector Learning*. 1999, pp. 169–184, MIT–Press.

[19] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in *Proceedings of COLT '92*, 1992, pp. 144–152.

[20] A.J. Smola and B. Schölkopf, "A Tutorial on Support Vector Regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[21] V. Tresp, "Scaling Kernel-Based Systems to Large Data Sets," *Data Mining and Knowledge Discovery*, vol. 5, no. 3, pp. 197–211, 2001.

[22] I.W.Tsang, J.T. Kwok, and P.M. Cheung, "Simpler Core Vector Machines with Enclosing Balls," in *Proceeding of ICML '07*, 2007, pp. 911–918.

[23] I.W.Tsang, J.T. Kwok, and P.M. Cheung, "Core Vector Machines: Fast SVM Training on Very Large Data Sets," *Journal of Machine Learning Research*, vol. 6, pp. 363–392, 2005.

[24] T. Joachims, "Training Linear SVMs in Linear Time," in *Proceedings of the KDD 2006*, 2006, pp. 217–226.

[25] V. Franc and S. Sonnenburg, "Optimized cutting plane algorithm for Support Vector Machines," in *Proceedings of ICML 2008*, 2008, pp. 320–327.

[26] V. Franc and S. Sonnenburg, "Optimized Cutting Plane Algorithm for Large-Scale Risk Minimization," *J. Mach. Learn. Res.*, vol. 10, pp. 2157–2192, Dec. 2009.

[27] C. Chang and C. Lin, "LIBSVM: a Library for Support Vector Machines," *ACM Transactions on Intelligent Systems and Technology 2 (3)*, vol. 2, no. 3, 2001.

[28] S. Yaman and J. Pelecanos, "Using Polynomial Kernel Support Vector Machines for Speaker Verification," *IEEE Signal Processing Letters*, vol. 20, no. 9, pp. 901–904, 2013.

[29] S. Cumani and P. Laface, "Analysis of Large–Scale SVM Training Algorithms for Language and Speaker Recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1585–1596, 2012.

[30] C. H. Teo, A. Smola, S. V. Vishwanathan, and Q. V. Le, "Bundle Methods for Regularized Risk Minimization," *J. Mach. Learn. Res.*, vol. 11, pp. 311–365, March 2010.

[31] C. J. C. Burges, "A tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.

[32] S. Sahni, *Data Structures, Algorithms and Applications in Java*, McGraw-Hill, Inc., 1999.

[33] "The NIST Year 2012 Speaker Recognition Evaluation Plan," Available at "http://www.nist.gov/itl/iad/mig/upload/NIST_SRE12_evalplan-v17-r1.pdf.

[34] S. Cumani, O. Glembek, N. Brummer, E. de Villiers, and P. Laface, "Gender independent discriminative speaker recognition in i-vector space," in *Proceedings of ICASSP 2012*, 2012, pp. 4361–4364.

[35] A. Hatch, S. Kajarekar, and A. Stolcke, "Within-Class Covariance Normalization for SVM-Based Speaker Recognition," in *Proceedings of ICSLP 2006*, 2006, pp. 1471–1474.

[36] V. N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, 1998.

[37] T. Downs, K.E. Gates, and A. Masters, "Exact simplification of support vector solutions," *Journal of Machine Learning Research*, vol. 2, pp. 293–297, 2001.

[38] S. Abe, *Support Vector Machines for Pattern Classification*, Springer Publishing Company, 2012.

[39] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra*, Society for Industrial and Applied Mathematics, 2000.