Non-Linear Stationary Subspace Analysis with Application to Video Classification

Mahsa Baktashmotlagh[†‡] MAHSA.BAKTASHMOTLAGH@NICTA.COM.AU
Mehrtash T. Harandi[‡] MEHRTASH.HARANDI@NICTA.COM.AU
Abbas Bigdeli[‡] ABBAS.BIGDELI@NICTA.COM.AU
Brian C. Lovell[†] LOVELL@ITEE.UQ.EDU.AU
Mathieu Salzmann[‡] MATHIEU.SALZMANN@NICTA.COM.AU
[†]University of Queensland, School of ITEE, QLD 4072, Australia
[‡]NICTA*, Locked Bag 8001, Canberra, ACT 2601, Australia

Abstract

Low-dimensional representations are key to the success of many video classification algorithms. However, the commonly-used dimensionality reduction techniques fail to account for the fact that only part of the signal is shared across all the videos in one class. As a consequence, the resulting representations contain instance-specific information, which introduces noise in the classification process. In this paper, we introduce Non-Linear Stationary Subspace Analysis: A method that overcomes this issue by explicitly separating the stationary parts of the video signal (i.e., the parts shared across all videos in one class), from its non-stationary parts (i.e., specific to individual videos). We demonstrate the effectiveness of our approach on action recognition, dynamic texture classification and scene recognition.

1. Introduction

Video classification is a challenging computer vision task with many potential applications, such as action recognition, anomaly detection, and face recognition. A key ingredient to video classification is the design of effective video models. A popular approach to this problem is to learn a low-dimensional representation of the videos for each individual class using methods such as PCA (Ali & Shah, 2010), ICA (Long et al., 2012), ISA (Le et al., 2011) or LPP (Tseng et al., 2012). Classification is then performed by projecting a query video to the low-dimensional representation of each class, and finding which projection is most similar to the training data.

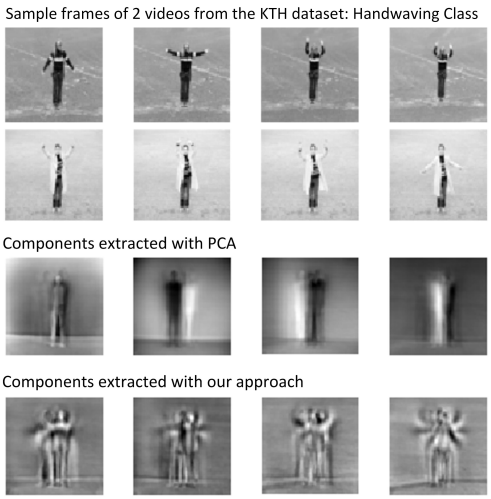

Figure 1. PCA components and stationary components extracted from the Hand Waving class of the KTH dataset.

Despite their success for video classification, an inherent limitation of most dimensionality reduction methods is their assumption that, within each class, the video signal of all the examples is drawn from a single, potentially multi-modal, distribution. While part of the signal is indeed common to all videos in the class, other parts are specific to each individual video. From a signal processing perspective, it would therefore seem more reasonable to model each video as the superposition of a stationary part, shared across all videos of the class, and a non-stationary part, only present in this particular video. In most cases, the non-stationary part is irrelevant for video classification, since it only carries information about a single video. Modeling this information, as most dimensionality reduction methods do, only introduces noise, and therefore might degrade the classification performance.

*NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and Digital Economy and the ARC through the ICT Centre of Excellence program.

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

In light of this observation, in this paper, we introduce the use of stationarity for video classification. Being shared by all the videos, a stationary signal makes a very good representative of the class. To illustrate this, in Fig. 1, we compare the components recovered with PCA and the stationary signal extracted from the Hand Waving class of the KTH dataset. Note that this stationary signal better captures the action than PCA.

To extract the stationary part of the data, we introduce two methods: Kernel Stationary Subspace Analysis (KSSA) and Non-Linear Stationary Subspace Analysis (NLSSA). These two methods are non-linear extensions to Stationary Subspace Analysis (SSA) (Hara et al., 2012). Given a time-varying multivariate signal, SSA searches for a subspace shared across multiple portions of the signal (i.e., epochs), such that the projections of these epochs onto the subspace are stationary. Following a weak notion of stationarity, SSA minimizes the KL divergence between the distributions of all pairs of projected epochs. Here, we show that SSA can be kernelized, thus allowing us to model a non-linear mapping between the original signal and its stationary representation via the use of kernels. Furthermore, we introduce a modification of this KSSA formulation that remains non-linear, while significantly reducing computational complexity.

To summarize, our contribution is twofold: We introduce the use of stationarity for video classification and propose two novel non-linear algorithms to extract the stationary part of an observed signal. We demonstrate the benefits of our approach over exiting techniques on dynamic texture recognition, scene classification, and action recognition.

2. Related Work

Video classification has attracted a lot of interest since it provides a solution to a wide class of problems, such as action, or dynamic scene recognition. For many existing approaches, constructing a compact discriminative representation of videos has been an important research focus. In particular, Principal Component Analysis (PCA), which computes a low-dimensional representation of the data so as to minimize its reconstruction error, has been widely used to model videos (Ali & Shah, 2010). More recently, several variations of PCA, such as Generalized PCA (Vidal et al., 2005), Mixtures of Probabilistic PCA (Gu et al., 2001; Tipping et al., 1999) and Kernel PCA (Hotta, 2012; Chan & Vasconcelos, 2007) have been proposed for video classification. Unfortunately, PCA-based approaches suffer from the fact that the components yielding the best reconstruction of the data might not be the most relevant ones for classification purpose.

As an alternative to PCA, Independent Component Analysis (ICA) has been utilized to find a subspace of the data. ICA minimizes the mutual information of the projections of the data along the different components. In (Long et al., 2012), it was employed to learn spatio-temporal filters from unlabeled video data. Similarly, Independent Subspace Analysis (ISA), a generalization of ICA, was used to learn invariant spatio-temporal features from video for action classification (Le et al., 2011). As PCA-based approaches, these methods try to model the entire data instead of focusing on the part that is shared across all videos of the same class, and thus relevant for classification.

Other subspace methods that more directly exploit the structure of the training data have been used for video classification. For instance, (Tseng et al., 2012) employed (Adaptive) Locality Preserving Projection (LPP) for silhouette-based human action recognition. However, while LPP attempts to preserve the neighborhood structure of the data, this structure may not necessarily reflect the underlying classes. More directly focused on the classification problem, an approach relying on Kernel Fisher Discriminant Analysis (FDA) was proposed in (Campos et al., 2011). While this approach better accounts for the underlying classification problem, its results do not seem to be competitive with the state-of-the-art.

Many other video classification methods that do not make use of subspace representations have also been proposed. For instance, Support Vector Machines (SVMs) have become popular to perform video classification using various image features, such as Histogram of Oriented Gradients (HOG) (Thurau & Hlavác, 2008), or SIFT (Sivic & Zisserman, 2003). Since SVMs do not directly take temporal information into account, spatio-temporal features had to be designed (Knopp et al., 2010; Niebles et al., 2008; Wang et al., 2012).Probabilistic generative models have also been proposed to represent a video as the output of a linear dynamical system (LDS) (Doretto et al., 2003; Chaudhry et al., 2009; Saisan et al., 2001) or of a Bag-of-Dynamical-systems (Ravichandran et al., 2009; Chan et al., 2010).

Here, we introduce a non-linear approach to video modeling that focuses on extracting the information that is stationary across all videos of the same class. As a consequence, the resulting video representation is particularly well-suited for classification purpose. Furthermore, the notion of stationarity is intuitively well-adapted to model the temporal nature of the video signal and lets us make use of many image features.

3. Background

In this section, we briefly review Stationary Subspace Analysis (SSA) (Bünau et al., 2009) that serves as a starting point for our approach. SSA is a blind source separation method that aims to factorize a multivariate time series into stationary and non-stationary sources. Its underlying assumption is that the observed signal $\boldsymbol{X}$ is generated from a linear mixture of d stationary sources $\boldsymbol{S}^s \in \mathbb{R}^{d\times T}$ and $D-d$ non-stationary ones $\boldsymbol{S}^n \in \mathbb{R}^{D-d\times T}$. This can be written as

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{S} = \begin{bmatrix}\boldsymbol{A}^s & \boldsymbol{A}^n\end{bmatrix}\begin{bmatrix}\boldsymbol{S}^s \\ \boldsymbol{S}^n\end{bmatrix}, \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{D\times D}$ is an invertible mixture matrix composed of $\boldsymbol{A}^s$ and $\boldsymbol{A}^n$, which span the stationary and non-stationary subspaces, respectively. The goal of SSA is to estimate a de-mixing transformation $\boldsymbol{W}$ from the data $\boldsymbol{X}$ such that stationary and non-stationary sources can be separated. The de-mixing matrix is related to $\boldsymbol{A}$ by

$$\boldsymbol{B} = \boldsymbol{A}^{-1} = \begin{bmatrix}\boldsymbol{W} \\ \boldsymbol{B}^n\end{bmatrix}. \tag{2}$$

SSA exploits the notion of weak, or wide-sense, stationarity, which translates into considering d sources as stationary if they have the same mean and variance over time. To this end, the data is divided into N time blocks, or epochs. The stationary part of the signal is then extracted by finding the projection $\boldsymbol{W}$, such that the mean and covariance of the projected signal are the same for all epochs. Within this linear framework, this is equivalent to comparing the projections of the mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ of the original signal of each epoch to the projections of the average mean $\bar{\boldsymbol{\mu}} = (1/N)\sum_{i=1}^{N}\boldsymbol{\mu}_i$ and average covariance $\bar{\boldsymbol{\Sigma}} = (1/N)\sum_{i=1}^{N}\boldsymbol{\Sigma}_i$. As a distance measure, SSA utilizes the Kullback-Leibler divergence, which yields the minimization problem

$$\min_{\boldsymbol{W}} \sum_{i=1}^{N} \mathrm{KL}\Big(\mathcal{N}(\boldsymbol{W}\boldsymbol{\mu}_i, \boldsymbol{W}\boldsymbol{\Sigma}_i\boldsymbol{W}^T) || \mathcal{N}(\boldsymbol{W}\bar{\boldsymbol{\mu}}, \boldsymbol{W}\bar{\boldsymbol{\Sigma}}\boldsymbol{W}^T)\Big). \tag{3}$$

Solving this problem for $\boldsymbol{W}$ is ambiguous. Therefore, additional orthogonality constraints of the form $\boldsymbol{W}\bar{\boldsymbol{\Sigma}}\boldsymbol{W}^T = \boldsymbol{I}$ are enforced to restrict the solution space. In practice, the solution is obtained by searching for a matrix $\boldsymbol{B}$ in the special orthogonal group $SO(D)$, and by taking $\boldsymbol{W} = \boldsymbol{I}^d\boldsymbol{B}$, with $\boldsymbol{I}^d$ containing the first d rows of the identity matrix. In (Bünau et al., 2009), a steepest descent method on $SO(D)$ was utilized to determine $\boldsymbol{B}$. However, the non-convexity and flatness of the objective near the global solution made the optimization process slow (Hara et al., 2012).

This was overcome in (Hara et al., 2012) by introducing an approximate analytical solution to SSA. While this formulation has proved more robust than the original SSA, it still assumes a linear mapping between the original signal and its stationary part. In the next section, we show how stationarity can be exploited for video classification, and how Analytic SSA (Hara et al., 2012) can be kernelized to model non-linear mappings.

4. Proposed Approach

In this section, we introduce the use of stationarity for video classification. We first discuss how to extract the stationary part of a set of videos belonging to the same class, and how to model a non-linear mapping between such stationary signal and the original video data. Finally, we show how to employ this process for video classification.

4.1. Stationarity for Video Modeling

In this section, we present our approach to obtaining a compact representation of videos. Let us recall that, ultimately, our goal is video classification. Therefore, we are not necessarily interested in deriving a representation that allows for the best reconstruction of the signal, but we rather seek to find the invariants of the videos belonging to the same class. These invariants can be thought of as the part of the signal that is stationary across all videos.

Let $\boldsymbol{V} = [\boldsymbol{V}_1, \cdots, \boldsymbol{V}_N]$ be the $D \times (\sum_{i=1}^{N} m_i)$ matrix containing the N training videos of one specific class, where $\boldsymbol{V}_i = [\boldsymbol{v}_{i,1}, \cdots, \boldsymbol{v}_{i,m_i}]$ represents one video in that class, with $\boldsymbol{v}_{i,j} \in \mathbb{R}^D$ the vector of image features for the j^{th} frame of video i. Following SSA, we assume that $\boldsymbol{V}$ is generated by a linear combination of d stationary sources and $D-d$ non-stationary sources, i.e., $\boldsymbol{V} = \boldsymbol{A}\boldsymbol{S}$ in Eq. 1. Taking each video in the class as one epoch, we then search for a de-mixing matrix $\boldsymbol{W}$, related to $\boldsymbol{A}$ by Eq. 2, such that the projections of the mean and covariance of each video are as similar to each other as possible. This can be expressed in terms of the KL divergence, as shown in Eq. 3. Here, to avoid the limitations of SSA discussed in Section 3, we rely on Analytic SSA (Hara et al., 2012).

More specifically, by expanding the KL divergence in Eq. 3, $\boldsymbol{W}$ can be obtained by solving the problem

$$\begin{aligned}\min_{\boldsymbol{W}}\ & \frac{1}{N}\sum_{i=1}^{N}\Big\{\|\boldsymbol{W}(\boldsymbol{\mu}_i - \bar{\boldsymbol{\mu}})\|^2 - \mathrm{logdet}(\boldsymbol{W}\boldsymbol{\Sigma}_i\boldsymbol{W}^T)\Big\} \\ \text{s.t.}\ & \boldsymbol{W}\bar{\boldsymbol{\Sigma}}\boldsymbol{W}^T = \boldsymbol{I}\,,\end{aligned} \tag{4}$$

where

$$\boldsymbol{\mu}_i = \frac{1}{m_i}\sum_{j=1}^{m_i}\boldsymbol{v}_{i,j}\ ,\ \Sigma_i = \frac{1}{m_i-1}\sum_{j=1}^{m_i}(\boldsymbol{v}_{i,j}-\boldsymbol{\mu}_i)(\boldsymbol{v}_{i,j}-\boldsymbol{\mu}_i)^T,$$

$\bar{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\mu}_i$, and $\bar{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\Sigma}_i$.

This problem is non-convex and therefore hard to solve (Hara et al., 2012). However, its objective function can be replaced with a convex upper bound. To this end, the logdet term is approximated with its second order Taylor expansion $\tilde{f}(\boldsymbol{W},\boldsymbol{W}_*)$ near the optimal solution $\boldsymbol{W}_*$. Although this optimal solution is unknown, the constraints $\boldsymbol{W}_*\bar{\boldsymbol{\Sigma}}\boldsymbol{W}_* = \boldsymbol{I}$ and $\boldsymbol{W}_*\boldsymbol{\Sigma}_i\boldsymbol{W}_* = \boldsymbol{I}$ make the dependency on $\boldsymbol{W}_*$ disappear. As shown in (Hara et al., 2012), this yields the bound

$$\tilde{f}(\boldsymbol{W},\boldsymbol{W}_*) \leq \frac{2}{N}\sum_{i=1}^{N}\mathrm{Tr}\left(\boldsymbol{W}(\boldsymbol{\Sigma}_i-\bar{\boldsymbol{\Sigma}})\bar{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{\Sigma}_i-\bar{\boldsymbol{\Sigma}})\boldsymbol{W}^T\right).$$

Since the other term in the objective is quadratic, this bound makes it possible to approximate the problem in Eq. 4 as

$$\begin{aligned}\min_{\boldsymbol{W}}\ & \mathrm{Tr}(\boldsymbol{W}\boldsymbol{C}\boldsymbol{W}^T)\\ \text{s.t.}\ & \boldsymbol{W}\bar{\boldsymbol{\Sigma}}\boldsymbol{W}^T = \boldsymbol{I}\ ,\end{aligned} \tag{5}$$

where

$$\boldsymbol{C} = \frac{1}{N}\sum_{i=1}^{N}\left[\boldsymbol{\mu}_i\boldsymbol{\mu}_i^T + 2\boldsymbol{\Sigma}_i\bar{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\Sigma}_i\right] - \bar{\boldsymbol{\mu}}\bar{\boldsymbol{\mu}}^T - 2\bar{\boldsymbol{\Sigma}}\ . \tag{6}$$

A solution to this problem can be obtained by solving the generalized eigenvalue problem $\boldsymbol{C}\boldsymbol{\varphi} = \lambda\bar{\boldsymbol{\Sigma}}\boldsymbol{\varphi}$, and taking $\boldsymbol{W} = [\boldsymbol{\varphi}_1\cdots\boldsymbol{\varphi}_d]^T$ as the d generalized eigenvectors with smallest eigenvalues. Details of the full derivation can be found in (Hara et al., 2012).

4.2. Kernel Stationary Subspace Analysis

Analytic SSA is a linear method in nature. As such, it cannot model nonlinear mappings between the video signal and its stationary parts. To overcome this limitation, we introduce a kernelized version of SSA, which, as all kernel methods, boils down to performing SSA in a high-dimensional feature space.

Let $\Phi : \mathbb{R}^D \to \mathcal{H}$ be the function mapping a frame $\boldsymbol{v}_{i,j}$ to a high-dimensional feature space. We can write the means and covariances required to encode stationarity in $\mathcal{H}$ as

$$\begin{aligned}\boldsymbol{\mu}_i^\Phi &= \frac{1}{m_i}\sum_{j=1}^{m_i}\Phi(\boldsymbol{v}_{i,j})\ ,\\ \boldsymbol{\Sigma}_i^\Phi &= \frac{1}{m_i-1}\sum_{j=1}^{m_i}(\Phi(\boldsymbol{v}_{i,j})-\boldsymbol{\mu}_i^\Phi)(\Phi(\boldsymbol{v}_{i,j})-\boldsymbol{\mu}_i^\Phi)^T\ ,\end{aligned}$$

as well as $\bar{\boldsymbol{\mu}}^\Phi = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\mu}_i^\Phi$ and $\bar{\boldsymbol{\Sigma}}^\Phi = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\Sigma}_i^\Phi$. Following the standard approach to kernelizing an algorithm, we represent the projection $\boldsymbol{W}$ as a linear combination of the examples in $\mathcal{H}$, which can be expressed as $\boldsymbol{W} = \boldsymbol{\alpha}\Phi(\boldsymbol{V})^T$, with $\boldsymbol{\alpha}$ unknown. By making use of a kernel function k, such that $k(\boldsymbol{v}_{i,j},\boldsymbol{v}_{i',j'}) = \Phi(\boldsymbol{v}_{i,j})^T\Phi(\boldsymbol{v}_{i',j'})$, we can define

$$\boldsymbol{W}\boldsymbol{\mu}_i^\Phi = \boldsymbol{\alpha}\left(\frac{1}{m_i}\sum_j\boldsymbol{k}(\boldsymbol{V},\boldsymbol{v}_{i,j})\right) \triangleq \boldsymbol{\alpha}\tilde{\boldsymbol{k}}_i\ , \tag{7}$$

$$\boldsymbol{W}\bar{\boldsymbol{\mu}}^\Phi = \boldsymbol{\alpha}\left(\frac{1}{N}\sum_i\tilde{\boldsymbol{k}}_i\right) \triangleq \boldsymbol{\alpha}\bar{\boldsymbol{k}}\ , \tag{8}$$

$$\boldsymbol{W}\boldsymbol{\Sigma}_i^\Phi\boldsymbol{W} \triangleq \boldsymbol{\alpha}\tilde{\boldsymbol{K}}_i\boldsymbol{\alpha}^T\ , \tag{9}$$

$$\boldsymbol{W}\bar{\boldsymbol{\Sigma}}^\Phi\boldsymbol{W} = \boldsymbol{\alpha}\left(\frac{1}{N}\sum_i\tilde{\boldsymbol{K}}_i\right)\boldsymbol{\alpha}^T \triangleq \boldsymbol{\alpha}\bar{\boldsymbol{K}}\boldsymbol{\alpha}^T\ , \tag{10}$$

where $\tilde{\boldsymbol{K}}_i = \frac{1}{m_i-1}\sum_j(\boldsymbol{k}(\boldsymbol{V},\boldsymbol{v}_{i,j})-\tilde{\boldsymbol{k}}_i)(\boldsymbol{k}(\boldsymbol{V},\boldsymbol{v}_{i,j})-\tilde{\boldsymbol{k}}_i)^T$.

As shown in supplementary material, we can then derive a similar bound as in Analytic SSA, which now takes the form

$$\tilde{f}(\boldsymbol{\alpha},\boldsymbol{\alpha}_*) \leq \frac{2}{N}\sum_{i=1}^{N}\mathrm{Tr}\left(\boldsymbol{\alpha}(\tilde{\boldsymbol{K}}_i-\bar{\boldsymbol{K}})\bar{\boldsymbol{K}}^{-1}(\tilde{\boldsymbol{K}}_i-\bar{\boldsymbol{K}})\boldsymbol{\alpha}^T\right).$$

This, in conjunction with the definitions of Eqs. 7-10, lets us re-write the problem in Eq. 5 as

$$\begin{aligned}\min_{\boldsymbol{\alpha}}\ & \mathrm{Tr}\left(\boldsymbol{\alpha}\boldsymbol{C}^K\boldsymbol{\alpha}^T\right)\\ \text{s.t.}\ & \boldsymbol{\alpha}\bar{\boldsymbol{K}}\boldsymbol{\alpha}^T = \boldsymbol{I}\ ,\end{aligned} \tag{11}$$

with $\boldsymbol{C}^K = \frac{1}{N}\sum_{i=1}^{N}\tilde{\boldsymbol{k}}_i\tilde{\boldsymbol{k}}_i^T + 2\tilde{\boldsymbol{K}}_i\bar{\boldsymbol{K}}^{-1}\tilde{\boldsymbol{K}}_i - \bar{\boldsymbol{k}}\bar{\boldsymbol{k}}^T - 2\bar{\boldsymbol{K}}$. The solution to this problem can be obtained by solving the generalized eigenvalue problem $\boldsymbol{C}^K\boldsymbol{\varphi} = \lambda\bar{\boldsymbol{K}}\boldsymbol{\varphi}$.

A drawback of KSSA is that it relies on the computation of the inverse of $\bar{\boldsymbol{K}}$, which is a rank-deficient matrix. Indeed, it can be shown that the individual $\tilde{\boldsymbol{K}}_i$ matrices have rank $m_i - 1$, which implies that the maximum rank of $\bar{\boldsymbol{K}}$ is $\sum_{i=1}^{N}m_i - N$. This problem can be alleviated by replacing $\bar{\boldsymbol{K}}$ with $\bar{\boldsymbol{K}} + \varepsilon\boldsymbol{I}$ with a small ε. This, however, does not address the high computational complexity of KSSA due to the inversion and multiplications of large matrices. Next, we introduce a solution to overcome these weaknesses.

4.3. Non-Linear Stationary Subspace Analysis

To address the limitations of KSSA, we observe that the presence of $\bar{\boldsymbol{K}}^{-1}$ in the final problem is related to the presence of $\bar{\boldsymbol{\Sigma}}^{-1}$ in the solution of Analytic SSA. We therefore propose to transform the data in such a way that the average covariance matrix $\bar{\boldsymbol{\Sigma}}$ becomes identity. To keep the benefits of working in a high-dimensional feature space, we search for such a transformation in $\mathcal{H}$. Note that this does not truly restrict

the solutions of our approach, since $\boldsymbol{W}$ was originally only defined up to a linear transformation.

More specifically, we search for a transformation $\boldsymbol{P}$ such that $\boldsymbol{P}\bar{\boldsymbol{\Sigma}}^{\Phi}\boldsymbol{P}^T = \boldsymbol{I}$. Equivalently, this can be expressed as the eigenvalue problem $\boldsymbol{U}\bar{\boldsymbol{\Sigma}}^{\Phi}\boldsymbol{U}^T = \boldsymbol{\Lambda}$. As before, $\boldsymbol{U}$ can be expressed as a linear combination of the examples in $\mathcal{H}$, i.e., $\boldsymbol{U} = \boldsymbol{\beta}\Phi(\boldsymbol{V})^T$. By exploiting the definitions introduced in Section 4.2, we obtain the eigenvalue problem

$$\boldsymbol{\beta}\bar{\boldsymbol{K}}\boldsymbol{\beta}^T = \boldsymbol{\Lambda}\ , \tag{12}$$

from which we can obtain $\boldsymbol{\beta}$. This yields the transformation $\boldsymbol{P} = \boldsymbol{\Lambda}^{-1/2}\boldsymbol{\beta}\Phi(\boldsymbol{V})^T$, which lets us project the data as

$$\boldsymbol{V}^P = \boldsymbol{P}\Phi(\boldsymbol{V}) = \boldsymbol{\Lambda}^{-1/2}\boldsymbol{\beta}\boldsymbol{K}\ , \tag{13}$$

where $\boldsymbol{K}$ is the kernel matrix of all the training videos. In practice, we keep the eigenvectors of $\bar{\boldsymbol{K}}$ that account for 98% of its variance.

Since the projected data is already expressed in terms of kernels (i.e., does not depend on the mapping Φ), we can directly apply SSA to it. By defining $\boldsymbol{\mu}_i^P = \frac{1}{m_i}\sum_j \boldsymbol{v}_{i,j}^P$ and $\boldsymbol{\Sigma}_i^P = \frac{1}{m_i-1}\sum_j(\boldsymbol{v}_{i,j}^P - \boldsymbol{\mu}_i^P)(\boldsymbol{v}_{i,j}^P - \boldsymbol{\mu}_i^P)^T$, we can re-write the problem in Eq. 3 as

$$\begin{aligned}\min_{\boldsymbol{W}} \sum_{i=1}^{N} \mathrm{KL}\left(\mathcal{N}(\boldsymbol{W}\boldsymbol{\mu}_i^P, \boldsymbol{W}\boldsymbol{\Sigma}_i^P\boldsymbol{W}^T)||\mathcal{N}(\boldsymbol{0},\boldsymbol{I})\right)\\ \text{s.t. } \boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}\ ,\end{aligned} \tag{14}$$

where we further assumed that $\bar{\boldsymbol{\mu}}^{\Phi}$, and thus $\bar{\boldsymbol{\mu}}^P$, is 0, which can be achieved by a centering procedure described in supplementary material. Following Analytic SSA, non-linear SSA (NLSSA) can be formulated as

$$\begin{aligned}\min_{\boldsymbol{W}} \ \mathrm{Tr}\left(\boldsymbol{W}\boldsymbol{C}^P\boldsymbol{W}^T\right)\\ \text{s.t. } \boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}\ .\end{aligned} \tag{15}$$

where $\boldsymbol{C}^P = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\mu}_i^P\boldsymbol{\mu}_i^{P\,T} + 2\boldsymbol{\Sigma}_i^P\boldsymbol{\Sigma}_i^P - 2\boldsymbol{I}$. This corresponds to the eigenvalue problem $\boldsymbol{C}^P\boldsymbol{\varphi} = \lambda\boldsymbol{\varphi}$. Algorithm 1 shows the pseudo-code for NLSSA.

4.4. Computational Complexity

The solutions of Analytic SSA, KSSA and NLSSA are obtained by solving (generalized) eigenvalue problems, whose computational complexity is $O(s^3)$ for $s \times s$ matrices. Computing $\boldsymbol{C}$ and $\bar{\boldsymbol{\Sigma}}$ in Eq. 6 requires $O((2N+1)D^3)$ and $O(NFD^2)$ operations, respectively, where $F = \sum_{i=1}^{N} m_i$. This yields $O((2N+2)D^3 + NFD^2)$ flops for Analytic SSA. For KSSA, computing $\boldsymbol{C}^K$ and $\bar{\boldsymbol{K}}$ requires $O((2N+1)F^3)$ and $O(F^3)$ operations, and hence $O((2N+2)F^3)$ flops overall. Finally for NLSSA, computing $\boldsymbol{P}$ and $\boldsymbol{C}^P$ requires $O(pF^2 + F^3)$ and $O(2Np^3)$, respectively, where p denotes the number of eigenvectors of $\bar{\boldsymbol{K}}$ retained to generate $\boldsymbol{P}$. As a result, the overall computational complexity of NLSSA is $O(pF^2 + F^3 + 2Np^3)$. The difference in computational cost between KSSA and NLSSA mostly comes from the factor $(2N+2)$ in KSSA.

Algorithm 1 : Non-Linear Stationary Subspace Analysis

Input:

- N training videos $\boldsymbol{V} = [\boldsymbol{V}_1, \cdots, \boldsymbol{V}_N]$ belonging to the same class.
- The dimensionality d of the stationary subspace.

Output:

- The stationary projection matrix $\boldsymbol{W}$.

1: Center the data in feature space such that $\bar{\boldsymbol{\mu}}^{\Phi} = \boldsymbol{0}$.
2: Compute the matrix $\bar{\boldsymbol{K}}$ from Eq. 10.
3: Compute the eigenvectors $\boldsymbol{\beta}$ and eigenvalues $\boldsymbol{\Lambda}$ of $\bar{\boldsymbol{K}}$.
4: Transform the data as $\boldsymbol{V}^P = (\boldsymbol{\Lambda})^{-1/2}\boldsymbol{\beta}\boldsymbol{K}$.
5: Compute $\boldsymbol{\mu}_i^P$ and $\boldsymbol{\Sigma}_i^P$ for each video $\boldsymbol{V}_i^P$.
6: Compute the matrix $\boldsymbol{C}^P$ in Eq. 15.
7: Compute $\boldsymbol{W}$ by solving the eigenvalue problem corresponding to Eq. 15.

4.5. Video Classification with KSSA and NLSSA

We now describe how our algorithms can be used for video classification. Let $\boldsymbol{V}^c = [\boldsymbol{V}_1^c, \cdots, \boldsymbol{V}_N^c]$ be the matrix of training videos for class c. For each class c, we use either KSSA to compute the matrix $\boldsymbol{\alpha}_c$ from Eq. 11, or NLSSA to obtain the projection $\boldsymbol{W}_c$ from Eq. 15. Since stationarity is expressed in terms of similarities with respect to the average over the epochs of the projected means and covariances , we use these quantities to represent each class. For KSSA, we define

$$\boldsymbol{m}^c = \boldsymbol{\alpha}_c\bar{\boldsymbol{k}}^c \ \text{ and } \ \boldsymbol{M}^c = \boldsymbol{\alpha}_c\bar{\boldsymbol{K}}^c\boldsymbol{\alpha}_c^T\ , \tag{16}$$

where $\bar{\boldsymbol{k}}^c$ and $\bar{\boldsymbol{K}}^c$ can be obtained from Eqs. 8 and 10. For NLSSA, because of the transformation of the data in feature space, we simply have $\boldsymbol{m}^c = \boldsymbol{0}$ and $\boldsymbol{M}^c = \boldsymbol{I}$.

Given a query video of m frames $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_m]$, we perform classification by projecting $\boldsymbol{Q}$ in the subspace of each class independently. We make use of the mean and covariance of the projected data. With KSSA, this yields

$$\boldsymbol{h}^c = \boldsymbol{\alpha_c}\left(\frac{1}{m}\sum_j \boldsymbol{k}(\boldsymbol{V}^c, \boldsymbol{q}_j)\right)\ ,\ \boldsymbol{H}^c = \boldsymbol{\alpha_c}\tilde{\boldsymbol{K}}_Q^c\boldsymbol{\alpha_c}^T\ ,$$

where $\tilde{\boldsymbol{K}}_Q^c$ is obtained similarly as $\tilde{\boldsymbol{K}}_i$ in Eq. 9. For NLSSA, the data is first centered in kernel space according to the training $\bar{\boldsymbol{\mu}}^{\Phi,c}$, and then transformed as $\boldsymbol{Q}^{P,c} = \boldsymbol{\Lambda}_c^{-1/2}\boldsymbol{\beta}_c^T\boldsymbol{K}_Q^c$, where $\boldsymbol{K}_Q^c$ is the kernel matrix computed between the training data in class c and

the query video. We then compute the corresponding mean and covariance of the projected data as

$$\boldsymbol{h}^c = \boldsymbol{W}_c \left(\frac{1}{m} \sum_j \boldsymbol{q}_j^{P,c} \right) \ , \ \boldsymbol{H}^c = \boldsymbol{W}_c \boldsymbol{\Sigma}_Q^c \boldsymbol{W}_c^T \ ,$$

where $\boldsymbol{\Sigma}_Q^c$ is the covariance of the transformed data. Classification is achieved by determining which class c gives the representation $(\boldsymbol{h}^c, \boldsymbol{H}^c)$ most similar to $(\boldsymbol{m}^c, \boldsymbol{M}^c)$. Since SSA employs the KL-divergence as a similarity measure, it comes as a natural choice for classification. We therefore search for the class that yields the lowest value $\mathrm{KL}\left(\mathcal{N}(\boldsymbol{h}^c, \boldsymbol{H}^c) || \mathcal{N}(\boldsymbol{m}^c, \boldsymbol{M}^c)\right)$.

While our final representation may not be ideal for a classifier such as SVM, it is very well-suited for the simple classifier that we used. Furthermore, it would not prevent us from using other discriminative classifiers, such as the Minimax Probability Machine of (Lanckriet et al., 2003). Note that, given $\boldsymbol{W}$, we could also use other representations to exploit SVMs. We believe that a full study of other possible classifiers goes beyond the scope of this paper.

5. Experiments

We evaluate our approach on the tasks of traffic scene classification, dynamic texture recognition and action recognition, and compare its performance against the state-of-the art methods in each task. To demonstrate the importance of modeling stationarity, we also compare our results with those obtained by replacing NLSSA with PCA and kernel PCA in our approach. Performance is measured as the average accuracy over all classes. In all the experiments, we used the linear kernel and the RBF kernel. We report the classification accuracies obtained with $N-1$ stationary components, where N is the number of videos in the class. According to (Hara et al., 2012), to avoid spurious stationary signals, the number of stationary directions should be less than the number of epochs (i.e., at most $N-1$). This gives us a systematic way of defining the number of stationary components, which proved very effective in practice. In our experiments we relied on 2D descriptors computed in individual frames. However, our method is not restricted to this choice. In particular, we could easily take temporal information into account in the features by concatenating the features of consecutive frames, or by making use of 3D spatio-temporal descriptors.

5.1. Synthetic Data

As a first experiment, we compare the performance of our NLSSA and KSSA algorithms with the linear ASSA method using synthetic data generated following a protocol similar to the one in (Müller et al., 2011): We first generated random stationary and non-stationary sources. We then mixed these sources, but used a non-linear mixing function based on an exponential mapping instead of the linear mixing matrix used in (Müller et al., 2011). Each epoch is thus a non-linear mixture of stationary sources (shared by all epochs) and non-stationary sources (specific to each epoch). We set the dimensionality of the observed signal to $D = 5$, the number of stationary sources to $d = 2$ and the number of samples in each epoch to $m = 5$. We varied the number of epochs N from 5 to 50.

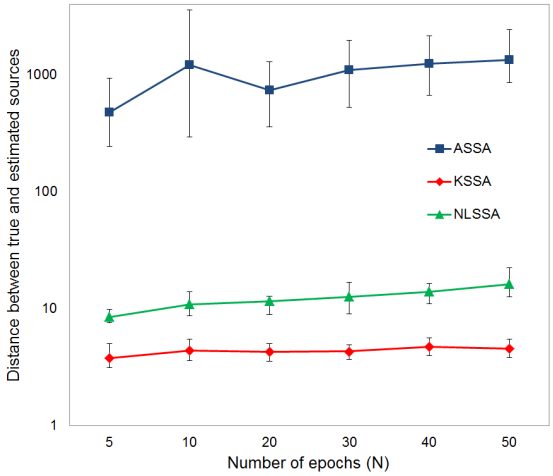

Figure 2. **Extracting stationary signal.** Distance between the extracted and true stationary sources for different epoch numbers.

We applied ASSA, KSSA and NLSSA to extract the stationary sources from the observed signals. To evaluate the quality of the results, we measured the L2 distance between the estimated stationary sources and the ground-truth ones. Fig. 2 shows the mean, min and max distance error over 10 splits as a function of N. Note that the error is shown on a log-scale axis. Our non-linear approaches with an RBF kernel yield much lower distances between the estimated and the true stationary sources than ASSA. Note that NLSSA yields a slightly higher error than KSSA. However, as will be seen in the next experiments, this entails no loss of accuracy on the classification results. Furthermore, NLSSA yields much faster runtimes than KSSA.

5.2. Traffic Scene Classification

As a first experiment on real data, we used the UCSD traffic video dataset (Chan & Vasconcelos, 2005) to classify videos based on the density of traffic. The dataset is partitioned into 3 classes corresponding to

Algorithm	Accuracy
LDS (Sankaranarayanan et al., 2010)	$87.50\% \pm 0.87$
CS-LDS (Sankaranarayanan et al., 2010)	$89.06\% \pm 2.16$
KL-SVM (Chan & Vasconcelos, 2005)	95%
PCA	$90.55\% \pm 0.08$
Kernel PCA	$91.34\% \pm 4.58$
NLSSA - Linear Kernel	$93.30\% \pm 3.27$
NLSSA - RBF Kernel	$94.49\% \pm 2.02$
KSSA - RBF Kernel	$94.49\% \pm 3.25$

Table 1. **Traffic scene classification.** Average accuracies of our approach and several baselines on the UCSD dataset.

Algorithm	Accuracy
DL-PEGASOS (Ghanem & Ahuja, 2010)	63.7%
DFS (Xu et al., 2011)	89.9%
PCA	$88.22\% \pm 0.32$
Kernel PCA	$89.72\% \pm 1.2$
NLSSA - Linear Kernel	$89.97\% \pm 0.49$
NLSSA - RBF Kernel	$94.22\% \pm 0.64$

Table 2. **Dynamic texture recognition.** Average recognition accuracies of our approach and several baselines on DynTex++.

light, medium and heavy highway traffic congestion. It contains a total of 254 video sequences: 165 sequences of light traffic, 45 of medium traffic, and 44 of heavy traffic. Each video contains between 42 and 52 frames of size 320×240, which, following common practice (Sankaranarayanan et al., 2010), are normalized and downsized to 48×48 greyscale images. We compare the performance of our two algorithms (KSSA and NLSSA) against the Linear Dynamical Systems model (LDS), compressive sensing LDS (CS-LDS) (Sankaranarayanan et al., 2010), and Probabilistic Kernels (KL-SVM) (Chan & Vasconcelos, 2005), and against our PCA and kernel PCA baselines. Note that the baselines learn an LDS model from raw pixel values and use the parameters of this model for classification. Since our formulation does not really let us make use of such representation, we utilized HOG features with 28 orientation bins as image descriptors, which are still extracted from the same raw pixel values. We employed the four train/test splits of (Sankaranarayanan et al., 2010).

Table 1 shows that our approach outperforms the baselines, with the exception of KL-SVM which achieves similar accuracy. Note that KSSA and NLSSA yield the same performance, which shows that the small loss of accuracy in the estimated signal noticed in Section 5.1 leaves classification accuracy unaffected. However, training with KSSA took $8070.2s$, as opposed to $135.1s$ with NLSSA. Since here on average $N = 60$, this ratio reflects the analysis in Section 4.4, and shows the benefit of NLSSA over KSSA. Henceforth, we therefore only present NLSSA results.

5.3. Dynamic Texture Recognition

Dynamic textures (DT) are video sequences depicting phenomena such as the motion of water, fire, clouds and smoke, that are known to exhibit stationary patterns. Our method therefore seems a natural choice for DT recognition. For this task, we used the DynTex++ dataset (Ghanem & Ahuja, 2010), which has been widely used for benchmarking DT classification methods. It is composed of 36 dynamic texture classes, each of which contains 100 video sequences of 50 greyscale frames of size 50×50. We used histograms of Local Binary Pattern (LBP) codes (Ahonen et al., 2006) as image features. In particular, we computed a total of 128 different codes for each image based on the relative intensities of a pixel and its 7 neighbors. Our choice of LBP as features in this experiment was motivated by the fact that other baselines used these features, which have shown to perform well in texture classification due to their robustness to monotonic illumination changes. Following (Ghanem & Ahuja, 2010), 50 sequences from each class were randomly selected as training data and the remaining sequences used for testing. We report the mean accuracy and standard deviation over 10 such splits.

We compare our results with our PCA and kernel PCA baselines, as well as with the state-of-the-art methods DL-Pegasos (Ghanem & Ahuja, 2010) and DFS (Xu et al., 2011). The classification accuracies are shown in Table 2. Our approach with an RBF kernel achieves the highest accuracy, which confirms the importance of having a non-linear mapping. Note that the results of DL-PEGASOS (Ghanem & Ahuja, 2010) (63.7%) were also obtained with LBP features. This clearly shows that the superiority of our results is not due to the choice of image features, but to the model itself.

5.4. Action Recognition

To evaluate the performance of our approach on action recognition, we used three publicly available datasets: KTH (Schuldt et al., 2004) , Ballet (Wang & Mori, 2009) and UCF sports (Rodriguez et al., 2008).

The KTH dataset consists of six different human actions performed by 25 subjects in four different scenarios: outdoor, outdoor with scale variation, outdoor with different clothes and indoor. The dataset contains 2391 sequences in total. We relied on the standard training/testing splits of (Castrodad & Sapiro, 2012), and used HOG features with 128 orientation bins as image descriptors. Recognition accuracies are

Algorithm	Accuracy
Subspace Forest (O'Hara & Draper, 2012)	97.9%
SM (Castrodad & Sapiro, 2012)	97.9%
KFDA (Campos et al., 2011)	93.52%
PCA	62.8%
Kernel PCA	75.3%
NLSSA - Linear Kernel	99.07%
NLSSA - RBF Kernel	98.87%

Table 3. **Action recognition: KTH.** Recognition accuracies of our approach and state-of-the-art baselines.

Algorithm	Accuracy
GDA (Hamm & Lee, 2008)	$78.05\% \pm 2.9$
KAHM (Cevikalp & Triggs, 2010)	$79.71\% \pm 2.3$
KAHM with HoG	$75.4\% \pm 2.6$
PCA	$49.38\% \pm 1.6$
Kernel PCA	$77.11\% \pm 2.4$
NLSSA - Linear Kernel	$69.51\% \pm 2.2$
NLSSA - RBF Kernel	$83.12\% \pm 2.7$

Table 4. **Action recognition: Ballet.** Recognition accuracies of our approach and state-of-the-art baselines.

reported in Table 3, where it can be seen that our approach with a linear kernel performs best. In contrast with previous experiments, our PCA and kernel PCA baselines perform very poorly. This demonstrates the benefits of exploiting stationarity for this task.

The Ballet dataset (Wang & Mori, 2009) contains 44 real video sequences with significant intra-class variations in terms of spatial and temporal scales, clothing and movements. It consists of 8 actions such as jumping, turning, leg swinging and standing still performed by 3 subjects. While some previous experiments with the Ballet dataset (Wang & Mori, 2009) tackle the problem of action recognition in still images, we perform recognition from full video sequences. For each action, the samples were randomly split into training and testing sets of similar sizes. We utilized HOG features with 28 orientation bins as image representation and used epochs of 12 frames. In Table 4, we report the average classification accuracy and standard deviation over 10 splits for the state-of-art kernel version of affine hull set matching (KAHM) (Cevikalp & Triggs, 2010), Grassmann discriminant analysis (GDA) (Hamm & Lee, 2008) and our approach. Note that our approach with an RBF kernel outperforms the baselines. We also compared our method against using the same HOG features with KAHM. Note that the classification accuracy of KAHM with HOG features decreased to 75.4%. This again shows that improvement over the baselines really comes from our model, and not just from the features we used.

Algorithm	Accuracy
Subspace Forest (O'Hara & Draper, 2012)	91.3%
SM (Castrodad & Sapiro, 2012)	97.3%
KFDA (Campos et al., 2011)	80.00%
PCA	55.62%
Kernel PCA	61.11%
NLSSA - Linear Kernel	89.9%
NLSSA - RBF Kernel	92.32%

Table 5. **Action recognition: UCF Sports.** Accuracies of our approach and state-of-the-art baselines.

The UCF sports dataset (Rodriguez et al., 2008) contains 150 real videos with non-uniform backgrounds and moving camera/subjects. It consists of 10 categories of human actions collected from various sports, such as kicking, lifting weights, running and skateboarding. The number of videos for each action varies from 6 to 22. We used the region of interest provided with the dataset, and, to deal with the small number of examples, duplicated each training video. We employed HOG features with 128 orientation bins for image representation. For testing, we followed a standard leave-one-out (LOO) protocol[1]. Table 5 compares the recognition accuracies of our approach and several baselines. Note that, again, our PCA and kernel PCA baselines perform poorly, which indicates the importance of stationarity. While Sparse Modeling (SM) (Castrodad & Sapiro, 2012) yields a slightly higher accuracy than our approach, it relies on features learnt from data. Replacing our simple HOG features with such features might improve our accuracy and will be a topic of our future work.

6. Conclusion and Future Work

We have proposed an approach to video classification that relies on extracting the stationary parts of the signal of each class from training videos. To this end, we have introduced two methods, KSSA and NLSSA, that allow modeling non-linear mappings between the video data and their stationary parts. Our experiments have shown the importance of exploiting stationarity for video classification, as well as the benefits of accounting for the non-linearity of the signal. A current limitation of our approach is its use of the notion of *weak* stationarity, which can be fooled by the presence of mixtures of non-stationary signals that appear to be stationary. In the future, we therefore intend to study the use of more sophisticated measures of stationarity. We also plan to investigate how subspace clustering ideas could be exploited to model each class with more than one subspace.

[1]The duplicate of the test video was also left out.

References

Ahonen, T., Hadid, A., and Pietikainen, M. Face description with local binary patterns: Application to face recognition. *PAMI*, 2006.

Ali, S. and Shah, M. Human action recognition in videos using kinematic features and multiple instance learning. *PAMI*, 2010.

Bünau, P., Meinecke, F., Király, F., and Müller, K. Finding stationary subspaces in multivariate time series. *Physical Review Letters*, 2009.

Campos, T., Barnard, M., Mikolajczyk, K., Kittler, J., Yan, F., Christmas, W., and Windridge, D. An evaluation of bags-of-words and spatio-temporal shapes for action recognition. In *WACV*, 2011.

Castrodad, A. and Sapiro, G. Sparse modeling of human actions from motion imagery. *IJCV*, 2012.

Cevikalp, H. and Triggs, B. Face recognition based on image sets. In *CVPR*, 2010.

Chan, A. and Vasconcelos, N. Probabilistic kernels for the classification of auto-regressive visual processes. In *CVPR*, 2005.

Chan, A. and Vasconcelos, N. Classifying video with kernel dynamic textures. In *CVPR*, 2007.

Chan, A., Coviello, E., and Lanckriet, G. Clustering dynamic textures with the hierarchical em algorithm. In *CVPR*, 2010.

Chaudhry, R., Ravichandran, A., Hager, G., and Vidal, R. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *CVPR*, 2009.

Doretto, G., Chiuso, A., Wu, Y., and Soatto, S. Dynamic textures. *IJCV*, 2003.

Ghanem, B. and Ahuja, N. Maximum margin distance learning for dynamic texture recognition. In *ECCV*, 2010.

Gu, L., Li, S., and Zhang, H. Learning probabilistic distribution model for multi-view face detection. In *CVPR*, 2001.

Hamm, J. and Lee, D. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *ICML*, 2008.

Hara, S., Kawahara, Y., Washio, T., Bünau, P., Tokunaga, T., and Yumoto, K. Separation of stationary and non-stationary sources with a generalized eigenvalue problem. *Neural networks*, 2012.

Hotta, K. Local co-occurrence features in subspace obtained by kpca of local blob visual words for scene classification. *Pattern Recognition*, 2012.

Knopp, J., Prasad, M., Willems, G., Timofte, R., and Van Gool, L. Hough transform and 3d surf for robust three dimensional classification. In *ECCV*, 2010.

Lanckriet, G., Ghaoui, L., Bhattacharyya, C., and Jordan, M. A robust minimax approach to classification. *JMLR*, 2003.

Le, Q., Zou, W., Yeung, S., and Ng, A. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011.

Long, F., Wu, T., Movellan, J., Bartlett, M., and Littlewort, G. Learning spatiotemporal features by using independent component analysis with application to facial expression recognition. *Neurocomputing*, 2012.

Müller, J., Bünau, P., Meinecke, F., Király, F., and Müller, K. The stationary subspace analysis toolbox. *JMLR*, 2011.

Niebles, J., Wang, H., and Li, F. Unsupervised learning of human action categories using spatial-temporal words. *IJCV*, 2008.

O'Hara, S. and Draper, B. Scalable action recognition with a subspace forest. In *CVPR*, 2012.

Ravichandran, A., Chaudhry, R., and Vidal, R. View-invariant dynamic texture recognition using a bag of dynamical systems. In *CVPR*, 2009.

Rodriguez, M., Ahmed, J., and Shah, M. Action MACH a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008.

Saisan, P., Doretto, G., Wu, Y., and Soatto, S. Dynamic texture recognition. In *CVPR*, 2001.

Sankaranarayanan, A., Turaga, P., Baraniuk, R., and Chellappa, R. Compressive acquisition of dynamic scenes. In *ECCV*, 2010.

Schuldt, C., Laptev, I., and Caputo, B. Recognizing human actions: A local SVM approach. In *ICPR*, 2004.

Sivic, J. and Zisserman, A. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

Thurau, C. and Hlavác, V. Pose primitive based human action recognition in videos or still images. In *CVPR*, 2008.

Tipping, M., Bishop, C., et al. Mixtures of probabilistic principal component analyzers. *Neural computation*, 1999.

Tseng, C., Chen, J., Fang, C., and Lien, J. Human action recognition based on graph-embedded spatio-temporal subspace. *Pattern Recognition*, 2012.

Vidal, R., Ma, Y., and Sastry, S. Generalized principal component analysis (GPCA). *PAMI*, 2005.

Wang, H., Klaser, A., Schmid, C., and Liu, C. Dense trajectories and motion boundary descriptors for action recognition. Research report, INRIA, 2012.

Wang, Y. and Mori, G. Human action recognition by semi-latent topic models. *PAMI*, 2009.

Xu, Y., Quan, Y., Ling, H., and Ji, H. Dynamic texture classification using dynamic fractal analysis. In *ICCV*, 2011.