Compressive Video Classification for Decision Systems with Limited Resources

Pavlos Charalampidis

Computer Science Department

University of Crete

A thesis submitted for the degree of

Master in Science

Heraklion, June 2013

University of Crete
Computer Science Department

Compressive Video Classification for Decision Systems with Limited Resources

Thesis submitted by
Pavlos Charalampidis
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author: _____
Pavlos Charalampidis

Committee approvals: _____
Panagiotis Tsakalides
Professor, Thesis Supervisor

_____
Apostolos Traganitis
Professor, Committee Member

_____
Athanasios Mouchtaris
Assistant Professor, Committee Member

Departmental approval: _____
Angelos Bilas
Professor, Director of Graduate Studies

Heraklion, June 2013

To my family

Acknowledgements

First of all I would like to thank my supervisor, prof. Panagiotis Tsakalides, for his encouragement, patience and guidance during this work.

I am also really grateful to Dr. Giorgos Tzagkarakis and Dr. Gregory Tsagatakis for their constructive ideas, support and technical assistances. Sharing their knowledge and experience made this work possible.

I would like to acknowledge the support, both financial and in facilities, by the Institute of Computer Science (FORTH-ICS). Special thanks to all my colleagues in Telecommunications & Networks Lab and Signal Processing Lab for their encouraging discussions and pleasant atmosphere.

In addition, I would like to thank the State Scholarships Foundation (IKY) for its financial support.

A dedicated thank you to my personal friends in Heraklion that always stood by me both at good and bad times with encouragement and patience.

Last but not least I would like to thank my family for all their support and encouragement that really brought me here.

Abstract

Due to the advent of digital TV and the availability of large video databases the task of automatic video classification has received a great research interest. The objective of video classification is to label a video sequence with its corresponding class, among a predefined set of classes. Typically, full resolution video data is required for the extraction of appropriate features. However, under the case of limited-resource sensing systems, which happens in applications like video surveillance and remote sensing such processing can be computationally and power demanding placing significant burden on the encoder's side. Additionally, a large bandwidth is required to transmit full-resolution data at a base station for further processing.

In this thesis we address the aforementioned problems by exploiting the framework of compressive sensing. Compressive sensing acting simultaneously as a sampling and compression protocol enables the efficient representation and reconstruction of a sparse signal from a set of non-adaptive linear incoherent measurements much fewer than what is described by the Nyquist theorem. Here, we exploit the properties of linear random projections for addressing the problem of video classification without handling the original high-resolution data. In particular, we introduce two compressive video classification systems that work directly in the compressed domain. We assume the scenario of a video classification system equipped with a single-pixel camera that can directly acquire compressive samples in the optical domain.

In the first system the compressively sampled frames are directly used as features along with an appropriate decision rule to classify a query sequence. In the second system a block-based compressive acquisition model is used together with dictionary learning, and a support vector machine (SVM) with a spatio-temporal pyramid matching kernel for the classification phase. The proposed methods are evaluated using a subset of the UCF50 activity recognition dataset. The results verify the efficiency of the proposed video classification systems and illustrate that features based on compressive measurements, in conjunction with an appropriate decision rule, results in an

effective video classification scheme, which meets the constraints of systems with limited resources. In addition, the comparison with a conventional video classification scheme that exploits the full-resolution video data illustrates that, although only a small percentage of the original data is used in the compressive video classification systems, no significant degradation in performance is observed.

Περίληψη

Εξαιτίας της έλευσης της ψηφιακής τηλεόρασης και της διαθεσιμότητας μεγάλων βάσεων δεδομένων βίντεο η αυτόματη κατηγοριοποίηση βίντεο έχει γίνει αντικείμενο ερευνητικής μελέτης. Ο στόχος της κατηγοριοποίησης βίντεο είναι η αντιστοίχηση μίας ακολουθίας σε μία κλάση ανάμεσα σε ένα προκαθορισμένο σύνολο κλάσεων. Συνήθως δεδομένα πλήρους ανάλυσης απαιτούνται για την εξαγωγή των κατάλληλων χαρακτηριστικών. Παρ΄ όλα αυτά, στην περίπτωση συστημάτων περιορισμένων πόρων, όπως αυτά σε εφαρμογές βιντεοπαρακολούθησης ή τηλεπισκόπησης αυτή η επεξεργασία μπορεί να αποδειχθεί υπολογιστικά και ενεργειακά απαιτική επιβαρύνοντας ιδιαίτερα την πλευρά του κωδικοποιητή. Επιπλέον, μεγάλο εύρος ζώνης απαιτείται για την αποστολή των δεδομένων πλήρους ανάλυσης σε ένα σταθμό βάσης για επιπλέον επεξεργασία.

Στην παρούσα εργασία αντιμετωπίζουμε τα προαναφερθέντα προβλήματα εντός του πλαισίου της συμπιεστικής δειγματοληψίας. Η συμπιεστική δειγματοληψία λειτουργώντας ταυτόχρονα ως πρωτόκολλο δειγματοληψίας και συμπίεσης επιτρέπει την αποδοτική αναπαράσταση και ανακατασκευή ενός αραιού σήματος από ένα σύνολο μη-προσαρμοσμένων γραμμικών μετρήσεων πολύ λιγότερων από αυτές που προβλέπει το θεώρημα του Nyquist. Στη συγκεκριμένη περίπτωση αξιοποιούμε τις ιδιότητες των γραμμικών τυχαίων προβολών στο πρόβλημα της κατηγοριοποίησης βίντεο χωρίς τη διαχείριση των αρχικών δεδομένων υψηλής ανάλυσης. Συγκεκριμένα παρουσιάζουμε δύο συστήματα συμπιεστικής κατηγοριοποίησης βίντεο τα οποία δουλεύουν απ΄ ευθείας στα συμπιεσμένα δεδομένα. Θεωρούμε την περίπτωση ενός συστήματος που διαθέτει μία κάμερα ενός πιξελ η οποία μπορεί να καταγράψει συμπιεστικά δείγματα στο οπτικό πεδίο.

Στο πρώτο σύστημα τα συμπιεστικά δειγματοληπτημένα καρέ χρησιμοποιούνται κατ΄ ευθείαν ως χαρακτηριστικά σε συνδυασμό με έναν κατάλληλο κανόνα απόφασης για την κατηγοριοποίηση μιας άγνωστης ακολουθίας. Στο δεύτερο σύστημα χρησιμοποιείται ένα μοντέλο συμπιεστικής δειγματοληψίας σε μπλοκ του καρέ μαζί με εκμάθηση λεξικού και έναν ταξινομητή Μηχανής Εδραίων Διανυσμάτων (SVM) με συνάρτηση πυρήνα χωροχρονικής πυραμίδας για τη φάση της ταξινόμησης. Οι προτεινόμενες μέθοδοι αξιολογούνται

χρησιμοποιώντας ένα υποσύνολο της βάσης αναγνώρισης δραστηριότητας UCF50. Τα αποτελέσματα επιβεβαιώνουν την αποδοτικότητα των συστημάτων και δείχνουν ότι χαρακτηριστικά που βασίζονται στις συμπιεστικές μετρήσεις σε συνδυασμό με κατάλληλους κανόνες απόφασης οδηγούν σε αποδοτικό σχήμα κατηγοριοποίησης, το οποίο πληροί τους περιορισμούς των συστημάτων περιορισμένων πόρων. Επιπλέον, από τη σύγκριση με ένα συμβατικό σύστημα κατηγοριοποίησης βίντεο που αξιοποιεί τα δεδομένα πλήρους ανάλυσης φαίνεται ότι παρά τη χρήση ενός μικρού μόνο ποσοστού των αρχικών δεδομένων στα συστήματα συμπιεστικής κατηγοριοποίησης αυτό δεν προκαλεί σημαντική μείωση στην απόδοση.

Contents

List of Figures

Chapter 1

Introduction

Modern high-resolution sensing devices, with signal processing and communication capabilities largely based on the seminal Shannon and Nyquist studies, have enabled the acquisition, storage, and transmission of ever increasing amounts of data. Apart from reconstructing the original signal, several tasks such as detection and classification are also of paramount importance in signal processing applications. Focusing on the classification task, the problem consists in finding the correct class of the sensed signal among a set of candidate classes.

An area which could benefit significantly by the introduction of efficient computational models is *video classification*. With the advent of digital TV and the availability of large digital video databases, it is desirable to classify and retrieve high-resolution video content automatically. Moreover, in a remote sensing application, the potentially limited power, storage, and bandwidth resources require the efficient representation of the video content in a precise and compact way for further decision making. A characteristic example in the later case is the design of unmanned aerial vehicles (UAVs) and terrestrial sensor networks, which have been increasingly used in surveillance and reconnaissance applications, where the captured video is exploited to classify a target of interest.

Consequently, a lot of research effort has been put on the development of automatic video classification algorithms. As any other classification task, the effectiveness of a video classification system is determined by two main factors, namely, i) the quality of extracted *features* that comprise the *video sequence signature* and ii) the selected *classifier* that is used for the final label assignment to a query video sequence.

Features used for the purpose of video classification generally are drawn from three modalities, namely text, audio and visual. However, based on the fact that humans

receive most information through their sense of vision, the majority of the approaches rely on visual elements, that are commonly drawn from keyframes representing a shot [1]. These include color-based features like color histograms or color correlograms that can capture the global and/or local distribution of colors in a video frame [2, 3], motion-based features which include computation of optical flow in global or local histograms or direct use of motion vectors in case of MPEG videos [4, 5, 6] as well as gradient-based features that provide appearance, shape and object information of video sequences [2, 7, 8]. As regards classification techniques, several are also employed in conjunction with the appropriate features, including support vector machines (SVM), hidden Markov models (HMM) and Bayesian methods based on maximum a posteriori (MAP) estimation.

However, the aforementioned procedures require the full resolution video data for the generation of the descriptors, which is highly inefficient for the case of limited-resource sensing systems. In particular, the onboard processing of a high-resolution video for the generation of the associated features may be computationally and power demanding placing significant burden on the encoder's hardware, while on the other hand, a large bandwidth is required to transmit full-resolution data at a base station for further processing and classification.

In this thesis we address the above drawbacks by exploiting the framework of *compressive sensing* (CS), which is acting simultaneously as a sensing and compression protocol and is based on non-adaptive linear incoherent projections for the representation and reconstruction of sparse signals [9]. We introduce two CS-based video classification approaches that are directly applied in the compressed domain without any need for signal reconstruction. More specifically, we consider the scenario of a sensing system equipped with a single pixel camera [10] having the ability to estimate the correct class without demanding the acquisition of the video data at full resolution. Instead, suitable feature vectors associated with the captured video sequence, along with the appropriate decision rule, are expressed in terms of the compressed measurements.

The remainder of the text is organised as follows. In Chapter 2 background theory and related work in automatic video classification and compressive sensing are summarized. In Chapter 3 a frame-based compressive video classification system is described that uses directly the compressed measurements of the video frames for the classification task. A block-based compressive video classification system using sparse coding and spatio-temporal pyramid matching is introduced in Chapter 4, while the proposed systems are evaluated in terms of classification accuracy in Chapter 5. Finally, in Chapter 6 we conclude and investigate possible future extensions.

Chapter 2

Background theory and related work

2.1 Automatic video classification

Automatic video classification (AVC) is defined as the task of assigning a meaningful label to video sequences according to a set of predefined labeled classes. As any other classification task, the effectiveness of an AVC system is determined by two main factors, namely, i) the quality of extracted *features* that comprise the *video sequence signature* and ii) the selected *classifier* that is used for the final label assignment to a query video sequence. In this section, commonly used features and classification rules are summarized.

2.1.1 AVC features

Video sequences generally include information from three modalities, that is *text*, *audio* and *visual* and as a result, features extracted can come from any combination of them. However, in the following, only the visual modality is explored, assuming that in the video sequences under consideration no audio channel is available and no textual information exists.

Color-based features: Each video frame is composed of a number of pixels whose color is described by a set of values from a color-space, e.g. RGB, HSV. One of the simplest features that can be extracted from color information is the *color histogram* that is the number of pixels in the frame at each value of the used color space. Thus, a color histogram represents the distribution of colors in a specific frame and is invariant

to translation and rotation of image content. Nevertheless, its simplicity comes not for free, since it exhibits two main drawbacks. Firstly, as a global frame feature it discards spatial information of the distribution and, secondly, it is highly sensitive to illumination changes.

Regarding the first drawback, one approach adopted is to first divide the frame into a dense grid of uniformly distributed pixel regions, compute the histogram of intensities over each cell, and then concatenate all the histograms into one large feature vector. A second solution for taking into account the local color spatial correlation as well as the global distribution of this spatial correlation is the *color correlogram*. In fact, a color correlogram of a frame forms a table of statistics for color value pairs, where the k-th entry for pair (i, j) specifies the probability of finding a pixel of value j from a pixel of value i at a distance k in the frame.

The second drawback can be alleviated by a normalization of each color channel in ℓ-1 or ℓ-2 norm, since during illumination change color values for pixels approximately undergo independent multiplicative changes in each color channel R, G and B.

Motion-based features: Motion in a video sequence can be generally due to both object movement and camera action. The features extracted to describe these patterns depend largely on the calculation of *optical flow*. In optical flow approaches dominating motion patterns are approximated by analysing pixel intensities across consecutive frames.

Under the two assumptions (constraints) of 1) *brightness consistency assumption* (BCA), namely color or intensity values of corresponding pixels in frame t and frame $t+1$ are constant and 2) smoothness of velocity, where motion of pixels in a small region is small and uniform, the optical flow vector field can be computed by discretizing the following equation:

$$E(u, v) = \iint_{\Omega} ((E_x u + E_y v + E_t)^2 + \alpha^2(|\nabla u|^2 + |\nabla v|^2))dxdy \tag{2.1}$$

where, E stands for image intensity, E_x, E_y and E_t are the partial derivatives along x, y and t axis respectively, u and v is the velocity along x and y direction respectively, Ω is a small region of interest and α^2 is a regularization parameter.

The final features used for the modeling of motion patterns are optical flow histograms or optical flow differential histograms computed in a global or local manner. An approach used commonly for the task of action/activity recognition for the description of localized motion is the histogram of optical flow (HOF). First, optical flow is

computed. Then, differentials of optical flow are calculated in order to compensate for camera motion. Finally, the sequence of differential optical flow images is split in overlapping subvolumes that are further divided in small space-time cuboids. Optical flow is accumulated in a 1D-histogram per cuboid and the histograms of all cuboids in the subvolume are concatenated, resulting in a feature vector per subvolume.

Gradient-based features: Intensity gradient is known to provide appearance and shape information in videos, since dominant gradient magnitudes correspond to image edges. A popular approach for the computation of localized gradient features is the histogram of oriented gradients (HOG), where image gradients are used to calculate feature descriptors based on histogram of dominant orientations within dense and overlapping space-time regions. Initially, the gradient is computed by applying appropriate filter kernels on the horizontal and vertical directions of the image (e.g. $[1, 0, -1]$ and $[1, 0, -1]^T$) and next the same philosophy as in calculation of HOF is followed: the gradient images are split in overlapping subvolumes, consisting of space-time cuboids, and 1D-histograms of gradient orientation are computed per cuboid. Finally, the histograms in each subvolume are concatenated to form the final subvolume feature vector.

2.1.2 AVC classifiers

After the signatures are extracted from the video sequences, they are used (with their corresponding labels) as input to a classification algorithm. Thus, a training model is firstly built which is then utilized for the classification of an unlabeled observed sequence.

k-Nearest Neighbors k-Nearest Neighbors (kNN) is one of the simplest machine learning algorithms used for classification. It is of the type of "lazy" learning schemes in the sense that no training model is built before classification. Given a query sequence, the distance between its signature and each of the training signatures is firstly computed. The commonest label among the k closest training signatures is then assigned to the unlabeled sequence. The main advantage of kNN algorithm is its simplicity that facilitates implementation. However, when the training set is large the algorithm faces speed and memory issues while the prediction accuracy can quickly degrade when the dimensionality of the feature vector grows.

Support Vector Machines: Support vector machines (SVM) belong to the category of binary discriminative classifiers, since they focus on separating two or more classes rather than modeling them. After mapping training signatures in a high dimensional space using the *kernel trick* in order to achieve linear separability, they learn the hyperplane that maximizes the geometrical margin between the two classes by solving a convex quadratic programming problem. This hyperplane is expressed through a weighted combination of a small (ideally) number of training samples, named the *support vectors*. A query sequence is assigned a label according to the side of the feature space it resides with reference to the separating hyperplane. Although SVM were initially introduced as a binary classifier, extensions for the multiclass case exist, either by finding one hyperplane for each pair of classes (one-against-one strategy) or by finding one hyperplane that optimally separates each class from the rest (one-against-all strategy).

Hidden Markov Models: Hidden Markov model (HMM) is a probabilistic technique used to model the temporal structure of the features extracted from a video sequence, which can indeed be a discriminative cue for the classification task. The HMM assumes that the system being modeled is a Markov process with a finite number of unobserved (*hidden*) states, following a prior distribution π. In each time instance the system enters one state according to a probability distribution A depending only on the previous state. After the state transition, an observable symbol (a feature vector in AVC framework) is generated based on a probability distribution B depending on the current state. The training set is used to learn the model $\lambda = (A, B, \pi)$ so that the probability $p(O|\lambda)$ is maximized, where O is a sequence of observations. Most times a number of HMMs are learned for each class (each one describing the temporal evolution of one feature type). A query sequence is labeled with the label of the class whose model maximizes the posterior probability $p(\lambda|O)$ (or the sum of posterior probabilities if more than one HMMs/class are trained).

2.2 Compressive sensing

According to the well-known Shannon/Nyquist theorem, which is dominating the procedure of signal acquisition, a signal's sampling rate must be at least twice its maximum bandwidth for loss of information to be avoided. However, as it is proven by the traditional scheme of transform coding, this is a highly redundant sampling procedure for most natural signals, such as images, which follow a sparse model. *Compressing sens-*

ing (CS) addresses this issue by acting as a simultaneous sampling and compressing protocol, enabling the effective sensing of a signal using a relatively small set of linear non-adaptive incoherent measurements [11].

To put it more formally, consider the signal $\boldsymbol{x}$, a $N \times 1$ column vector in $\mathbb{R}^N$, and a $N \times N$ basis matrix $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \cdots, \boldsymbol{\psi}_N]$. Such a signal, that can be represented as $\boldsymbol{x} = \boldsymbol{\Psi s}$, is said to be K-sparse in the basis $\boldsymbol{\Psi}$ if the vector $\boldsymbol{s} \in \mathbb{R}^N$ has only $K \ll N$ non-zero entries.

Under the CS paradigm the signal is directly acquired in a compressed form using the following measurement model,

$$\boldsymbol{y} = \boldsymbol{\Phi x} = \boldsymbol{\Phi \Psi s} \tag{2.2}$$

where $\boldsymbol{\Phi}$ is an $M \times N$ random measurement matrix with $M < N$ and $\boldsymbol{y}$ is the $M \times 1$ resulting vector of compressed measurements. The above sampling procedure is characterized as incoherent if the largest correlation between the rows of the measurement matrix $\boldsymbol{\Phi}$ and the columns of the basis matrix $\boldsymbol{\Psi}$,

$$\mu(\boldsymbol{\Phi}, \boldsymbol{\Psi}) = \max_{k,j} | < \boldsymbol{\phi_k}, \boldsymbol{\psi_j} > | \tag{2.3}$$

is sufficiently small so that the rows of $\boldsymbol{\Phi}$ cannot sparsely represent the columns of $\boldsymbol{\Psi}$ (and vice versa).

Common choices for $\boldsymbol{\Phi}$ are random matrices with independent and identically distributed (i.i.d.) Gaussian or Bernoulli entries, with columns normalized to unit ℓ_2 norm. Such matrices are proven to exhibit a very low coherence with any fixed basis $\boldsymbol{\Psi}$, building up the universality of the sampling scheme.

2.2.0.1 Single-pixel camera architecture

The *single-pixel camera* (SPC) is one of the hardware applications proposed for the direct application of CS in the optical domain. As it is revealed by its name this imaging device has a single photon detector in contrast to a conventional camera which incorporates a vast array of photon detectors, one for each pixel.

As shown in Fig. 2.1 the light-field is focused by Lens 1 onto a digital micromirror device (DMD), a type of reflective spatial light modulator (SLM) that selectively redirects parts of the light beam. The DMD consists of an array of micro-mirrors each of which is suspended above an individual static random access memory (SRAM) cell. Any mirror rotates about a hinge and can be positioned in one of two states (± 10

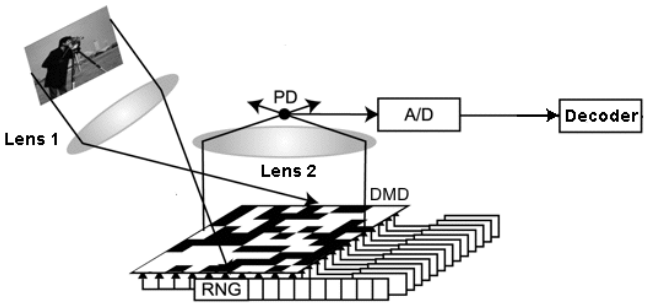


Figure 2.1: Single-pixel camera architecture

degrees from horizontal) according to which bit is loaded into the SRAM cell. As a result, light falling on the DMD can be reflected in two directions depending on the orientation of the mirrors.

Assume a measurement matrix $\Phi = [\boldsymbol{\phi_1} \cdots \boldsymbol{\phi_j} \cdots \boldsymbol{\phi_M}]^T \in \mathbb{R}^{M \times N}$ and an image $\boldsymbol{x} \in \mathbb{R}^N$. During the acquisition of j-th compressive measurement, each element of the SLM corresponds to a particular element of $\boldsymbol{\phi_j}$. The corresponding element of the SLM can be oriented either towards (for 1 in SRAM cell) or away from (for 0 in SRAM cell) Lens 2. Lens 2 collects the reflected light and focuses it onto a single photon detector that integrates the product of $\boldsymbol{x}$ and $\boldsymbol{\phi_j}$ to compute the measurement $y_j = \boldsymbol{\phi_j}^T \boldsymbol{x}$ as its output voltage. This voltage is then digitized by an A/D converter and sent to the decoder for further processing. It is noted that since the DMD is programmable, arbitrary measurement matrices can be applied, for instance by dithering the mirrors back and forth during the photon detector integration time.

Two issues arise in the above setting. Firstly, since the acquisition of compressed samples is realized in a sequential manner, the scene captured is assumed to be static or at least slowly changing. When it comes to a video sequence, it is assumed that the scene remains unchanged during the sampling of each frame. Secondly, storing large measurement matrices in a SPC system is impractical, thus instead of a purely random Φ a more appropriate choice is a *structurally random matrix* [12], such as Block Walsh-Hadamard operator (BWHT) that admits a fast transform-based and memory efficient implementation.

2.2.0.2 CS signal reconstruction

Under the assumptions of signal sparsity and incoherent sampling and if additionally $M = O(K \log(N/K))$ it is possible with high probability to reconstruct the initial signal

by solving the following ℓ_1-norm minimization problem

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s}}{\operatorname{argmin}} \|\boldsymbol{s}\|_1, s.t. \;\; \boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{\Psi}\boldsymbol{s} \tag{2.4}$$

When noise is present a perfect reconstruction may not be possible, so a relaxed version of the above formulation is considered:

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s}}{\operatorname{argmin}} \|\boldsymbol{s}\|_1, s.t. \;\; \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\Psi}\boldsymbol{s}\|_2 \leq \epsilon \tag{2.5}$$

where ϵ bounds the signal noise.

Several algorithms have been proposed to solve the optimization problems of (2.4) and (2.5) after appropriate reformulations. These include linear programming methods ([9], [13]) as well as greedy methods from the class of matching pursuit algorithms ([14], [15]).

2.2.0.3 Classification in the compressed domain

Precise signal reconstruction is not always the goal in a signal acquisition application. Since the meaningful information of the signal is preserved through CS measurement process the compressed samples acquired can be directly exploited for the task of signal classification/detection as well. The far less aggressive goal of classification compared to reconstruction is proven to be accomplished with much fewer compressive samples ([16], [17]).

It is noted, that under a machine learning framework the CS measurement procedure can be seen as a non-adaptive dimensionality reduction technique, known as *random projections* (RPs). RPs are known to generate a low-dimensional representation of the initial signal that encodes its salient information content, in a simple linear and data-independent universal way.

2.2.1 Sparse coding and dictionary learning

According to the *linear generative model* a signal $\boldsymbol{x} \in \mathbb{R}^M$ can be expressed as a linear combination of several basis items coming from a dictionary $\mathbf{D} \in \mathbb{R}^{\mathbf{M}\times\mathbf{N}}$, that is

$$x = \boldsymbol{D}a \tag{2.6}$$

where $a \in \mathbb{R}^N$ is the representation vector.

In case where an orthonormal basis set, like DCT or wavelets, constitutes $\boldsymbol{D}$ it is

straight-forward to compute a by solving the above system of linear equations. However, if $\mathbf{D}$ is overcomplete, that is, $M < N$, the system above is underdetermined and further assumptions on the signal properties have to be made to solve (2.6).

Sparse coding tackles this problem by representing the signal under consideration as a weighted linear combination of only a small number of items coming from $\mathbf{D}$. Sparsity assumption has been validated for several natural signals, like images, and has been successfully applied in a number of computer vision related tasks, such as image denoising, impainting and classification.

Formally, the sparse coding problem can be formulated as the following ℓ_1 minimization problem:

$$\hat{\boldsymbol{a}} = \underset{\boldsymbol{a}}{\operatorname{argmin}} \|\boldsymbol{a}\|_1, s.t. \;\; \boldsymbol{x} = \mathbf{D}\boldsymbol{a} \tag{2.7}$$

If signal noise exists the problem is reformulated as follows:

$$\hat{\boldsymbol{a}} = \underset{\boldsymbol{a}}{\operatorname{argmin}} \|\boldsymbol{a}\|_1, s.t. \;\; \|\boldsymbol{x} - \mathbf{D}\boldsymbol{a}\|_{\mathbf{2}} \leq \epsilon \tag{2.8}$$

As it can be seen, equation (2.8) describes exactly the same problem as equation (2.5) and as a result the same algorithms can be applied.

A second issue that arises in the framework of sparse coding is the nature of dictionary $\mathbf{D}$. Although preconstructed fixed dictionaries are always a choice, they are typically limited to their ability to sparsify the signals they are designed to handle. In order to overcome this limitation the dictionary $\mathbf{D}$ can be built through a learning process using a training database of signal instances.

Concretely, the problem of sparse dictionary learning can be formulated as follows:

$$\min_{\mathbf{D}, \{\boldsymbol{a_i}\}_{\mathbf{i=1}}^{\mathbf{M}}} \frac{1}{M} \sum_{i=1}^{M} \frac{1}{2} \|\boldsymbol{x}_i - \mathbf{D}\boldsymbol{a_i}\|_{\mathbf{2}}^{\mathbf{2}} + \lambda \|\boldsymbol{a_i}\|_{\mathbf{1}} \tag{2.9}$$

where λ is a regularization parameter.

The above problem is normally solved through an alternating iterative process. At each stage of one iteration, firstly the sparse codes $\boldsymbol{a}_i$ are updated as in (2.8), while $\mathbf{D}$ is kept constant. Next, $\mathbf{D}$ is updated by keeping $\boldsymbol{a}_i$ fixed.

Common learning algorithms process the training database in batch-mode ([18], [19], [20]), accessing the whole dataset at each iteration. Although effective in case of moderate-sized training sets, these algorithms are inappropriate for larger databases, a typical scenario in computer vision tasks (e.g. learning a dictionary on millions of small image patches). This issue is addressed by a recent online dictionary learning

approach proposed in [21]. The algorithm, which is processing the training samples one at a time (a mini-batch extension also exists), is based on stochastic approximation for the dictionary update step and is outlined below.

Algorithm 1 Online dictionary learning [21]

1: $\mathbf{A_0} \leftarrow \mathbf{0}$, $\mathbf{B_0} \leftarrow \mathbf{0}$
2: **for** $t = 1$ to M **do**
3: Draw $\boldsymbol{x}_t$
4: Sparse coding:

$$\boldsymbol{a}_t = \underset{\boldsymbol{a}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{x}_t - \mathbf{D_{t-1}}\boldsymbol{a}\|_{\mathbf{2}}^{\mathbf{2}} + \lambda\|\boldsymbol{a}\|_{\mathbf{1}}$$

5: $\mathbf{A_t} \leftarrow \mathbf{A_{t-1}} + \boldsymbol{a_t}\boldsymbol{a_t^T}$
6: $\mathbf{B_t} \leftarrow \mathbf{B_{t-1}} + \boldsymbol{x_t}\boldsymbol{a_t^T}$
7: Dictionary update:
repeat
for $j = 1$ to N **do**

$$\boldsymbol{u}_j \leftarrow \frac{1}{\mathbf{A_{jj}}}(\boldsymbol{b}_j - \mathbf{D}\boldsymbol{a_j}) + \boldsymbol{d_j}$$
$$\boldsymbol{d}_j \leftarrow \frac{1}{\max(\|\boldsymbol{u}_j\|_2, 1)}\boldsymbol{u}_j$$

end for
until convergence
8: **end for**
9: **return** $\mathbf{D_M}$

2.3 Related work

A lot of research effort has been put on the application of CS theory in the field of image and video processing, both for the tasks of reconstruction and classification/detection. Based on our assumption for a system equipped with single-pixel cameras only, in the following, we focus on recent works concerning on the use of the CS theory for video classification.

In [22] a compressive classification framework that operates directly on the compressive measurements without a reconstruction step is proposed. Building on the fact that, under the condition that sufficient number of measurements is acquired, the random measurement procedure preserves the essential structure of a smooth manifold

and by extending the generalized maximum likelihood classification (GMLC) notion to the compressed domain, the authors propose the *smashed filter classifier* a compressive version of the traditional matched filter detector. Evaluation of the scheme was realized under a controlled experiment of target classification, using a dataset of images depicting rotated around vertical axis versions of three vehicles. The single-scaled smashed filter was later extended to a multiscale version in [23], where random samples of regularized versions of each image were taken in various scales, in order to obtain differentiable image appearance manifolds at this set of scales.

Classification of texture images under unknown viewpoint and illumination using compressive sensing is presented in [24]. Each texture image is initially split in a number of non-overlapping patches, named *textons* which are projected to a low-dimensional space through the CS measurement model. The compressed textons extracted from the training set are used to learn a codebook of K words with k-means clustering and each texture image is assigned a signature based on a bag-of-words model: the compressed textons are quantized using the learned codebook and a histogram of word relative frequencies is computed for each image, forming its signature. A signature is formed likewise for a query image which is finally classified using a nearest neighbor rule in the χ^2 distance meaning.

In [25] the use of classical dimensionality reduction embeddings in the compressed domain is explored. The authors assume a network of vision sensors that capture multiple compressed measurements of visual signals. The random measurements are then projected onto a subspace defined by the embedding of a linear classification subspace (learned on the training set through, PCA or LDA) in the compressed domain. The new projections form the feature vectors used for the classification task performed by a NN classifier.

Video sequences have been successfully modeled as linear dynamical systems (LDS). In [26] the CS and LDS frameworks are unified by proposing a novel compressive measurement strategy named CSLDS. Only the low-dimensional dynamic parameters forming the state sequence are measured and after measurements are accumulated over time at the decoder the static parameters (the observation matrix) are estimated. Among other applications including dynamical texture reconstruction and hyper-spectral imaging, the CSLDS framework is also applied to video classification of traffic density and human activity sequences. The classification is performed directly on the estimated LDS parameters, without any reconstruction of initial frames, using a NN classifier and the Procrustes distance as distance metric.

Human activity video classification results are further improved by describing hu-

man activities as a non-linear dynamical system in [27]. Distance matrices of differences between successive frames are computed and interpreted as intensity texture images. Thus, the problem of activity classification is translated to at texture recognition problem and the extraction of features is accomplished by a texture classification method named *local binary patterns* (LBP). A simple NN classifier is again used as the decision rule.

Discriminative nature of sparse representation is commonly exploited for the classification task in computer vision problems. In the seminal work of [28] the face recognition problem is reformulated as a sparse vector reconstruction problem. More specifically, features extracted from a training set of face images of several subjects under various illumination scenarios form a training dictionary. During runtime phase the feature of a query image is expressed as a sparse linear combination of the training dictionary entries by solving an ℓ-1 minimization problem. Ideally the large non-zero coefficients of the recovered sparse vector should correspond to dictionary entries belonging to a specific class, which is the one assigned to the query image.

Sparse representations are used in the problem of patch-based texture classification via texton learning in [29]. Initially, a sparse coding formulation is used to learn one dictionary for each texture class and an overall texton dictionary $\mathbf{D}$ is formed by uniting all class-specific dictionaries. For computational efficiency sparse codes are evaluated over a small subset of elements of $\mathbf{D}$, the closest to the patch to be encoded and an image signature is formed by summing sparse representations of all patches belonging to the same image. Classification is performed by an χ^2-NN classifier.

An image categorization approach via sparse coding is explored in [30], based on SIFT sparse codes and a linear spatial pyramid matching (SPM) kernel. SIFT appearance descriptors computed are sparsely coded on a learned dictionary, that is calculated using the feature-sign search algorithm of [20]. The final image representation is computed by max pooling of the SIFT sparse codes across different locations and over different spatial scales, following a spatial pyramid structure. Spatial pyramid is preferred to a simple bag-of-features (BoF) approach since the first can better model the spatial layout of descriptors in the image. The classification is accomplished by a training an SVM with a simple linear SPM kernel instead of the standard non-linear kernels, like intersection or χ^2 kernel, used in the majority of state-of-the-art approaches, thus enabling better scalabitily of the proposed system.

The authors of [31] propose an algorithm for online unusual event detection in videos. Based on the intuition that usual events are more likely to be reconstructible from an event dictionary while unusual are not, sparse reconstruction codes and dic-

tionary are inferred jointly, by minimizing an objective function measuring the abnormality of events. The algorithm is completely unsupervised and dictionary is updated in an online fashion, in order to cope with concept drift. Events having an abnormality value higher than a predefined user threshold are considered unusual.

Chapter 3

Frame-based compressive video classification

Conventional approaches for video classification build on the availability of full resolution video data in order to extract meaningful features and form a suitable sequence signature. However, under a typical resource-constraint sensing system scenario, such systems are highly inefficient since strict power, processing, memory and bandwidth limitations exist. In this chapter we introduce a simple compressive video classification (CVC) architecture that works directly in the compressed domain. We consider the futuristic scenario of a sensing system whose encoder is equipped with a *single pixel camera*, as it is described in Section 2.2.0.1, that is able to estimate the correct class without demanding the acquisition of video data in full resolution. Equally importantly no reconstruction of the uncompressed video sequence is necessary, reducing both the processing burden at the decoder and the required sampling rate and thus saving transmission bandwidth.

3.1 CS-based video acquisition model

Let $\mathbf{V} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_R\}$ be a video sequence consisting of R frames $\boldsymbol{x}_j$, $j = 1, \ldots, R$, where each frame is represented by its luminance component. For convenience, in the following we consider that each frame is expressed as a column vector, $\boldsymbol{x}_j \in \mathbb{R}^N$. Then, a vector of compressed measurements $\boldsymbol{g}_j$, $j = 1, \ldots, R$ is generated for each frame using a suitable measurement matrix $\boldsymbol{\Phi}$ (for simplicity we use the same matrix for each frame) as follows,

$$\boldsymbol{g}_j = \boldsymbol{\Phi}\boldsymbol{x}_j \ , \tag{3.1}$$

where $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ is a random measurement matrix with $M < N$.

Common choices for $\mathbf{\Phi}$ are random matrices with independent and identically distributed (i.i.d.) Gaussian or Bernoulli entries, whose columns are normalized to unit ℓ_2-norm. In a decision system with limited resources, some additional requirements should be posed on the choice of the desired matrix $\mathbf{\Phi}$, such as the use of minimal number of compressed measurements, and the fast and memory efficient computation along with a "hardware-friendly" implementation. A class of matrices satisfying these requirements, the so-called *structurally random matrices*, was introduced recently [12]. The block Walsh-Hadamard (BWHT) operator is a typical member of this family and is used subsequently.

Notice that with the use of a single-pixel camera the generation of CS measurements does not require the acquisition of frames at full resolution, thus reducing significantly the processing and storage expenses of the sensing device.

In the framework of compressive video classification (CVC), we consider that the given video sequence belongs to the class c, where $c \in \{1, \ldots, C\}$. Following a supervised learning approach, a set of training video samples is obtained for each class, $\mathcal{T}_c = \{\mathbf{V}_1^c, \ldots, \mathbf{V}_Q^c\}$. For simplicity, we consider that the number of training samples Q is equal for all the classes. Let also $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_C\}$ denote the overall set of training samples. The CVC problem is stated as follows: *Given a low-dimensional signature of the acquired video sequence, a training dictionary* $\mathbf{D}$*, and a measurement matrix* $\mathbf{\Phi}$*, estimate the correct class* $c \in \{1, \ldots, C\}$.

3.2 Proposed CVC system

A typical classification system consists of two main phases, namely, a *feature extraction phase*, where a more compact representation of the original information is generated in a low-dimensional space, with the goal of preserving a high discriminative power, and a *classification phase*, where the extracted feature vector of the given signal is compared with the corresponding features of the training samples by means of a suitable similarity criterion resulting in the estimated class. In the following sections, the main characteristics of the two phases are introduced in detail for the proposed CVC system, which is depicted in Fig. 3.1

3.2.1 Feature extraction

During the feature extraction phase, a set of CS measurements is generated for the frames of each video sequence. More specifically, let us denote by $\boldsymbol{x}_{j,q}^c \in \mathbb{R}^N$ the j-

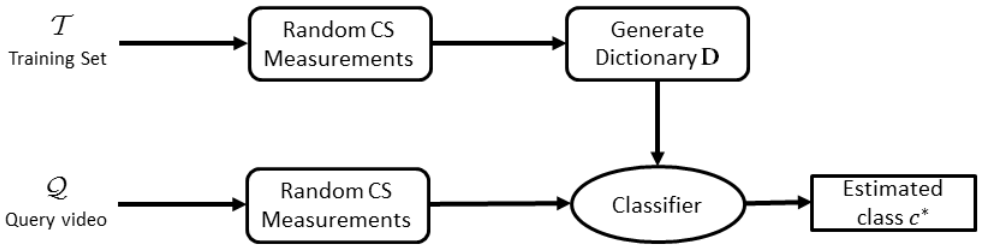


Figure 3.1: Proposed frame-based CVC architecture.

th frame of the q-th sequence belonging to class c. Then a low-dimensional (feature) measurement vector $\boldsymbol{g}_{j,q}^c \in \mathbb{R}^M$ is assigned to $\boldsymbol{x}_{j,q}^c$ as given in (3.1). The overall signature of the q-th sequence in class c is given by,

$$\mathbf{V}_q^c \longmapsto \mathcal{F}_q^c = \{\boldsymbol{g}_{1,q}^c, \ldots, \boldsymbol{g}_{R,q}^c\}, \tag{3.2}$$

and by augmenting all the training signatures we get the overall signature for the training set, $\mathcal{F} = \bigcup_{\forall q,c} \mathcal{F}_q^c$ and, finally, the following $M \times RQC$ overall training dictionary $\mathbf{D}$,

$$\mathbf{D} = [\boldsymbol{g}_{1,1}^1, \ldots, \boldsymbol{g}_{j,q}^c, \ldots, \boldsymbol{g}_{R,Q}^C]. \tag{3.3}$$

3.2.2 Classification phase

Following the feature extraction step, the classification phase is performed by means of an appropriate decision rule. In the following, three decision criteria are employed: the first two exploit directly the CS measurements forming the signature of the sequence, while the third one is based on the solution of a convex optimization problem for the recovery of a sparse class-indicator vector. It is noted that in the proposed scheme each projected frame is classified separately and the estimated class is the one with the highest frequency of appearance among the separately classified projected frames of each sequence.

Nearest-Neighbor

The simplest decision rule for estimating the optimal class of compressed video frame

g_{query} is given by the *nearest-neighbor* (NN) criterion defined by

$$c^* = \underset{c\in\{1,\ldots,C\},\,\forall\, j,q,c}{\operatorname{argmin}} \|\boldsymbol{g}_{query} - \boldsymbol{g}_{j,q}^{c}\|_2^2 \,. \tag{3.4}$$

In other words, the query measurement vector is assigned the class label of the closest training measurement vector in a euclidean distance sense.

Support Vector Machine

Support vector machine (SVM) is a discriminative classifier, originally designed for binary classification. It determines the hyperplane that linearly separates the training data with the maximum possible margin. Given input training vectors $\boldsymbol{g}_i \in \mathbb{R}^M,\ i = 1,\ldots,L$ and their associated labels $y_i \in \{-1, 1\}$ the optimal hyperplane is found by solving the following quadratic maximization problem:

$$\begin{aligned} &\max_{\boldsymbol{a}} \sum_{i=1}^{L} a_i - \frac{1}{2}\sum_{i,1=1}^{L} y_i a_i y_j a_j K(\boldsymbol{g}_i, \boldsymbol{g}_j) \\ &\text{s.t.} \ \sum_i a_i y_i = 0, \ \ 0 \le a_i \le C,\ i = 1,\ldots,L \end{aligned} \tag{3.5}$$

where $K(\boldsymbol{g}_i, \boldsymbol{g}_j) = \phi(\boldsymbol{g}_i)^T\phi(\boldsymbol{g}_j)$ an appropriate kernel function, with $\phi(\boldsymbol{g}_i)$ a function that maps $\boldsymbol{g}_i$ into a high dimensional space. The training samples associated to the non-zero Lagrange multipliers $a_i,\ i = 1,\ldots,L$ are the *support vectors* and are the ones falling on the margin.

A query video frame $\boldsymbol{g}_{query}$ is classified according to the discriminant function given by

$$d(\boldsymbol{g}_{query}) = w^T\phi(\boldsymbol{g}_{query}) + b = \sum_{i=1}^{L} a_i y_i K(\boldsymbol{g}_i, \boldsymbol{g}_{query}) + b. \tag{3.6}$$

Various kernel functions $K(\boldsymbol{g}_i, \boldsymbol{g}_j)$ have been used in order to estimate high dimensional inner products. In our case, a simple linear kernel, defined as $K(\boldsymbol{g}_i, \boldsymbol{g}_j) = \boldsymbol{g}_i^T\boldsymbol{g}_j$, is chosen. Additionally, the multi-class SVM version is used. More specifically, let $\mathcal{D} = \{\boldsymbol{g}_{j,q}^{c}\}, j = 1,\ldots,R,\ q = 1,\ldots,Q,\ c = 1,\ldots,C$ denote the labeled training data. A way to solve the problem of multi-class classification is to follow a one-against-one approach, where an SVM is constructed for every pair of classes by training it to discriminate them. The number of SVMs to be trained in this approach is equal to $C(C-1)/2$. Let also $(k,\, l)$ be a pair of classes and $d_{k,l}(\cdot)$ the associated discriminant

function (cf. (3.6)). Then, given the query feature vector $\boldsymbol{g}_{query}$, if $d_{k,l}(\boldsymbol{g}_{query}) > 0$ a vote is assigned to the k-th class, otherwise the vote is given to the l-th class. The process is repeated for each pair of classes and finally, the class with the *maximum number of votes* is assigned to the query $\boldsymbol{g}_{query}$.

Sparse Representation Classification

Regarding the later category of classification methods, an alternative way to estimate the class of the query video frame is obtained by reformulating the classification problem as a problem of recovering an appropriate sparse vector. More specifically, a class-indicator vector $\boldsymbol{\alpha}$ is introduced, where

$$\boldsymbol{\alpha} = [\alpha_1^1, \ldots, \alpha_Q^1, \ldots, \alpha_1^i, \ldots, \alpha_Q^i, \ldots, \alpha_1^C, \ldots, \alpha_Q^C] \in \mathbb{R}^{CQ} \ .$$

If the query video frame belongs to the i-th class, in the ideal case we expect that its measurement vector $\boldsymbol{g}_{query}$ will be similar to the corresponding training data of the i-th class, or equivalently to the corresponding columns of the training dictionary $\mathbf{D}$ (cf. (3.3)). Accordingly, the class-indicator vector has the following Q-sparse structure

$$\boldsymbol{\alpha} = [0, \ldots, 0, \alpha_1^i, \ldots, \alpha_Q^i, 0, \ldots, 0] \ , \tag{3.7}$$

with the non-zero components corresponding to the Q indices of the i-th class. Thus, the CVC problem is reduced to a problem of recovering the sparse support of $\boldsymbol{\alpha}$, which is expressed as the solution of a convex optimization problem as follows,

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha} \in \mathbb{R}^{CQ}}{\operatorname{argmin}} \|\boldsymbol{\alpha}\|_1 \ , \ \text{s.t.} \|\boldsymbol{g}_{query} - \mathbf{D}\boldsymbol{\alpha}\|_2 < \epsilon \ . \tag{3.8}$$

Numerous algorithms have been proposed to solve the optimization problems of (3.8) after appropriate reformulations. These include linear programming methods ([9], [13]) as well as greedy methods from the class of matching pursuit algorithms ([14], [15]). Motivated by its simple and fast implementation, the orthogonal matching pursuit (OMP) algorithm is used in our case.

Notice that in practice, especially in noisy conditions, the recovered class-indicator vector is not exactly Q-sparse. In this case, an additional step is applied to obtain the final class estimate by enforcing the Q-sparsity of $\boldsymbol{\alpha}^*$ as follows,

$$c^* = \underset{c=1,\ldots,C}{\operatorname{argmin}} \|\boldsymbol{g}_{query} - \mathbf{D}\delta_c(\boldsymbol{\alpha}^*)\|_2 \ , \tag{3.9}$$

where $\delta_c(\boldsymbol{\alpha})$ denotes the block-Kronecker operator, which sets to zero all the components of $\boldsymbol{\alpha}$ except for these corresponding to the Q indices of the c-th class.

Chapter 4

Block-based compressive video classification using dictionary learning and spatio-temporal pyramid matching

In this chapter a more sophisticated compressive video classification system is introduced that takes into consideration the limitations of a resource-constraint sensing system scenario and is directly applied in the compressed domain. Under the assumption of a video classification system equipped with a single-pixel camera, each video frame is acquired compressively in a block-basis. An appropriate signature is computed through sparse coding of compressively sampled blocks on a trained dictionary, in conjunction with a pooling function applied in spatio-temporal video cubes. The system is able to estimate the correct class demanding neither the acquisition of full resolution video data nor the reconstruction of the uncompressed video sequences, reducing both the processing burden at the decoder and the required sampling rate and thus saving transmission bandwidth.

4.1 Block-based CS video acquisition model

Block-based image/video processing is a commonly adopted philosophy alleviating large computational and memory burdens imposed on an imaging system in case of processing the image/frame in its entirety. Under the single-pixel camera scenario, sampling in blocks enables substantial reduction to the memory requirements for storing the random

measurement operator.

A key parameter in block-based classification tasks is the size of the block. Blocks of too small size cannot capture the large-scale structures that may be dominant features of video appearance and are highly sensitive to noise, while blocks of large dimensionality lead to the presence of irrelevant and noisy features that can deteriorate the performance of the classifier.

Let $\mathbf{V} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_R\}$ be a video sequence consisting of R frames, where each frame is represented by its luminance component. Each frame is divided into B non-overlapping blocks of size $\sqrt{N_B} \times \sqrt{N_B}$ pixels and acquired using an appropriately sized measurement matrix. Suppose that $\boldsymbol{x}_j^i \in \mathbb{R}^{N_B}$ is a vector representing, in raster-scan fashion, the i-th block of j-th input frame. A vector of random measurements $\boldsymbol{g}_j^i \in \mathbb{R}^{M_B}$ is then generated for each block $\boldsymbol{x}_j^i$ using an appropriate measurement matrix $\boldsymbol{\Phi_B}$ (the same for each block for simplicity reasons) according to the following *block compressive sensing* (BCS) model,

$$\boldsymbol{g}_j^i = \boldsymbol{\Phi_B} \boldsymbol{x}_j^i. \tag{4.1}$$

where $\boldsymbol{\Phi_B}$ is a $M_B \times N_B$ random measurement matrix with $M_B < N_B$. It is straightforward to see that (4.1) applied block-by-block to a frame is equivalent to a block-diagonal total frame measurement matrix $\boldsymbol{\Phi}$,

$$\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\Phi_B} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Phi_B} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{\Phi_B} \end{bmatrix} \tag{4.2}$$

resulting in a total of $M = BM_B$ measurements per frame and a total sampling rate of $\frac{M}{N} = \frac{BM_B}{BN_B} = \frac{M_B}{N_B}$. We note that a single-pixel camera system can easily accommodate BCS acquisition by simply driving the micro-mirror array with the block-diagonal matrix Φ.

4.2 Proposed CVC system

Typically, a classification system is implemented in two distinct phases, namely the *training phase* where an appropriate signature is computed for each video sequence of the training set with the goal of preserving high discriminative power and a *run-time/classification phase* where the extracted feature vector of a query sequence is compared to the corresponding features extracted from the training set by means of an

appropriate similarity criterion resulting in the estimated class. In the following, the aforementioned phases for the proposed CVC system are described in detail.

4.2.1 Feature extraction phase

Feature extraction phase consists of three distinct steps: i) acquisition of random measurements of training sequences according to the BCS sampling model, ii) dictionary learning and sparse coding (SC) of the compressively sampled blocks and iii) video signature computation by pooling block sparse codes across different spatio-temporal locations and over different scales, following a spatio-temporal pyramid video decomposition.

4.2.1.1 Random BCS measurements

During the training phase a set of CS measurements is generated for each video sequence following the BCS acquisition model described in the previous section. More specifically, let us denote by $\boldsymbol{x}_{j,q}^{c} \in \mathbb{R}^{N_B}$ the rasterized j-th block with size $\sqrt{N_B} \times \sqrt{N_B}$ pixels of the q-th sequence belonging to class c. Then a low-dimensional vector $\boldsymbol{g}_{j,q}^{c} \in \mathbb{R}^{M_B \times 1}$ is assigned to $\boldsymbol{x}_{j,q}^{c}$ as given by (4.1). As a result the q-th sequence of class c is mapped to a set of measurement vectors as follows

$$\mathbf{V}_{\mathbf{q}}^{\mathbf{c}} \longmapsto \mathcal{S}_q^c = \{\boldsymbol{g}_{1,q}^{c}, \ldots, \boldsymbol{g}_{BR,q}^{c}\} \tag{4.3}$$

By augmenting the measurement vectors of all training sequences, we get the following overall set of compressed frame blocks

$$\mathcal{S} = \{\mathcal{S}_q^c\}_{q=1,\ldots,Q}^{c=1,\ldots,C}. \tag{4.4}$$

4.2.1.2 Block-based dictionary learning and sparse coding

Research in image statistics clearly reveals that image/video blocks (also found in bibliography as *patches*) are sparse signals. Apart from that, sparsity in general allows the representation to be specialized and capture salient properties of image/video blocks, in others words be of discriminative nature. During the second step of the training phase a sparse vector is assigned to each compressed training block by solving a joint dictionary leaning and sparse coding problem.

Let us denote by $\boldsymbol{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_L]$ a matrix with all the L elements of $\mathcal{S}$ as its

columns. Then the problem of sparse coding is formulated as follows:

$$\min_{\mathbf{D},\{\boldsymbol{a_i}\}_{\mathbf{i=1}}^{\mathbf{L}}} \frac{1}{L}\sum_{i=1}^{L}\frac{1}{2}\|\boldsymbol{s}_i - \mathbf{D}\boldsymbol{a_i}\|_{\mathbf{2}}^{\mathbf{2}} + \lambda\|\boldsymbol{a_i}\|_{\mathbf{1}} \tag{4.5}$$

where $\mathbf{D}$ is an overcomplete dictionary of size $M_B \times K$ (with $M_B < K$), λ is a regularization parameter and $\boldsymbol{a}_i \in \mathbb{R}^K$, $i = 1, \ldots, L$ the assigned sparse codes.

The above problem is normally solved through an alternating iterative process. At each stage of one iteration, firstly the sparse codes $\boldsymbol{a}_i$ are updated by solving an ℓ-1 minimization problem, while $\mathbf{D}$ is kept constant. Next, $\mathbf{D}$ is updated by keeping $\boldsymbol{a}_i$ fixed.

Common learning algorithms process the training database in batch-mode, accessing the whole dataset at each iteration. Although effective in case of moderate-sized training sets, these algorithms are inappropriate for larger databases, a typical scenario in the case of a large number of compressed patches. This issue is addressed by a recent online dictionary learning approach proposed in [21], which is processing the training samples one at a time and uses a stochastic approximation for the dictionary update step.

Since a sparse code is retained for each compressed block, at the end of this step each video sequence is assigned a set of sparse codes $\mathbf{A_q^c} = [\boldsymbol{a}_{\mathbf{1,q}}^{\mathbf{c}}, \ldots, \boldsymbol{a}_{\mathbf{BR,q}}^{\mathbf{c}}]$.

4.2.1.3 Spatio-temporal pyramid representation

For each training video sequence described by a set of compressed blocks, it is possible to compute a single feature vector based on some statistics of the sparse codes computed in the previous step. This is commonly accomplished by the use of an appropriate *pooling function*. Pooling is used to achieve invariance to image transformations, more compact representations and better robustness to noise.

Assuming a set of descriptors $\mathbf{U} = [\boldsymbol{u}_{\mathbf{1}}, \ldots, \boldsymbol{u}_{\mathbf{M}}]$, common choices for a pooling function $\boldsymbol{z} = \mathcal{F}(\mathbf{U})$ are:

- *average* pooling

$$\boldsymbol{z} = \frac{1}{M}\sum_{i=1}^{M}\boldsymbol{u}_i \tag{4.6}$$

- *max* absolute pooling

$$z_j = max\{|u_{1,j}|, \ldots, |u_{M,j}|\}, \tag{4.7}$$

where z_j is the j-th element of $\boldsymbol{z}$ and $u_{i,j}$ is the j-th element of the i-th descriptor.

In our case, since the descriptors to be pooled are sparse, the max pooling function is preferred. This is well established by biophysical evidence in the visual cortex [32] and is verified both theoretically [33] and empirically [30].

Pooling can be accomplished according to the simple bag-of-features (BoF) approach where all local descriptors are pooled in an unordered fashion. However, in this way the spatio-temporal order of the descriptors is discarded, which limits the discriminative power of the representation. In order to overcome this problem a spatio-temporal pyramid representation is used in the following. Informally, the sequence's spatio-temporal layout of compressed blocks is kept by applying a pooling function in various partitions of different levels of the video sequence and concatenating the results in a single vector.

More specifically, assume the q-th video sequence of class c partitioned into sub-volumes in spatial and temporal space over L different levels. At each level l ($l \in [0, \ldots, L-1]$) there exist 2^l subvolumes in each dimension. Block sparse codes of each subvolume $(x, y, t),\ 0 \leq x, y, t \leq 2^l - 1$ are pooled in a vector $\boldsymbol{z}_q^{c,l}(x, y, t)$ and the results across all spatio-temporal locations and over all scales are concatenated into a single vector $\boldsymbol{z}_q^c$ that is normalized, forming the final video sequence signature. The procedure is illustrated in Fig. 4.1.

4.2.2 Classification phase

A query sequence $\mathbf{V}_{query}$ that is given as input to the proposed CVC system, due to the assumption that it is acquired directly by a single-pixel camera following the BCS acquisition model, is described initially by a set of BR random block-based compressive measurements, where B is the number of blocks per frame and R the number of frames per sequence, according to the following mapping

$$\mathbf{V}_{query} \longmapsto \mathcal{S}_{query} = \{\boldsymbol{g}_{1,query}, \ldots, \boldsymbol{g}_{BR,query}\} \tag{4.8}$$

Then, the sparse code of each compressed block $\boldsymbol{g}_{i,query}$ in $\mathcal{S}_{query}$ is computed on

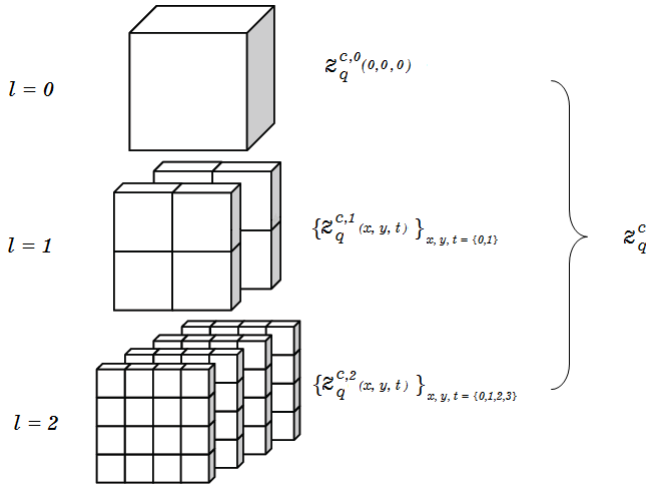


Figure 4.1: Spatio-temporal Pyramid Representation for $L = 3$ levels

the trained dictionary $\mathbf{D}$ by solving the following ℓ-1 minimization problem

$$\boldsymbol{a}_{query} = \underset{\boldsymbol{a}}{\operatorname{argmin}} \|\boldsymbol{a}\|_1 \, s.t. \, \|\boldsymbol{g}_{i,query} - \mathbf{D}\boldsymbol{a}\| \leq \epsilon \tag{4.9}$$

resulting in a set of sparse codes $\mathbf{A}_{query} = [\boldsymbol{a}_{1,query}, \ldots, \boldsymbol{a}_{BR,query}]$ for the query video.

Then, the sparse codes are pooled according to the spatio-temporal pyramid representation described in the previous section and the classification of the query sequence is done by means of an appropriate decision rule. In our case a support vector machine (SVM) classifier is used to discriminate between the classes of the classification problem.

Assuming that the training set is described by $\{(\boldsymbol{z}_i, y_i)\}_{i=1}^{L}$, where $\boldsymbol{z}_i$, $i = 1, \ldots, L$ are the training signatures and y_i the corresponding labels, for a binary classification problem (where $y_i \in \{-1, +1\}$), an SVM aims to learn the optimal hyperplane separating the data of the two classes by solving the following maximization problem

$$\begin{aligned} &\max_{a} \sum_{i=1}^{L} a_i - \frac{1}{2} \sum_{i,1=1}^{L} y_i a_i y_j a_j K(\boldsymbol{z}_i, \boldsymbol{z}_j) \\ &\text{s.t.} \quad \sum_{i} a_i y_i = 0, \quad 0 \leq a_i \leq C, \; i = 1, \ldots, L \end{aligned} \tag{4.10}$$

The kernel $K(\boldsymbol{z}_i, \boldsymbol{z}_j)$ can be in general any valid Mercer kernel function. Here, we

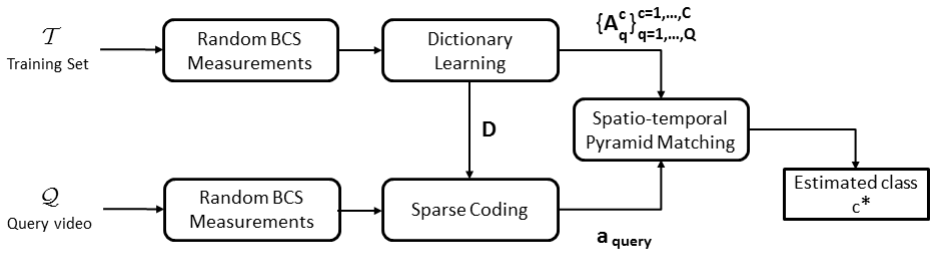


Figure 4.2: Proposed block-based CVC.

use a simple linear spatio-temporal pyramid matching kernel

$$K(\boldsymbol{z}_i, \boldsymbol{z}_j) = \boldsymbol{z}_j^T \boldsymbol{z}_i = \sum_{l=0}^{L-1} \sum_{x=0}^{2^l-1} \sum_{y=0}^{2^l-1} \sum_{t=0}^{2^l-1} \langle \boldsymbol{z}_i^l(x,y,t) \boldsymbol{z}_j^l(x,y,t) \rangle, \tag{4.11}$$

where $\langle \boldsymbol{z}_i, \boldsymbol{z}_j \rangle = \boldsymbol{z}_i^T \boldsymbol{z}_j$ and $\boldsymbol{z}_i^l(x,y,t)$ is the vector of pooled sparse codes in the (x,y,t)-th subvolume of the sequence at scale l. Although linear, this kernel has been shown to achieve high classification accuracy when applied directly to sparse representations, by maintaining a low training and testing cost [30].

Since the video sequences in general come from $C > 2$ classes, a multi-class SVM classification problem should be solved. The approach used is the one-against-one strategy, where an SVM is constructed for every pair of classes. The number of SVMs to be trained in this approach is equal to $C(C-1)/2$. Let also $(k,\ l)$ be a pair of classes and $d_{k,l}(\cdot)$ the associated discriminant function. Then, given the query signature $\boldsymbol{z}_{query}$, if $d_{k,l}(\boldsymbol{z}_{query}) > 0$ a vote is assigned to the k-th class, otherwise the vote is given to the l-th class. The process is repeated for each pair of classes and finally, the class with the *maximum number of votes* is assigned to the query $\boldsymbol{z}_{query}$.

The proposed system is summarized in the block diagram of Fig. 4.2.

Chapter 5

Experimental evaluation

In this chapter the proposed compressive video classification systems are evaluated in terms of classification accuracy. In particular, our database consists of 8 classes from the UCF50 dataset, namely "Baseball Pitch", "Bench Press", "Biking", "Breast Stroke", "Clean and Jerk", "Diving", "Drumming" and "Fencing". This dataset includes videos categorized in classes corresponding to different actions and is particularly challenging due to large variations in camera motion, object appearance and pose, as well as the illumination conditions. Each class consists of 50 video sequences with 100 frames per sequence. A preprocessing step is applied on each frame, by converting into grayscale and downsampling at 128×128 pixels. The classification accuracy is expressed in terms of the average success rate, which is defined by

$$success\ rate = \frac{number\ of\ correctly\ classified\ sequences}{total\ number\ of\ query\ sequences}\ . \tag{5.1}$$

5.1 Frame-based CVC performance

In the case of the frame-based CVC presented in Chapter 2, a distinct block Walsh-Hadamard measurement matrix Φ is used in each run while the sampling ratio M/N varies in $[0.001, 0.10]$. For each class we execute 20 Monte-Carlo runs, where in each run a different separation of the 50 videos in T training and $50 - T$ testing samples is generated, with $T \in \{10, 25, 40\}$. The overall success rate averaged over the 20 Monte-Carlo runs and the 8 classes, as a function of the sampling ratio M/N is depicted in Fig. 5.1. Moreover, three classification methods are compared, namely the NN, the multi-class SVM and the SRC using the OMP.

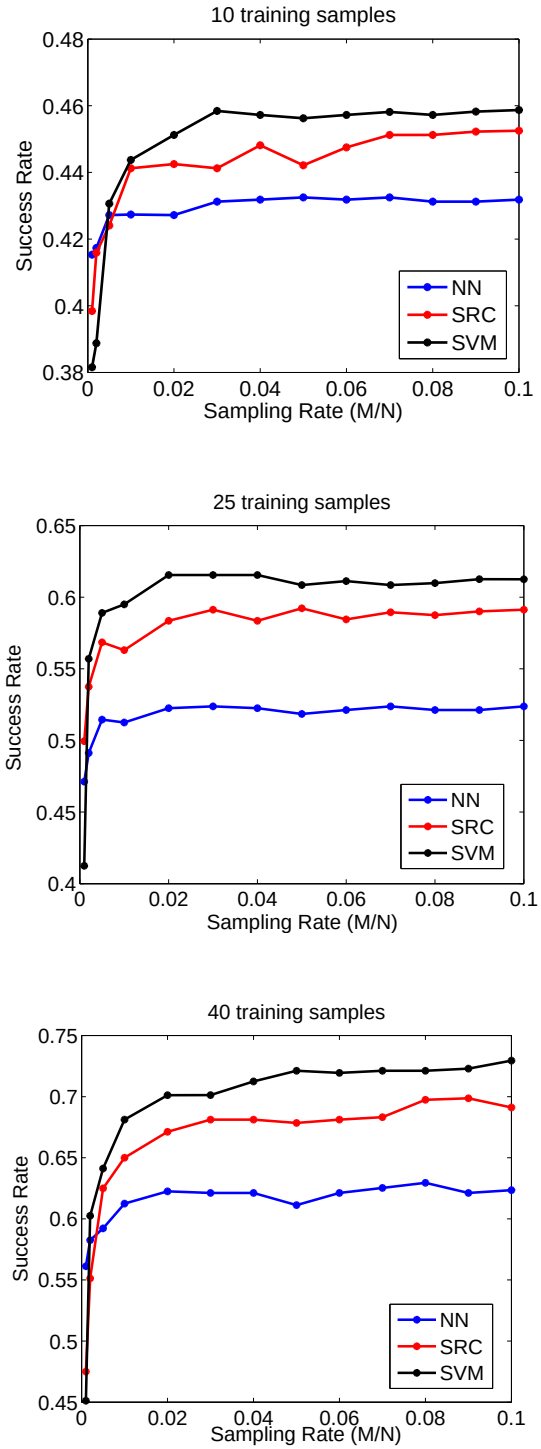


Figure 5.1: Total mean success rate as a function of the sampling rate, for 8 classes and three methods (NN,SVM,SRC) using CS features.

As it is expected, the classification accuracy increases as the sampling ratio M/N and the number of training samples increases. In addition, we observe that the performance of the pairwise voting approach employed by the multi-class SVM is superior to the other two classification methods, followed by the SRC approach while the NN classifier that applies the most simplistic decision metric appears to be the most prone to misclassification errors. More specifically, for the case of 40 training samples the maximum accuracy for $M/N = 10\%$ is around 63% for the NN classifier and 69% and 74% for the case of the SRC and SVM respectively.

Furthermore, the confusion matrices between the 8 classes by fixing $M/N = 10\%$ for the case of linear SVM, SRC and NN are shown in Table 5.1, Table 5.2 and Table 5.3 respectively.

Table 5.1: Confusion matrix for the SVM method with $M/N = 10\%$ and 40 training samples.

	Baseball pitch	Bench press	Biking	Breast stroke	Clean and jerk	Diving	Drumming	Fencing
Baseball pitch	**0.75**	0	0.12	0.04	0.06	0.02	0.01	0
Bench press	0	**0.79**	0.08	0.04	0.03	0.04	0	0.02
Biking	0.09	0	**0.75**	0.07	0.06	0	0.03	0.01
Breast stroke	0	0.08	0.06	**0.76**	0.06	0.04	0	0
Clean and jerk	0.02	0.03	0.04	0.09	**0.78**	0	0.04	0
Diving	0	0.02	0.14	0.02	0.08	**0.73**	0	0.01
Drumming	0.01	0.01	0	0.09	0.03	0.01	**0.76**	0.06
Fencing	0.05	0.05	0.12	0.04	0.08	0.01	0.02	**0.63**

Table 5.2: Confusion matrix for the SRC method with $M/N = 10\%$ and 40 training samples.

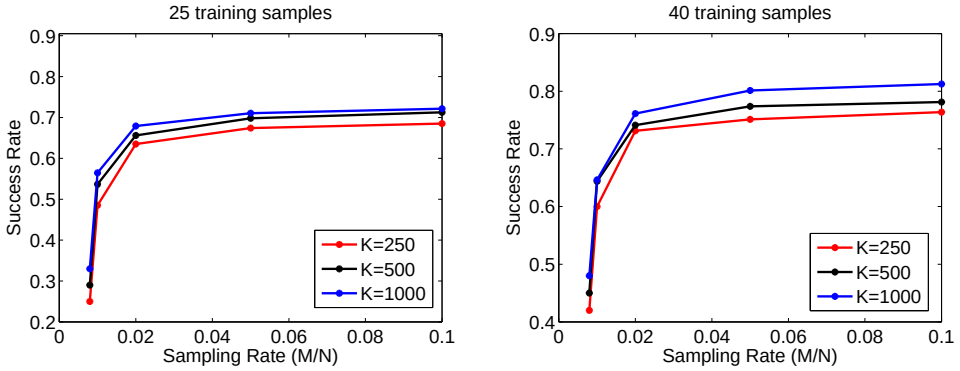
	Baseball pitch	Bench press	Biking	Breast stroke	Clean and jerk	Diving	Drumming	Fencing
Baseball pitch	**0.67**	0	0.15	0.08	0.06	0.01	0.03	0
Bench press	0	**0.75**	0.09	0.04	0.03	0.04	0	0.05
Biking	0.07	0.01	**0.69**	0.08	0.08	0	0.07	0
Breast stroke	0	0.09	0.12	**0.71**	0.05	0.02	0	0.01
Clean and jerk	0.05	0.04	0.05	0.08	**0.72**	0	0.06	0
Diving	0.03	0	0.14	0.08	0.03	**0.66**	0.02	0.04
Drumming	0.02	0.04	0	0.10	0.05	0.01	**0.69**	0.09
Fencing	0.08	0.05	0.12	0.05	0.07	0	0.02	**0.61**

Table 5.3: Confusion matrix for the NN method with $M/N = 10\%$ and 40 training samples.

	Baseball pitch	Bench press	Biking	Breast stroke	Clean and jerk	Diving	Drumming	Fencing
Baseball pitch	**0.66**	0.02	0.15	0.04	0.06	0.02	0.04	0.01
Bench press	0.01	**0.64**	0.09	0.07	0.09	0.05	0	0.05
Biking	0.08	0	**0.66**	0.08	0.10	0.02	0.06	0
Breast stroke	0.05	0.10	0.06	**0.65**	0.08	0.02	0	0.04
Clean and jerk	0	0.06	0.15	0.05	**0.69**	0	0.04	0.01
Diving	0.03	0.05	0.10	0.07	0.08	**0.62**	0.02	0.03
Drumming	0.05	0.08	0	0.12	0.03	0.05	**0.60**	0.07
Fencing	0.01	0.02	0.12	0.10	0.07	0.05	0.05	**0.58**

5.2 Block-based CVC performance

In the following the proposed block-based CVC presented in Chapter 3 is evaluated. A distinct block Walsh-Hadamard measurement matrix Φ is used in each run while the sampling ratio M_B/N_B varies in $[0.008, 0.10]$ and accordingly the total measurement rate M/N varies in the same interval. For each class we execute 20 Monte-Carlo runs, where in each run a different separation of the 50 videos in T training and $50-T$ testing samples is generated, with $T \in \{25, 40\}$. More specifically, we investigate the effect of four parameters on the success rate, namely: i) the number K of dictionary atoms, ii) the size N_B of the block, iii) the number L of the levels of the spatiotemporal pyramid and iv) the pooling function. During the sparse coding phase of the dictionary learning step the OMP algorithm is used. The sparsity level (number of non-zero coefficients) is fixed to 20 which yields empirically good results.



(a) (b)

Figure 5.2: Total mean success rate as a function of the sampling rate, for 8 classes, block size 16×16 pixels, $L = 2$ spatiotemporal pyramid levels, max absolute pooling and 3 different sizes of dictionary $K = \{250, 500, 1000\}$.

In Fig. 5.2 is depicted the overall success rate averaged over 20 Monte-Carlo runs for the 8 classes as a function of the sampling ratio M/N for a block size of 16×16 pixels, $L = 2$ spatiotemporal pyramid levels, max absolute pooling and 3 different sizes of dictionary $K = \{250, 500, 1000\}$. As it is expected the success rate increases as the sampling rate M/N and the number of training samples increases. As regards the dictionary size we observe that the discriminant power of the algorithm increases with the increase of the number of atoms. Specifically, for a dictionary with $K = 1000$ atoms the success rate is 72% for 25 training samples and 81.2% for 40 training samples.

Table 5.4: Total mean success rate for different block sizes and different number of atoms K for $L = 2$ pyramid levels, sampling rate $M/N = 0.1$ for (a) 25 and (b) 40 training samples

(a)

K	Block size 16×16	Block size 32×32
250	68.4%	62.4%
500	71.0%	64.3%
1000	72.0%	65.9%

(b)

K	Block size 16×16	Block size 32×32
250	76.1%	61.2%
500	78.4%	66.5%
1000	81.2%	70.4%

In Table 5.4 the total mean success rate for 2 different block sizes, namely 16×16 and 32×32 pixels and 3 dictionary sizes $K = \{250, 500, 1000\}$ is illustrated. We observe that for a smaller block size a higher success rate is achieved for all dictionary sizes compared to a larger one. A smaller size of block can therefore omit irrelevant and noisy information.

Table 5.5: Total mean success rate for different pyramid levels and different number of atoms K for block size 16×16, sampling rate $M/N = 0.1$ for (a) 25 and (b) 40 training samples

(a)

K	Pyramid levels $L = 1$	Pyramid levels $L = 2$
250	57.7%	68.4%
500	62.1%	71.0%
1000	65.5%	72.0%

(b)

K	Pyramid levels $L = 1$	Pyramid levels $L = 2$
250	62.3%	76.1%
500	70.3%	78.4%
1000	72.3%	81.2%

In Table 5.5 the effect of pyramid levels L is investigated. It is shown that the increase in pyramid levels of the spatio-temporal pyramid representation better preserves the layout of the descriptors, increasing the accuracy of the system.

In Table 5.6 the use of average and max absolute pooling is compared. It is obvious that max pooling produces better performance in all cases. Specifically, there is a difference of about 3-4% between the two pooling approaches. The increased performance of max pooling can be attributed to its robustness to local spatio-temporal variations and is consistent with the results in [30].

Table 5.6: Total mean success rate using different pooling methods for block size 16×16 and sampling rate $M/N = 0.1$.

	K	25 training samples	40 training samples
Max Pooling	250	68.4%	76.1%
	500	71.0%	78.4%
	1000	72.0%	81.2%
Avg. Pooling	250	64.2%	72.3%
	500	66.3%	75.2%
	1000	68.1%	76.4%

5.3 Comparative evaluation

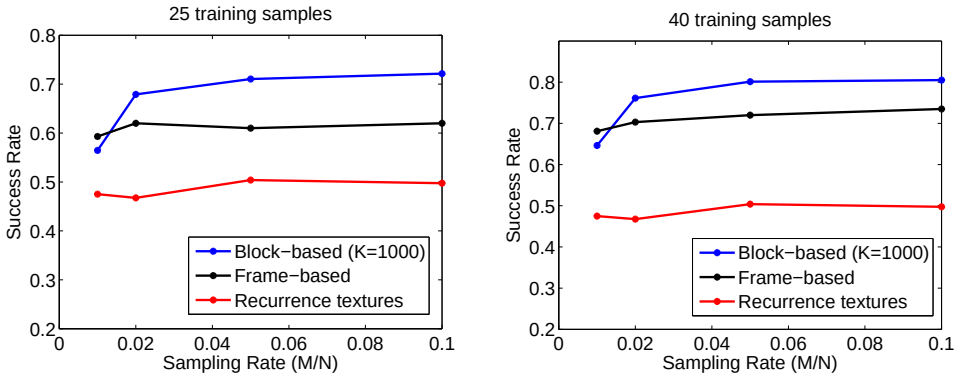
In this section we intend to compare the proposed methods with the method in [27] by means of total mean success rate. The non-thresholded recurrence textures computed from the first derivative of frame measurements are fed to the local binary patterns (LBP) algorithm. Several configurations were tested by varying the number of binary patterns, showing that a normalized histogram of 38 binary patterns was sufficient to represent each sequence (as in [27]). In order to improve the performance of the original scheme we use an SVM classifier with a χ^2 kernel,

$$K(x_i, x_j) = 1 - \sum_{k=1}^{N} \frac{(x_i(k) - x_j(k))^2}{\frac{1}{2}(x_i(k) + x_j(k))} \tag{5.2}$$

which is commonly prefered in case of histogram matching, instead of the simple nearest-neighbor used in the original study in [27].

The performance of both proposed systems is superior to that of [27] across all sampling rates as it is shown in Fig. 5.3, noted as "recurrence textures" in the figure legend. The performance is degraded because of the fact that the method in [27] assumes a static background for the video sequences, so that the features are mainly sensitive to the movement of the scene. However, in the dataset used the background includes useful information that can be of discriminative value for the classification procedure.

As a second comparative evaluation we are interested in comparison with a video classification method exploiting the full resolution video data, in order to illustrate the efficiency of the proposed method. For this, we extract the histogram of oriented gradients (HOG) and histogram of optical flow (HOF) accumulated in space-time neighborhoods of interest points as described in [8]. In particular, we use dense sampling to



(a) (b)

Figure 5.3: Total mean success rate as a function of the sampling rate, for 8 classes, and different compressive approaches for (a) 25 training and (b) 40 training samples.

extract video blocks in regular positions and scales in space and time, so there are five dimensions to sample from: (x, y, t, σ, τ), where σ and τ are the spatial and temporal scale, respectively. The minimum size of a 3D block is 18×18 and there is a 50% overlap in spatial and temporal sampling, as proposed in [34]. Each video block is subdivided into an $n_x \times n_y \times n_t$ grid of cells. For each cell, a 4-bin HOG and a 5-bin HOF are computed and the results are concatenated in a single feature vector. We use the original grid parameters $n_x = n_y = 3$ and $n_t = 2$ and the implementation available online [1].

Then, we use a bag of local spatio-temporal features to represent each sequence. k-means clustering is used for creating a visual vocabulary that is used for quantizing the features. At the end, each sequence is represented by a frequency histogram over the visual words. We choose to train a vocabulary of $K = 1000$ visual words using only a subset of $100,000$ randomly selected training features in order to reduce complexity. The features are assigned to the closest in the Euclidean distance sense visual word and, finally, a non-linear SVM with χ^2-kernel is used for the classification phase.

[1] http://lear.inrialpes.fr/software

Table 5.7: Total mean success rate for different methods and different number of training samples. For the case of compressive video classification the sampling rate is $M/N = 0.1$, the block size is 16×16 and the size of the dictionary is $K = 1000$.

	25 training samples	40 training samples
Frame-based	62.3%	73.8%
Block-based	72.0%	81.2%
HOG/HOF	77.2%	84.9%

In Table 5.7 the performance of the compressive video classification systems and the classification based on HOG/HOF descriptors is illustrated. The total mean success rate for the HOG/HOF features that are extracted from the full-resolution data are able to better represent the video sequences, so the mean success rate in this case is increased by around 4-5%. However, the slight degradation in performance of the compressive classification schemes is acceptable based on the fact that only the 10% of the initial data size is used, proving their efficiency.

Chapter 6

Conclusions and Future work

In this thesis, two compressive video classification methods were introduced. More specifically, the design of both proposed CVC systems is primarily based on the assumption of limited resources, where the video data are captured directly in the CS domain using a single pixel camera.

In the first system a supervised learning approach is followed, where each column of the training dictionary is formed by the CS measurement vectors over all the frames of a given training video sequence. Finally, the estimated class is obtained by means of typical classification methods, namely, the NN and the multi-class SVM. An alternative way is also tested, where the classification problem is reduced to a problem of reconstructing a sparse class-indicator vector as the solution of a convex optimization problem.

In the second system a spatio-temporal pyramid matching approach is proposed. The frames of the sequence are initially sampled according to the block-based compressive sampling scheme and their sparse codes are computed following an unsupervised dictionary learning approach. The estimated class is obtained by means of a multi-class SVM with a spatio-temporal pyramid kernel.

The experimental results revealed that the classification performance is robust to the number of samples captured, even at very low sampling rates at the order of 2%, significantly smaller than the rates required for solving the problem of sparse reconstruction. Apart from that, the block-based approach showed an increased performance compared to the simple frame-based for the expense of increased complexity, mainly due to the dictionary learning step.

As future work, it would be useful to incorporate the color information in the generation of the CS features that could increase the classification margin. Under a

single-pixel camera (SPC) assumption this could be possible by adding a color-rotating filter in the typical SPC architecture, as described in [35]. Additionally, the use of a classification-oriented dictionary should be investigated in the case of the block-based CVC, by either enforcing directly the dictionary to be discriminative, or by making the sparse coefficients discriminative. Recent works [36, 37] reveal that the incorporation of class label information in the dictionary training phase can offer a boost to the performance of the classification scheme compared to a simple representative dictionary.

Bibliography

[1] D. Brezeale and D. J. Cook, "Automatic video classification: A survey of the literature," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 38, no. 3, pp. 416–430, 2008. 2

[2] W. Zhou, A. Vellaikal, and C. Kuo, "Rule-based video classification system for basketball video indexing," in *Proceedings of the 2000 ACM workshops on Multimedia*, pp. 213–216, ACM, 2000. 2

[3] M. Rautiainen and D. Doermann, "Temporal color correlograms for video retrieval," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 1, pp. 267–270, IEEE, 2002. 2

[4] M. J. Roach, J. Mason, and M. Pawlewski, "Video genre classification using dynamics," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 3, pp. 1557–1560, IEEE, 2001. 2

[5] N. Dimitrova and F. Golshani, "Motion recovery for video content classification," *ACM Transactions on Information Systems (TOIS)*, vol. 13, no. 4, pp. 408–439, 1995. 2

[6] A. Efros, A. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 726–733 vol.2, 2003. 2

[7] L.-Y. Duan, M. Xu, Q. Tian, C.-S. Xu, and J. S. Jin, "A unified framework for semantic shot classification in sports video," *Multimedia, IEEE Transactions on*, vol. 7, no. 6, pp. 1066–1083, 2005. 2

[8] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human

actions from movies," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008. 2, 34

[9] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, 2006. 2, 9, 19

[10] http://dsp.rice.edu/cscamera. 2

[11] R. Baraniuk, "Compressive sensing [lecture notes]," *Signal Processing Magazine, IEEE*, vol. 24, no. 4, pp. 118–121, 2007. 7

[12] T. T. Do, T. D. Tran, and L. Gan, "Fast compressive sampling with structurally random matrices," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pp. 3369–3372, IEEE, 2008. 8, 16

[13] D. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, 2006. 9, 19

[14] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *Information Theory, IEEE Transactions on*, vol. 53, no. 12, pp. 4655–4666, 2007. 9, 19

[15] D. L. Donoho, Y. Tsaig, I. Drori, and J. luc Starck, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," tech. rep., 2006. 9, 19

[16] R. Calderbank, S. Jafarpour, and R. Schapire, "Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain," tech. rep., 2009. 9

[17] J. Haupt, R. Castro, R. Nowak, G. Fudge, and A. Yeh, "Compressive sampling for signal classification," in *Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference on*, pp. 1430–1434, 2006. 9

[18] K. Engan, S. Aase, and J. Hakon Husoy, "Method of optimal directions for frame design," in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, vol. 5, pp. 2443–2446 vol.5, 1999. 10

[19] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *Signal Processing, IEEE Transactions on*, vol. 54, pp. 4311–4322, Nov. 2006. 10

[20] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *In NIPS*, pp. 801–808, NIPS, 2007. 10, 13

[21] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 689–696, 2009. 11, 24

[22] M. A. Davenport, M. F. Duarte, M. B. Wakin, J. N. Laska, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "The smashed filter for compressive classification and target recognition," in *Proceedings of Computational Imaging V at SPIE Electronic Imaging*, Jan. 2007. 11

[23] M. F. Duarte, M. A. Davenport, M. B. Wakin, J. N. Laska, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "Multiscale random projections for compressive classification," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2007. 12

[24] L. Liu and P. Fieguth, "Texture classification using compressed sensing," in *Proceedings of the 2010 Canadian Conference on Computer and Robot Vision*, CRV '10, pp. 71–78, IEEE Computer Society, 2010. 12

[25] D. Thanou and P. Frossard, "Compressed classification of observation sets with linear subspace embeddings," in *ICASSP'11*, pp. 1353–1356, 2011. 12

[26] A. C. Sankaranarayanan, P. K. Turaga, R. G. Baraniuk, and R. Chellappa, "Compressive acquisition of dynamic scenes," in *Computer Vision–ECCV 2010*, pp. 129–142, Springer, 2010. 12

[27] K. Kulkarni and P. Turaga, "Recurrence textures for human activity recognition from compressive cameras," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pp. 1417–1420, IEEE, 2012. 13, 34

[28] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, pp. 210–227, Feb. 2009. 13

[29] J. Xie, L. Zhang, J. You, and D. Zhang, "Texture classification via patch-based sparse texton learning," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pp. 2737–2740, IEEE, 2010. 13

[30] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 1794–1801, IEEE, 2009. 13, 25, 27, 33

[31] B. Zhao, L. Fei-Fei, and E. P. Xing, "Online detection of unusual events in videos via dynamic sparse coding," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 3313–3320, IEEE, 2011. 13

[32] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, pp. 994–1000 vol. 2, 2005. 25

[33] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *International Conference on Machine Learning*, pp. 111–118, 2010. 25

[34] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, C. Schmid, *et al.*, "Evaluation of local spatio-temporal features for action recognition," in *BMVC 2009-British Machine Vision Conference*, 2009. 35

[35] P. Nagesh and B. Li, "Compressive imaging of color images," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pp. 1261–1264, IEEE, 2009. 38

[36] Z. Jiang, Z. Lin, and L. S. Davis, "Learning a discriminative dictionary for sparse coding via label consistent k-svd," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1697–1704, IEEE, 2011. 38

[37] I. Ramirez, P. Sprechmann, and G. Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3501–3508, IEEE, 2010. 38