Action Recognition in Videos Using Geometrical Representation and Kernel Logistic Regression with Feature Relevance

par

Ouiza Ouyed

Thèse présentée au Département d'informatique et d'ingénerie
en vue de l'obtention du grade de Docteur ès sciences (Ph.D.)

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

Gatineau, Québec, Canada, décembre 2018

Ce mémoire a été évalué par un jury composé des personnes suivantes :

Professeure Nadia Baaziz	Présidente du jury
Professeur Mohand Said Allili	Directeur de recherche
Professeur Marek Zaremba	Membre interne du jury
Professeur Abdessamad Ben Hamza	Membre externe du jury

SOMMAIRE

La reconnaissance d'activités humaines en vidéos est un problème très important en vision par ordinateur. Le défi principal dans ce domaine est le développement d'algorithmes capables d'analyser automatiquement les vidéos pour reconnaître et interpréter les activités générées. Récemment, les méthodes statistiques ont montré leur efficacité pour classer et prédire des activités à différents niveaux de complexité (ex. gestes, actions, interactions). Cependant, ces méthodes se basent souvent sur des représentations d'activités utilisant des vecteurs de caractéristiques à très haute dimensions, noyant ainsi dans le bruit l'information utile pour discriminer les activités. Notons également que les bases de données d'activités sont parfois composées de peu de données, ce qui peut entraîner un problème de sur-apprentissage lors de l'entrainement des modèles de classification (c-à-d., capacité de généralisation limitée).

Dans cette thèse, nous abordons ces problèmes avec de nouvelles approches statistiques que nous avons appliquées pour la reconnaissance d'actions et les interactions entre personnes. En premier temps, nous avons proposé un nouveau modèle de classification d'actions (FR-MKLR), basé sur la régression logistique multinomiale à noyaux et intégrant la pertinence de caractéristiques sous forme de poids dans la fonction du noyau. Ce modèle permet de réduire l'effet des caractéristiques indésirables en inhibant leurs poids durant l'apprentissage. Nous avons appliqué avec succès notre modèle pour la reconnaissance d'actions simples d'individus (ex. marcher, courir, tomber, etc.), en se basant sur une nouvelle description géométrique de ces dernières basée sur la forme contextuelle spatio-temporelles des silhouettes. Dans un second temps, nous avons généralisé notre modèle de classification FR-MKLR en intégrant la pertinence de groupes de caractéristiques. Le nouveau modèle, baptisé GFR-MKLR, permet de rechercher la pertinence de groupes de caractéristiques contribuant comme des facteurs latents à part entière pour la discrimination de classes. La nécessité de développer un tel modèle vient du fait que pour les activités humaines, en particulier les interactions entre deux individus, plusieurs caractéristiques de mouvement peuvent être regroupées en fonction des parties du corps humain les ayant générées via des gestes (ex. mains, jambes, tête, etc.). Nous avons appliqué avec succès notre modèle pour la reconnaissance d'interations entre deux personnes (ex. saluer, frapper, etc.). Nous avons testé de manière extensive nos approches FR/GFR-MKLR sur différents ensembles de données standard d'actions et d'interactions entre individus. Les résultats obtenus ont démontré les capacités de nos approches et leur performance par rapport aux travaux récents de la littérature.

ABSTRACT

Human activity recognition in videos is a very important problem in computer vision. The main challenge in this area is how to develop algorithms capable of analyzing automatically videos to recognize and interpret generated activities. Recently, statistical methods have shown their effectiveness to classify and predict activities at different levels of complexity (e.g., gestures, actions, interactions). However, these methods are often based on representations of activities using very high-dimensional feature vectors which overwhelm in noise useful information for classification. Note also that activity databases contain sometimes insufficient number of data, which can lead to classification over-fitting during model training (i.e., limitation in generalization capability).

In this thesis, we tackled these problems with new statistical approaches that we have applied to single actions and inter-person interactions recognition. First, we have proposed a new model action classification (FR-MKLR) which is based on kernel multinomial logistic regression by integrating features relevance in the form of weights embedded in the kernel function. This model reduces the contribution of non-relevant features by inhibiting their weights during the training process. We have successfully applied our model for recognizing simple human actions (e.g. walking, running, falling, etc.), using a new geometric description of actions based on the spatiotemporal shape context of silhouettes. Second, we have generalized the FR-MKLR model by integrating the relevance of groups of features. The new model, called GFR-MKLR, allows to encode the relevance of feature groups constituting fully-fledged latent factors for class discrimination. The need to develop such a model stems from the fact that in activities, particularly involving interactions between individuals, several motion features can be grouped according to the human limbs having them generated via gestures (e.g. hands, legs, head, etc.). We have applied successfully our model for the recognition of interactions between two people (e.g. greeting, punching, etc.). We extensively tested our FR/GFR-MKLR models on several standard datasets for action and interaction recognition. Obtained results have demonstrated the capabilities of our models and their performance compared to recent work in the literature.

REMERCIEMENTS

Je tiens à remercier mon directeur de thèse Mohand Saïd Allili, Professeur à l'Université du Québec en Outaouais, qui m'a encadrée tout au long de cette thèse. Qu'il soit aussi remercié pour sa disponibilité permanente et pour les nombreux encouragements qu'il m'a prodigués ainssi que pour ses conseils qui m'ont permis de bien mener ce travail. Mes remerciements vont également aux membres du jury, les professeurs: Nadia Baaziz, Marek Zaremba et Abdessamad Ben Hamza qui ont accepté et pris le temps d'évaluer ce travail. Des remerciements particuliers vont à mes parents, mon frère et mes sœurs, sans oublier mon époux Said Messaoui qui m'a soutenue et encouragée durant mes études. Je ne pourrais terminer ce paragraphe sans remercier mes deux anges Léa et Tiziri d'avoir illuminé ma vie et de l'avoir remplie de joie.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMES AND SYMBOLES

HCI Human Computer Interaction

CCTV Closed-Circuit Television

SIFT Scale-Invariant Feature Transform

HOG Histogram of Gradient

STIP Spatiotemporal Interest Points

HOF Histogram of Flow

MBH Motion Boundary Histogram

SVM Support Vector Machine

NN Nearest Neighbors

KNN k-Nearest Neighbors

PCA Principal Component Analysis

LPP Locality Preserving Projection

LLE Locality Linear Embedding

LASSO Least Absolute Shrinkage and Selection Operator

SC Shape Context

ASC Action Shape Context

fr-MKLR feature relevance for Multinomial Kernel Logistic Regression

Gfr-MKLR Group of features relevance for Multinomial Kernel Logistic Regression

HAR Human Action Recognition

LR Logistic regression

HMM Hidden Markov Models

BN Bayesian Networks

DBN Dynamic Bayesian Networks

SCFG Stocastic Context-Free Grammar

MEI Motion Energy Image

MHI Motion History Image

PCOG Pyramid Correlogram of Oriented Gradient

OF Optical Flow

ROI Region Of Interest

MHV Motion History Volumes

STV Spatiotemporal Volumes

RGB Red, Green, Bleu

CNN Convolutional Neural Networks

RNN Recurrent Neural Networks

LTC-CNN Long-term Temporal Convolutional networks

TDD Trajectory-pooled Deep convolutional Descriptor

ANN Artificial Neural Networks

MLNN Multilayer Neural Network

DBAN Dynamic Bayesian Actio Network

MIL Multiple Instance Learning

PSM Pictorial Structure Model

DLN Deep Learning Network

LSTM Long-Short Term Memory

AR Action Recognition

IR Interaction Recognition

MAP Maximum a Posteriori

SMLR Sparse Multinomial Logistic Regression

DAGSVM Directed Acyclic Graph Support Vector Machine

KSVM Kernel Support Vector Machine

NLL Negative Log-Likelihood

MKLR Multinomial Kernel Logistic Regression

RBF Radial Basis Function

N-R Newton-Raphson

GMM	Gaussian Mixture Models
CA	Classification Accuracy
BS	Background Subtraction
MoGG	Mixture of Generalized Gaussian
RBM	Restricted Boltzmann Machines
CTM	Correlated Topic Model
3D-MC	3D Motion Contex
HMC	Harmonic Motion Contex
RT	R-Transform
DFT	Discreet Fourier Transform
HOD	Histogram Of Displacement
HOC	Histogram Of Curvature
CV	Cross Validation
LOOCV	Leave One Out Cross Validation
α_i	Training examples weights
$\mathbb{R}$	Space of Real Number
β	Logistic regression Coefficients
$\mathcal{K}$	Kernel Function
λ	Regularization parameter
a_i	real-valued coefficients

σ Kernel width

Ψ Features weights vector

β Approximation parameter for ℓ_0-"norm"

μ Regularization parameter

$\tilde{g}$ Gradient vector

$\tilde{H}$ Hessian matrix

$\pi_{j,k}$, $\mu_{j,k}$ and $\Sigma_{j,k}$ priori probability, the mean vector and covariance matrix of the *k*th component of the Gaussian mixture

δ Overlapping step

n_r Radial bins

n_θ Angular bins

(v, h) visible and hidden units

b_i and b_j Biases of visible and hidden units

$w_{i,j}$ Weight between visible and hidden units

Chapter 1

Introduction

1.1 Action recognition problem

The ever-increasing use of images and videos for individual and public purposes and the availability of digital cameras at lower cost have considerably increased the daily volume of information stored and analyzed. Images and videos are becoming an essential tool in different domains such as video surveillance [17], entertainment and healthcare systems [18], to name a few. In video surveillance, for example, automatic detection of abnormal activities can be used to alert authorities of potential criminal or dangerous behaviors. In human computer interaction, activity recognition can be used to improve the human/computer interaction (HCI) experience [19], while in entertainment it can be used to create more realistic characters in video games and augmented reality [20]. In healthcare systems, activity recognition can help in the rehabilitation and monitoring of patients [18].

Human vision system has a unique ability to understand and interpret the dynamics generated by moving objects in video sequences. However, the amount of information to be processed and interpreted by human operators, for monitoring and analysis purposes, can lead to high costs, not to mention errors caused by human fatigue and distraction. Despite the recent advances made in video processing, computers are still far from reaching the human abilities in visual recognition. One of the main objectives in computer vision is to develop methods and algorithms based on image/video processing to try to match these abilities as closely as possible. In recent years, automatic human activity recognition has drawn much attention in the field of computer vision and image processing due to the growing demand from applications such as video surveillance, HCI and robotics. For example, the proliferation of CCTV camera systems in video surveillance has brought the need for real-time algorithms to perform activity recognition and detect abnormal behaviors [21, 22, 23].

Depending on the complexity of motion patterns, activities captured in videos can range from simple gesture to ones involving two or more humans/objects. A single person activity can be composed of a sequence of *elementary actions* obeying certain spatial and temporal relationships. An *elementary action*, in turn, can be composed of multiple temporally/spatially organized *gestures*. When an activity involves at least two persons (resp. a person and an object), it is referred to as human/human (resp. human/object) interaction [24]. Contrarily to the periodic nature of elementary actions, where repetitive gestures can be made for carrying an action (e.g., walking, boxing, etc.), human/human interactions can generate non-periodic motion patterns since each person can perform a different sequence of gestures [24, 25, 26]. Since actions are the building blocks of human activities, it is of prominent importance to have efficient and fast methods for real-time basic (elementary) action recognition. Generally, the task of representing and recognizing human actions is a challenging problem due to the following factors:

- *Variation in action generation*: the same action can be performed by different humans in different ways and speeds. This causes a high overlapping between different classes of actions (e.g., walking and running). Therefore, good action recognition requires approaches that are able to take into account intra-class variations and adequately identify inter-class differences.

- *Variation in action environment:* environment here refers to the context where the action is performed. False motion patterns generated by moving cameras, cluttered/non-stationary backgrounds and illumination changes can distract action recognition systems and lead to many false positives. Other difficulties can come also from viewing angle changes and partial/total object occlusions that may drastically change the visual appearance of the objects and actions [27, 28].

- *Action representation:* several methods describe actions using high-dimensional vector representation containing features extracted from videos (e.g., SIFT, HOG [24], etc.). However, when using features extracted from raw video data, relevant information for action discrimination can be overwhelmed by a huge amount of noise and non-informative features. This, in turn, can prevent classifiers from efficiently separating action classes. Therefore, appropriate action representation should be cleared of non-relevant features before or during the classification process.

1.2 Action recognition systems

In order to perform human activity recognition, typical methods are generally decomposed into three main steps as shown in Figure 1.2.1:

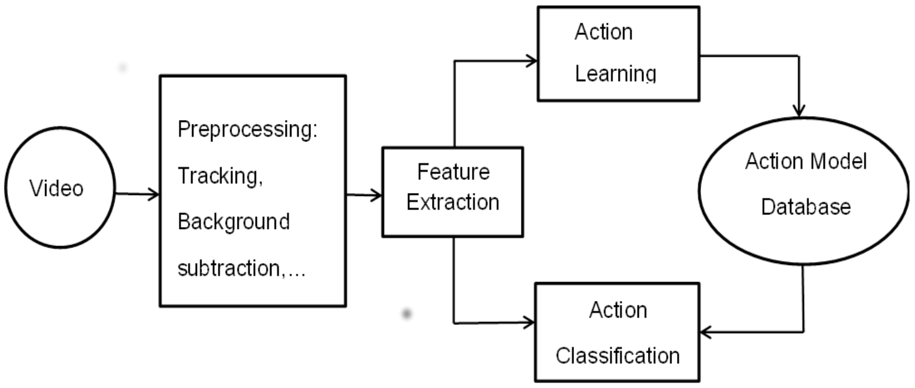


Figure 1.2.1 – A generic activity recognition system.

1. *Pre-processing*: Several methods for activity recognition require moving object detection or background subtraction for action representation. However, changes in the environment due to illumination changes and camera motion/vibrations can generate noise and false objects or motion patterns, decreasing the quality of extracted features for activity recognition. Also, the presence of shadows and illumination changes pose major difficulties for distinguishing real from false objects [29]. In order to remove background noise and instabilities, pre-processing techniques that comprise filtering, video stabilization and motion compensation can be used to enhance and prepare video sequences for reliable feature extraction.

2. *Features for action representation*: Consists of extracting posture and motion cues that are discriminative for human actions [23]. Features can be considered as abstraction of raw data, which are used to represent and characterize actions in a compact way [30]. Therefore, the feature extraction process produces feature vectors that are a reduced representation of video sequences. In the past, several methods have proposed features for representing and recognizing human actions [24, 31, 1, 32]. Depending on the type of features used for activity description, we can categorize these methods into two main categories:

   (a) *Methods based on local features*: are used to represent local properties of actions and do not require in general to detect/localize humans. Methods such as scale invariant feature transform (SIFT) [33], space time interest points (STIP) [1] and histogram of oriented gradient (HOG) [34] use local 2D windows in images or 3D cuboids in video streams to extract features for action recognition. These representations have some invariance to viewpoint and appearance changes, but do not capture the whole body motion or detailed motion description of body parts in deformable objects. Note also that there is some redundancy in generated points such as SIFT or STIP features since several points can be extracted at the same location. Consequently, relevant information for action discrimination can reside in a handful of salient points buried in a huge number of non-informative ones.

(b) *Methods based on global features*: also called holistic representations [10, 7, 9], generally require object tracking and/or background subtraction to extract moving objects. These methods capture the spatial structure of human body by computing features on a regular grid bounding the detected object(s). Unlike local descriptors, global representations encode the whole body activity and give more information about deformable objects. These representations include silhouettes, color and motion segmentations [8]. To be robust to noise, partial occlusion and viewpoint changes, some methods include features such as optical flow, HOG and shape information for action description [35, 36].

(c) *Hybrid methods using dictionary learning*: recently, authors in [37, 38] have proposed to combine local and global descriptors to extract sparse representation based on *dictionary learning*. These methods construct feature spaces in the form of *vocabularies* which are suitable for activity recognition and classification [39]. However, vocabulary construction require a huge amount of training data consisting of video chunks related to activities to be classified. In addition, given that activities can exhibit a huge variability, it is difficult to build a common and stable vocabulary that can account for the variability within each activity category. Finally, these methods are usually computationally intensive.

3. *Activity learning and classification*: Consist of building statistical models that can be trained on labelled data and then used to classify new observations. A major challenge in these models is dealing with the variability in action classes that can be accentuated when actions are performed by different subjects and with different genders and clothing [24]. For example, actions that can be clearly and easily separable by humans (e.g., *boxing* and *hand-waving*) can be very difficult to classify by computers when generated with different view points. Moreover, depending on the context and/or culture, actions can be carried out with different speeds and styles. It is therefore important to design

Table 1.2.1 – Dimensionality of feature vectors given in the literature: S: represents the total number of scales, T represents the number of frames and n represents the number of considered codewords in the extracted feature).

Methods	features vector dimension
Blank *et al.* [10]	440
Sun *et al.* [40]	192
Zhao *et al.* [41]	1250
Klaser *et al.* [42]	960
Wang *et al.* [5]	(30, 96, 108, 192) for (trajectory, HOG , HOF , MBH)
Bregonzio *et al.* [4]	$(8S + 2) \times T$
Jiang *et al.* [43]	$\frac{n \times n}{2} \times 4 \times 4$

efficient classifiers capable of identifying relevant features characterizing each action category, while maintaining flexibility to cope with action variation.

In the past, several classifiers have been used for action recognition in videos, such as kernel-SVM [32], Nearest Neighbors (NN) [8] and Adaboost [31], to name a few. These methods are usually designed for binary classification problems, and use a *one-versus all* strategy to deal with multi-class cases. These methods have shown good performance for recognizing simple actions when videos are clutter- and noise free. However, this performance drastically decreases in the presence of noise, clutter and intra-class variation. Moreover, these methods do not efficiently handle high-dimensional data where useful information for action discrimination lies in a few dimensions which are overwhelmed by noise and irrelevant information occupying the remaining dimensions. Table 1.2.1 shows typical sizes of the most used descriptors for representing actions in the literature.

To mitigate the problem of high-dimensional data, dimensionality reduction techniques can be used in order to find a lower representation of actions and make classification faster and more efficient. For example, methods such as principal component analysis (PCA) [44], locality preserving projections (LPP) [45], Isomap [46] and locally linear embedding (LLE) [47] can be used to reduce the dimensionality of data. However, these methods are class-agnostic since they do not take into account the class labels. Consequently, they can eliminate useful information for action discrimination. Finally, dimensionality reduction methods applied as additional steps in the recognition process can increase the computation time, especially with large volumes of data.

Sparse models for classification have been proposed recently to cope with the problem of high-dimensional data [48, 49, 50]. Those methods incorporate directly selection of individual feature or groups of features in the classification process, which allows to enhance the classification accuracy. These methods include the *least absolute shrinkage and selection operator*(LASSO) [51, 52, 53] and the group LASSO [54]. Several methods have proposed models integrating sparsity in classification models for action recognition [55, 56]. These have been developed mainly with dictionary learning and support vector machines (SVMs) [38, 37] which use regularization terms in their objective functions, inducing sparsity. However, these methods are usually designed for linearly-separated classes and they are sensitive to the intra-class variability and inter-class overlapping. In addition, they require a huge amount of learning data for their training.

1.3 Contributions

Given the aforementioned challenges in action recognition, several research works have been proposed recently to improve action representation and learning [24, 27, 28, 23]. For the pre-processing step, more robust algorithms for object tracking and silhouettes extraction have been proposed [57, 58]. These can locate and track in real-time one or multiple targets in the presence of noise, illumination/background changes and camera vibration. For action representation, recent works have exploited new local and/or global features to describe activities in a unique and reliable way [24]. However, the problem of action variability still poses a problem since most descriptors do not provide invariance to intra-class variation. Finally, descriptors are usually high-dimensional and contain several noisy features which reduce classification accuracy. Using dimensionality reduction techniques is usually a necessary process before classification. However, for these techniques to be successful, they must encode efficiently discriminative information about classes while being able to deal with multi-class data in a computationally efficient way.

Based on these remarks, two major problems in our view remain among the most challenging for action recognition. The first problem concerns finding a good representation describing both *local* and *global* aspects of the actions. The second problem concerns building classification models that can select relevant features for efficient discrimination between multiple action classes. The training of such models should be made possible without requiring huge amounts of data, contrarily to methods such as dictionary-based learning [38, 37].

In this thesis, we address these problems by proposing novel approaches for action representation and classification that have several good properties for enhancing the accuracy of action recognition. We can briefly summarize our main contributions in the following points:

1. **Single action recognition:**

    - *Action representation*: We propose an object-based representation of actions based on shape context (SC) analysis of silhouettes [16]. Our descriptor, called *action shape context* (ASC), uses the variation of human shape over time to describe actions. It is constructed based on the analysis of the object contour with regard to its center of gravity in the video sequence. Among the properties of the ASC representation, it is invariant to changes in scale, translation and rotation of objects.

      Compared to the original version of the SC [16], the ASC has less dimensionality than SC and incorporates *local* and *global* information about the actions. *Local* information is obtained from analyzing local object motion of ASC bins over time, which is obtained by observing local object silhouette contour displacements. *Global* information is obtained by combining the local information in a spatial grid configuration which provides the global aspect of the actions performed by an object.

- *Action classification*: To efficiently classify actions into multiple categories, we propose sparse model (coined FR-MKLR) incorporating feature relevance directly in the classification model. Our model allows to reduce the effect of non-desirable features and boost the contribution of discriminative ones based on the training data. It is also robust to noise and easily generalizable to the multi-class case by encoding feature relevance (FR) for each class separately. Compared to several methods proposed in the literature, our model allows to perform an accurate action classification, even with few training data. Moreover, the results are more interpretable since feature weights are directly associated with the original dimensions of data. To the best of our knowledge, kernel logistic regression with feature relevance has never been used in the literature of action recognition.

2. **Inter-person interaction recognition:**

- *Interaction representation*: We propose an approach for inter-person interaction recognition by analyzing over time gestures performed by each person. Since gestures are related to local motion of body parts, we build a representation based on body parts trajectories which are extracted by tracking over time body joints position using the method proposed in [59]. We then combine shape and motion descriptors to build a feature group corresponding to each trajectory. Finally, the set of feature groups are concatenated following a predefined spatial structure to form the final representation, which is used to classify interactions.

- *Interaction classification*: By analyzing interactions at a gesture level, it can be assumed that the optimal sparsity will tend to involve clusters or groups corresponding to preexisting body parts [53]. Based on the proposed interaction representation, we propose a generalization of our classification models FR-MKLR to encode groups of features sparsity by assigning weights to entire groups instead of individual features as in FR-MKLR. We coin this new model GFR-MKLR. While the structure of the groups can be a priori known (a group correspond to all features associated with a part of the body performing a gesture), the subset of groups that is relevant for discriminating one interaction for another is a priori be unknown. The GFR-MKLR model aims at finding these groups directly from the training data and assigning weights to achieve the highest discrimination between interaction classes.

3. **Validation on standard datasets**: We first validate our classifier (FR-MKLR) on several synthetic and real world data from the UCI datasets [60]. We compare FR-MKLR to classifiers such as MKLR, K-SVM , LASSO, Naive Bayes. We then validate our action recognition method by using three datasets: KTH [34], UIUC [49] and the I3DPost multi-view database [61] by combining ASC representation with FR-MKLR. These datasets present several challenges such as different gender and size of persons generating the actions. UIUC and I3DPost present actions with different view points and KTH has 4 scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. We compared our approach to baseline and recent state-of-art methods dealing with action recognition. Finally, we evaluate the performance of method for inter-person interaction recognition using the UT-interaction dataset [26]. Experimental results show the effectiveness of our method compared to state-of-art ones.

1.4 Organization of the thesis

The thesis is organized as follows: Chapter 2 provides a literature overview of single action and inter-person interaction recognition using local and global video features. Chapter 3 presents a brief theory of multinomial kernel logistic regression and we present our proposed sparse models FR-MKLR and GFR-MKLR. Results obtained on synthetic and real-world data using FR-MKLR are presented and compared with popular supervised classifiers such as MKLR, K-SVM , LASSO, Naive Bayes. Chapter 4 presents our methods for recognizing single actions and inter-person interactions using FR-MKLR and GFR-MKLR, respectively. Finally, Chapter 5 concludes the work with a summary and discussion of the presented approaches and future work perspectives.

Chapter 2

A review on human action recognition

Human action recognition (HAR) is an important area of computer vision research, and is useful for several applications such as video surveillance, patient monitoring, robotics, and human-computer interfaces. Most of HAR systems require an automated recognition of high-level activities, composed of multiple simple (or atomic) actions of persons [24]. Therefore, action recognition is a cornerstone for general and complex activity recognition in videos. Methods for an analysis of simple actions of a single person are first presented in this chapter. Then, we present methods for recognizing inter-person interactions.

Over the recent years, numerous techniques have been proposed for human action recognition and several surveys have overviewed recent techniques [24, 27, 18, 28, 62]. Most of these techniques try to exploit spatial and temporal video information to design good representation for actions to facilitate their recognition. With advances in statistical methods and machine learning techniques, several works have successfully cast human activity recognition as a classification problem [24]. In these methods, action recognition systems are generally composed of two main steps:

- *Action representation*: consists of extracting compact descriptions of the actions using spatiotemporal information of videos. Depending on the way descriptors are extracted, they can be qualified either as local or global. Local descriptions are calculated on salient points or object/motion discontinuities, whereas global descriptors are calculated on the whole video data [28].

- *Action classification* : Given an action representation, a classifier can be trained on labeled examples of actions and then used on newly-observed videos to predict the categories of contained actions. Classification methods can be either discriminative or generative. Discriminative methods, such as support vector machines (SVM), K-nearest neighbors (KNN) and logistic regression (LR), do not assume distribution models of action classes. Generative models such hidden Markov models (HMM), Bayesian networks (BN) and stochastic context-free grammar (SCFG) assume the actions distribution is governed by parametric probability models.

One major challenge, however, remains for such methods, which consists of finding good representation and classification models that can account for the variability inside action categories. This variability stems from diverse factors such as the diversity in person gender, size, speed and gait, not to mention the presence of occlusions and viewpoint changes. In this chapter, we provide a comprehensive literature review on single human action and two persons interaction recognition by highlighting strengths and limitations of each method.

2.1 Action representation

Consists of extracting posture and motion features that are discriminative for human actions [23]. These features can be considered as abstraction of raw data which are used to represent and characterize the performed actions [30]. In other words, the process of feature extraction produces feature vectors that are a reduced representation of video sequences to describe the actions. In the past, several methods have been proposed for representing human actions [24, 31, 1, 32]. Depending on the type of features (measurements) used for action representation, we can categorize these methods into two main categories: *local* and *global* methods.

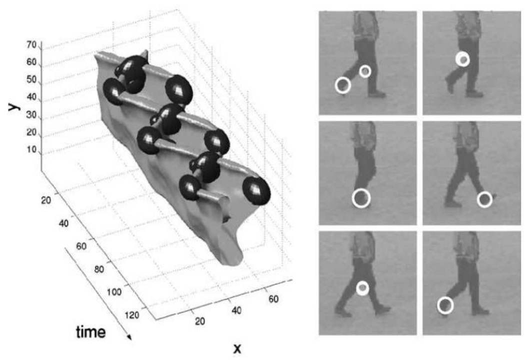
2.1.1 Local action representation

Local representation methods aim at encoding actions using 2D or 3D features extracted from local patches or interest points in the video volume [28]. Common local features are derived from spatiotemporal information, such as the invariant feature transform (SIFT) extended to 3D [63], and trajectories of local body parts.

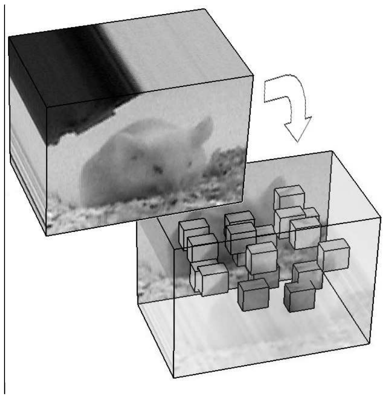
a) Action representation using interest points (IP)

Scovanner *et al.* [33] proposed to generate 3D SIFT features and cluster their values to build a codebook for action classification. In the same vein, methods based on sparse spatiotemporal interest points (STIP) have been developed in [4, 64]. One of the first action recognition methods using features based on local interest points has been proposed by Laptev *et al.* [1]. In their work, they have extended the Harris corner detector [65] to 3D corner detection in order to extract interest points describing significant local spatiotemporal variations. Such variations are estimated by calculating local maxima of the normalized Laplacian operator over spatial and temporal scales. This approach has shown a good success in recognizing simple actions such as people *walking*, *sitting* in non-cluttered scenes (see Fig. 2.1.1(a) for illustration).

Dollar *et al.* [2] observed that the STIP features porposed in [1] are too sparse. To overcome this issue, they propose to use Gabor filters using a Gaussian smoothing kernel separately along spatial and temporal dimensions to extract interest points which are associated to cuboids. A dictionary of cuboid prototypes is constructed for each dataset by clustering cuboid appearances with the K-means algorithm. Using this descriptor, the method was enabled to recognize facial expressions, mice activities and human actions (see Fig.2.1.1(b) for illustration). Niebles *et al.* [66] used a similar approach to detect interest points and generate bag-of-word features after their clustering.



(a)



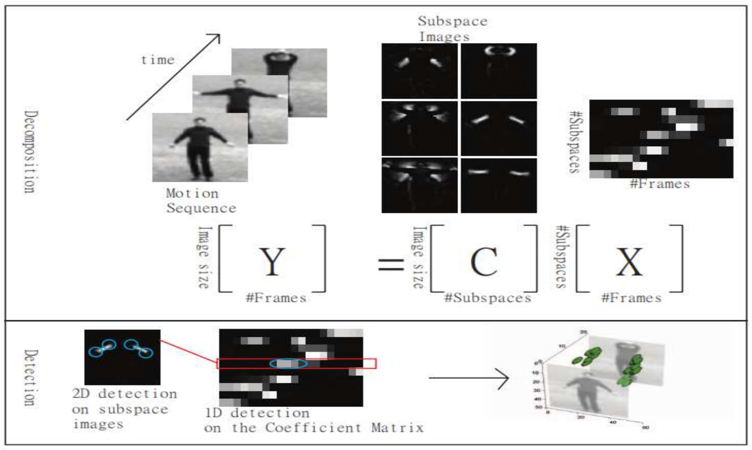
(b)

Figure 2.1.1 – (a) [1] Detected interest points using 3D Harris; left image: 3D plot with a thresholded level surface of a leg pattern (upside down) and the detected points (ellipsoids); right image: interest points overlayed on single frames in the original video sequence, (b) Cuboid Features using Gabor detector [2].

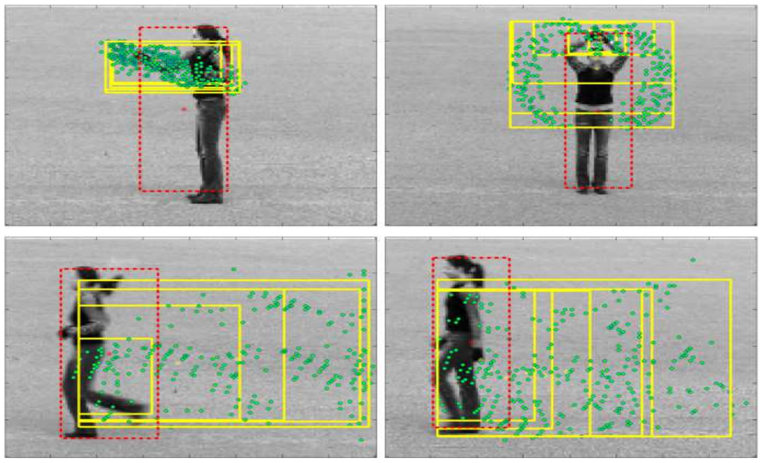
Oikonomopoulos *et al.* [67] proposed to detect salient regions by measuring color variation of pixels neighborhood in both space and time domains. They proceed with the automatic selection of the scale by analyzing entropy as a function of position and scale. On the other words, a region center is deemed salient point at a scale at which the entropy achieve local maxima. Wong *et al.* [3] and Bregonzio *et al.* [4] suggested to detect interest points by using global information of a moving object. They use the whole video to extract dynamic texture and apply non-negative matrix factorization to build a subspace corresponding to moving parts of a video. Interest points are then detected in the selected subspace and their saliency is measured by statistical analysis of motion (see Fig. 2.1.2(a) for illustration). In a similar way, Bregonzio *et al.* [4] propose to detect interest points in a two-steps process. First, they calculate the difference between two consecutive frames to detect moving objects, then, they use 2D Gabor filters at different orientations to detect salient points (Fig. 2.1.2(b) for illustration). Instead of using local 3D cuboid, Willems *et al.* [68] propose to use the video integral [69]. They calculate the 3D Hessian matrix and localize interest points at multiple scales in the spatiotemporal domain.

Dalal *et al.* [35] were the first to propose HOG (histogram of oriented gradients) features for human detection. Since then, several researchers have used the HOG approach to characterize local appearance and motion patterns to characterise actions. For example, Laptev *et al.* [34] have developed a popular descriptor considering the spatiotemporal neighborhood for the extracted interest point. Each volume formed by the selected neighborhood is divided into spatiotemporal cuboids. For each cuboid, HOG and HOF (Histogram of optical flow) descriptors are calculated and their normalized values are concatenated to form a combined descriptor. Kläser *et al.* [42] have proposed an extension of the HOG approach by developing a descriptor based on histogram of 3D gradient orientations. The gradients are computed using the video integral at a given scale by convolution with a box filter. Regular polyhedrons are then used as 3D patches to quantize the orientations. Resulting patches are finally divided into 3D cells over which the histogram is calculated. The final descriptor is formed by concatenating the normalized histograms of all combined cells. Zelnik *et al.* [70] used intensity gradient orientation for action description. They calculate gradients at multiple temporal scales to represent actions by building a temporal pyramid representing the different temporal scales. After calculating intensity gradients at all space-time points (STP), only STPs having a high temporal variance are selected for the final action descriptor.

Despite the effectiveness of methods using interest points to discriminate human actions, they have some major limitations such as sensitivity to instabilities caused by lighting changes, noise, camera jitter and background motion. This sensitivity is caused mainly by the fact that most of the descriptors are based on the gradient information calculated locally. Finally, note that the HOG performance depend on the size of the human body; too small or large silhouettes can yield non-representative features. Also, STP-based methods are computationally intensive with large video sizes.



(a)



(b)

Figure 2.1.2 – (a) An overview of spatiotemporal interest points using global information on single frames in the original video sequence [3], (b) Examples of clouds of interest points as proposed in [4].

b) Action representation using trajectories

Some approaches use directly object trajectories to represent actions [24]. Actions are interpreted as a set of space-time 2D/3D points corresponding to the joint position (same position) of the moving person. Campbel *et al.* [71] proposed to represent human actions as 3D curves modelling body parts trajectories. These curves are projected into multiple 2D subspaces to recognize to build descriptors for action recognition. Sun *et al.* [72] proposed tracking the trajectories of salient SIFT points and model hierarchically the spatiotemporal information of the latter. In this setting, three levels of context were used to model motion patterns for action representation: *point-level context*, *intra-trajectory context* and *inter-trajectory context*. In a similar way, Messing *et al.* [73] proposed to track 3D Harris interest points with Kalman filtering. Extracted trajectories are then represented as sequences of log-polar quantized velocity history.

Wang *et al.*[5] used dense trajectories for action recognition, which are generated by tracking salient points using dense optical flow field. Then, four descriptors are extracted from these trajectories for action recognition, namely: shape, HOG, HOF and motion boundary history (MBH). Haiam *et al.* [74] and Jiang *et al.* [43] used a similar approach to extract trajectories up to a difference in the way the final trajectories are used for action recognition. While previous work use the volume around trajectories to describe the action, [74, 43] combine local and global reference points to characterize actions. Figure 2.1.3 shows an example of dense trajectories. We can note, however, that discontinuity of some trajectories caused by occlusion or background clutter can cause false recognition results [28].

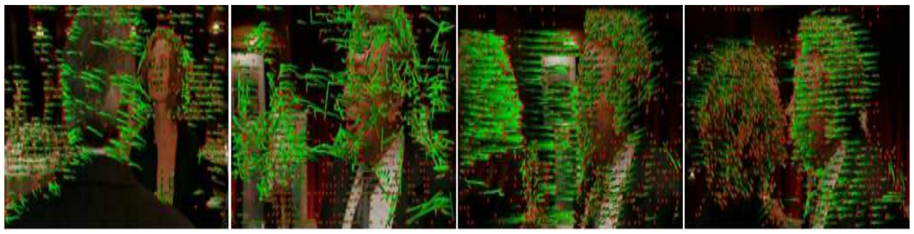


Figure 2.1.3 – Example of generated dense trajectory for 'kiss' action. Red dots indicate the points position in the current frame [5].

2.1.2 Global action representation

Global representation approaches, also called *holistic representation*, use the entire videos or encode a region of interest (ROI) containing the action of a person as a whole. ROIs can be obtained by background subtraction or tracking [24]. Common global approaches are derived from silhouette (shape), edges or optical flow. For instance, Yamato *et al.* [6] used object silhouettes (binary foreground image) to extract mesh feature vectors by counting the number of black and white pixels within the selected region (see Figure 2.1.4(a) for illustration). These features are then classified using hidden Markov Models to recognize simple actions such as *tennis smashing* and *strike*.

One of the most popular methods using object silhouettes has been proposed by Bobick *et al.*[7]. In their work, actions are represented using temporal templates composed of two images: a *motion-energy image* (MEI) and a *motion history image* (MHI). MEI indicates generally the location of the action in the sequence and MHI indicates the history of motion, where recent motion is represented by higher scalar values (brighter) than older one. Action recognition is then performed using (MEI, MHI) matching. Davis [75] has proposed a hierarchical extension to the original MHI to compute local motion patterns. He transformed the MHI into an image pyramid capturing motion speed at different levels using a fixed-size gradient mask. Shao *et al.* [76] extended this approach by building a shape feature generated by combining MEI and MHI with pyramid correlogram of oriented gradients (PCOG) to recognize indoor actions.

Optical flow (OF) is another important feature that has been used to recognize actions by analyzing dominant motion in videos [77]. A pioneering algorithm for action recognition using OF was proposed by Efros *et al.* [8]. They introduced four motion channels by splitting OF of a spatiotemporal volume generated by the motion of a human body. Each channel correspond to the horizontal, vertical, negative and positive component of OF. Figure (2.1.5)

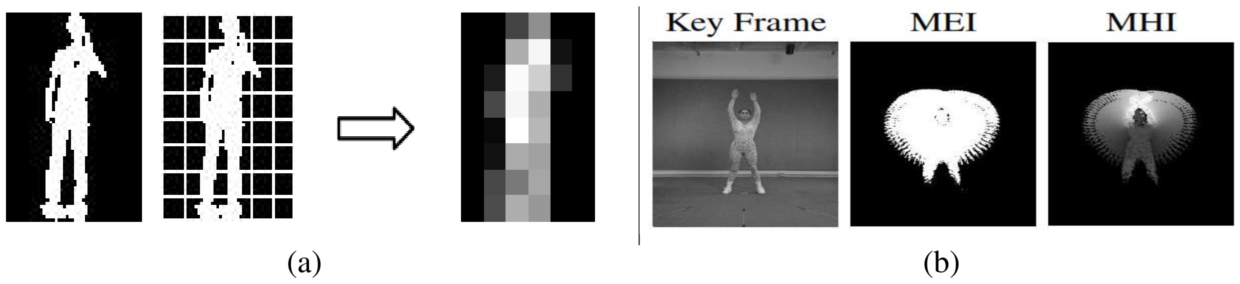


Figure 2.1.4 – Examples of global representation: (a) normalized silhouette and bloc based representation [6], (b) MEI and MHI representation for a sample action [7].

shows examples of OF channels used to generate motion descriptors. This approach has been also used in other works for action representation [78, 79]. Danafar *et al.* [78] applied OF for action recognition in video surveillance. They partitioned the ROI (human silhouette) into three parts: head, body and legs, to capture local motion patterns. Then, the three parts flow histogram are combined in a final representation of action recognition. Ikizler *et al.* [79] used OF and edges in separate histograms to build motion features for action recognition. OF histograms are formed by binning and matching dense blocks over successive frames, whereas shape information is represented by the histogram of edge orientation.

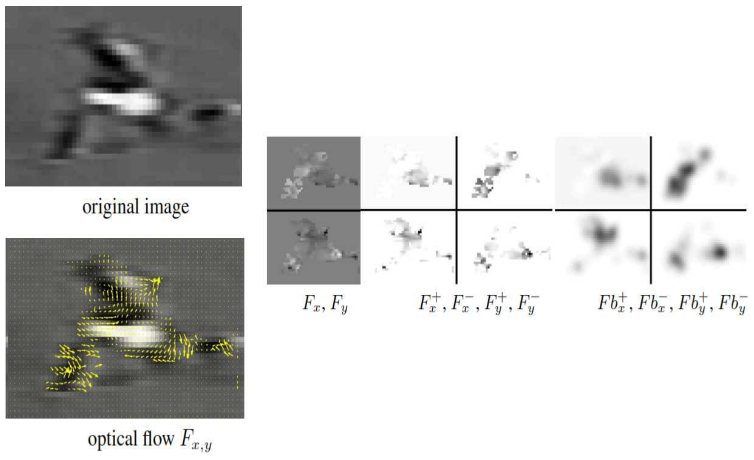

original image
optical flow $F_{x,y}$
F_x, F_y
$F_x^+, F_x^-, F_y^+, F_y^-$
$Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^-$

Figure 2.1.5 – Representation of optical flow channels and their blurred components [8].

Global representation of human action can be also built using object contours. This can be achieved, for example, by analyzing the spatial distribution of boundary points surrounding objects. To describe object silhouettes, Kholgade *et al.* [80] used chain code analysis to build a descriptor for action recognition. Another popular approach describing object contours is the *shape context* (SC) method [16]. This method uses 2D shape context of points described by the log-polar distance histograms of bin points. For each reference point, a histogram is generated by counting the number of points that fall within a certain distance and orientation from a reference point [36]. Grundmann *et al.* [81] have generalized the 2D shape context to a 3D version to recognize spatial and temporal details of human silhouettes. An action is represented by a 3D point cloud extracted by applying non-uniform sampling of 2D silhouettes and use leave-one-out classifier to recognize actions. Kholgade *et al.* [82] have proposed a 4D spatiotemporal shape-context descriptor which captures the magnitude and direction of points along the human contour over consecutive frames of a video. The main limitation of SC-based methods is that the SC descriptor is high-dimensional and is computationally intensive since the distance between each pair of boundary points should be calculated.

Action representation can also be achieved by analyzing motion history (MHI) of silhouettes. For instance, MHI has been extended to motion history volumes (MHVs) by Weinland *et al.* [9]. They use visual hulls computed from multi-view sequences instead of the 2D silhouettes (see Figure 2.1.6(a) for illustration). Blank *et al.* [10] have used local space-time measures of shapes generated from silhouettes and apply a simple nearest neighbor classification with the Euclidean distance to recognize actions (see Figure 2.1.6(b) for illustration). Yilmaz *et al.* [83] have proposed an action representation by exploiting contours of extracted silhouettes, which are used to generate spatiotemporal action volumes (STV). They extract the differential geometric proprieties from the STV surface, such as maxima/minima in the space-time domain to describe action attributes. Finally, recent development of human pose estimation algorithms [59, 84], mainly using Microsoft Kinect, provide RGB and depth maps enabling skeleton joints extraction for action recognition [85, 86, 87]. In [84], human pose is estimated by joint location and quantization of features extracted from human body parts, which are organized part-sets histograms for action representation. In the same vein, [88] propose to represent actions by extracting features in coarse-, mid-, and fine-level scales, and use SVM for action classification.

Generally, using action representation based on shape and edge features require robust background subtraction techniques to dissociate human silhouettes from the background. Therefore, they can be applied only for videos acquired by static cameras. However, imperfections caused by background subtraction, due to background motion, noise, etc., can decrease the accuracy of action representation. On the other hand, human pose estimation can be sensitive to video quality where ambiguous body joints can be detected in the presence of occlusion or background clutter. Finally, motion information can be used to represent actions in videos acquired by moving cameras. More generally, object detection and tracking techniques can be applied in these videos to extract object attributes like pose, orientation and trajectories, which can be used for action description [49].

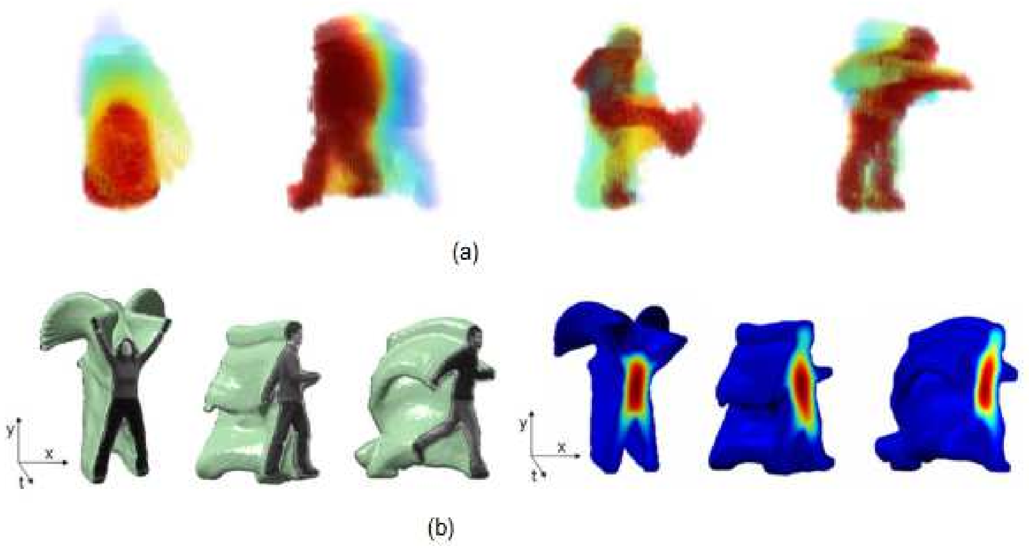


Figure 2.1.6 – Examples of global representation: (a) motion history volume (MHV)[9], (b) space time shapes [10].

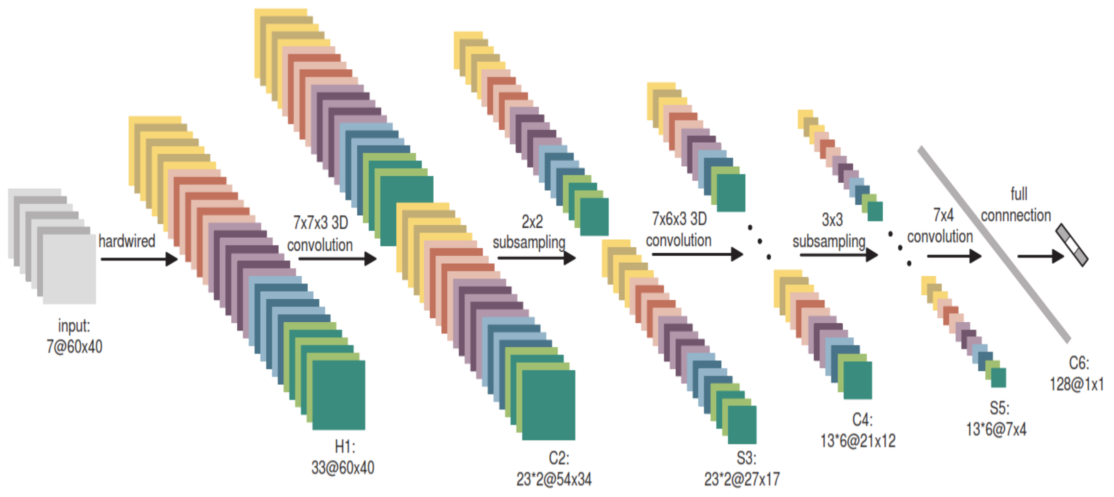
2.1.3 Deep learning for action representation and recognition

Over the last years, deep learning-based approaches have become very popular for human action recognition in videos. Deep learning aims at learning representations (features) automatically from multiple layers hierarchically and build a high-level representation of the raw inputs [89]. Deep convolutional neural networks (CNNs) are the most successful methods for learning high levels of abstraction representations in image/video data [90, 91]. CNNs have achieved very good performance in recognition problems such as objects, faces and scenes in still images [92, 93]. Recently, using deep representation for action recognition has been investigated by several researchers where some success has been achieved. A review on proposed methods in action classification using CNNs can be found in [94, 89].

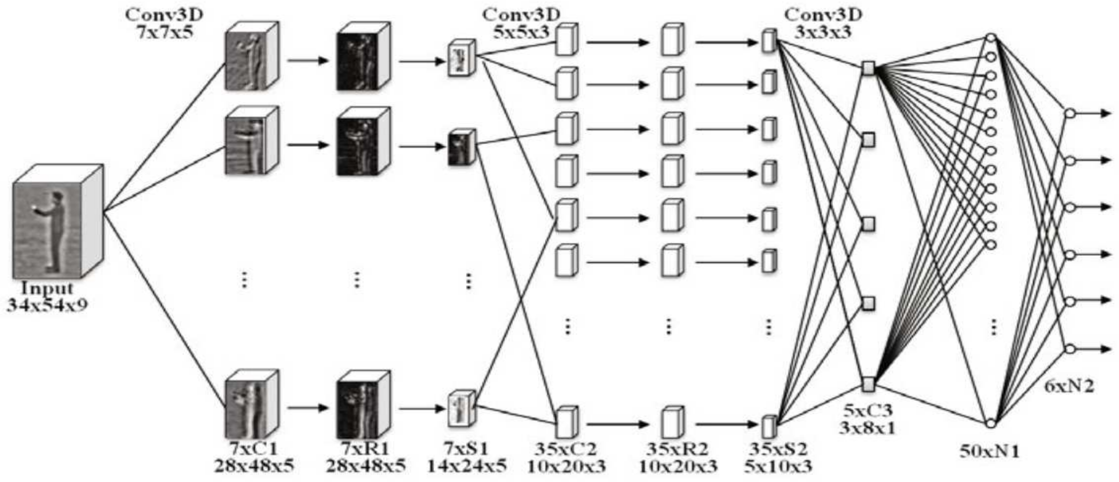
Several methods have been proposed to extend the 2D CNN architecture to a 3D CNN one for human activity recognition. Ji *et al.* [11] propose an extension of a 2D CNN to a 3D to capture discriminative spatio-temporal features. Their CNN model generate multiple channels information from adjacent video frames using hardwired kernels then performs 3D convolution in each channel. The final feature representation is obtained by combining information from all channels and one versus all linear SVM is applied to classify actions. Baccouche *et al.* [12] extracted spatio-temporal features by extending a 2D-ConvNets into a 3D architecture. Similar to [11] they also use 3D convolutions but directly in raw input videos (see Figure 2.1.7), extracted features are then trained using recurrent neural networks (RNN) to recognize actions. Tran *et al.* [95] propose a 3D CNN which capture the spatio-temporal features at large scale video by using a homogeneous architecture with a fixed size convolution kernels at all layers and learned final representation using multi-class linear SVM. Liu *et al.* [96] combined high-level features learned from depth sequences and taking into account skeleton joints position and angle for recognizing actions using SVM.

Varol *et al.* [97] learned video representation using long-term-temporal convolutional networks (LTC-CNN) for different temporal and spatial resolution with different input data (RGB and optical flow). Wang *et al.* [98] proposed a trajectory-pooled deep-convolutional descriptor (TDD) by combining the benefit of handcrafted features (improved trajectories) and deep learned features using 2 stream ConvNets: spatial nets trained in a single frame to capture appearance and temporal nets in a volume to capture motion. Ravanbakhsh *et al.* [99] use the pyramid architecture to extract CNN features which capture a discriminative sub-actions from a video snippet. Key frames are first identified by tracking spatial changes in CNN feature space and then pyramid CNN flow is constructed from a snippet in hierarchical way to represent action, the final action representations are classified using kernel SVM. Banerjee and Murino [100] combined coarse and fine representations extracted respectively form the whole video and a set of snippets using CNN and apply max-pooling to the difference vectors of consecutive frames, action recognition is done using one against all SVM. Kai *et al.*[101] suggests to add attributes to boost input data and guide representation learning procedure of ConvNets. They use two-stream ConvNets: one in the row video frame to capture spatial appearance and the second stream similar to the spatial one but take frame difference as input.

Charalampous *et al.*[102] use another deep learning paradigm named Hierarchical Temporal Memory (HTM), its architecture comprises identical nodes in a tree-shaped hierarchy. Their model considers the entire sample sequence and extracts the descriptors in a frame by-frame manner, action recognition is achieved using KNN and SVM. Despite the results achieved using deep learning for action recognition, performance of the models depends on the amount of the data and how to use temporal information and how to reduce the number of calculated weights. The most challenging problem in deep learning-based approaches is that they require a huge amount of labelled data for their training. In addition, they do handle efficiently the temporal dimension of data since videos can have varying temporal length.



(a)



(b)

Figure 2.1.7 – Illustration of 3D CNN architecture: (a) 3D CNN for human action recognition with hardwired layers [11], (b) 3D convNets for spatiotemporal feature construction [12].

2.2 Action classification models

When an action representation is available for an observed frame or sequence, human action recognition can be cast as a classification problem [27]. Methods used for classification and recognition for human actions can be divided into two categories: *discriminative methods* which learn boundaries separating two or more classes, and *generative methods* that learn the statistical distribution for action which generate the observable data [103].

1. **Discriminative models**: The commonly used algorithm in action recognition is the Support Vector Machines (SVM) classifier [104]. The SVM method constructs an hyperplane maximizing the margin (largest separation) between two classes. The nearest data points to the hyperplane are called support vectors [104]. Schuldt *et al.* [32] proposed to use local spatiotemporal features in videos with SVM classifier to recognize human actions. Laptev *et al.* [34] porposed to classify actions in movies using non-linear SVM with multi-channel Gaussian kernel. Moreover, Ikizler *et al.* [79] proposed to use kernel SVM separately on shape and motion descriptors and combine obtained results to recognize actions. The main drawback of SVM is its inability to cope naturally with multi-class problems, where usually a one-versus-all strategy is used. This requires, however, a high computation time required for training when increasing the size of data and the number of classes.

   Another simpler classifier used in action recognition is the K-nearest neighbor (KNN) which uses the distance between a given data vector and it nearest neighbors to assign to it the most represented class label [105]. Efros *et al.* [8] and Blank *et al.* [106] have demonstrated the effectiveness of their descriptors using the Euclidean distance as similarity measure in the KNN. Kumari *et al.* [22] proposed to represent actions with the information about silhouettes in the spatial domain using the discrete Fourier transform (DFT). They use the resulting feature vectors as input for KNN-based classification. The main limitation of the KNN classier is its drop in inefficiency as the size of data and dimensionality increases. Moreover, it is sensitive to the choice of the parameter K.

   Classifiers based on artificial learning such as artificial neural networks (ANN) are also used to recognize human actions. ANN use the concept of biological neural networks to learn nonlinear input-output relationship using three types of layers: input, output and hidden layers [107]. Faroughi *et al.* [108] proposed to use a multilayer neural network (MLNN) jointly with motion and eigen-space technique to detect falling persons. Fiaz and Ijaz [109] proposed a method to detect and track suspicious activity in a surveillance environment using ANN. Moreover, Iosifidis *et al.* [110] proposed to use MLNN to recognize actions in a multi-view setting. Despite their ability to describe the relationship between data, ANN require high computation when using huge amount of data. Furthermore, they are not always guaranteed to converge when learning the parameters.

Boosting algorithms [111] such as AdaBoost (adaptive boosting) [31] classifiers are also used for human action recognition. Boosting algorithms create strong classifier by learning iteratively weak classifiers. The final classifier is the combination of these weak classifiers with their weights which correspond to their corresponding accuracy [111]. Fathi and Mori [31] proposed to use the AdaBoost classifier in two steps for action recognition. They first step uses Adaboost to combine low-level features to create informative mid-level features, and then uses, for the second time, adaBoost to train the final classifier from the mid-level features to choose the best subset of them for action classification.

2. **Generative models**: Given a sequential and temporal nature of actions, models such as hidden Markov models (HMM) [112] and dynamic Bayesian networks (DBN) [113] are used to model human action. HMMs were proposed for the first time by Yamato *et al.* [6] to recognize tennis actions. They trained HMMs using feature vectors extracted from silhouette (considered as sequence of observations). Trained HMMs are used for action recognition, where the model having the best match with a new input action video is chosen as the recognition category. HMMs are also used to recognize hand gestures by Bobick and Wilson [21] and the American sign language (ASL) by Starner and Pentland [114].

   Several variants of HMM were proposed to deal with action recognition. Thuc *et al.* [115] proposed to use cyclic HMM to model the quasi-periodic cycles of body movement. DBN is a generalization of HMMs which can provide detailed description of the characteristics of human motion. DBNs are mostly used to recognize human interactions such as proposed by Park et al. [116]. They constructed a tree-structured DBN to take advantage of the dependent nature of body parts motion. The tree is composed of two levels: a low-level one to estimate poses of simultaneously tracked body parts and a high-level one to estimate the overall body pose.

Recently, Singh *et al.* [117] proposed to use dynamic Bayesian action network (DBAN) to recognize single human action combined with 2D part models. An action is first decomposed into simple primitive actions (gestures) by projecting the 3D pose into 2D to determine the visible body parts. Each primitive is associated to a duration in term of number of frames. A 2D part-based model is used to align the 2D object pose. The states of DBAN are composed of three principal layers: action, primitives (gestures) and 2D part model and one intermediate layer which is the primitive duration. Despite the ability of HMMs and its variants and DBN to model human action, they still have the main drawback of requiring long training sequences for their training. In addition, they are rigid to account for variability within each action category.

2.3 Two-person interaction recognition

Daily human activities do not consist only in performing single actions, people are constantly interacting with each other or with objects. Recently, several works have been proposed for group activity recognition, where individual actions and interactions between multiple persons and their environment are studied [118, 119]. Compared to the sheer number of methods proposed for single human action recognition, group activity recognition is less investigated given the complexity of modelling human interactions. Among group activity recognition problems, two-person interaction has been one of the main problem studied [26]. Several works have been proposed in the last few years for extending methods of single action recognition to inter-person activity recognition .

Ryoo *et al* [26] designed a method to measure structural similarity between sets of spatiotemporal features extracted from two videos. The authors then derived a kernel for interaction classification based on support vector machines (SVM). In the same vein, Slimani *et al.* [120] exploited spatial and temporal correlation between 3D-SIFT features extracted from videos to classify activities based on the KNN. Sefidgar *et al.* [121] proposed to select a subset of frames capturing discriminative information from the video sequences, then encode relative spatiotemporal information of each subset using HOG. The classification of interactions is performed using linear SVM. To recognize simple two-person interactions, Sener *et al.* [122] proposed to extract multiple visual features (e.g. HOG and HOF) from each person regions (face and body) and combine them using multiple instance learning (MIL) to construct descriptors and SVM for classification. Cho *et al.* [13] proposed to describe interaction in term of individual, local and global motion. Descriptors are constructed by calculating the relationship between detected interest points and three reference points (individual centroid, local centroid and global centroid), then applying SVM for interaction recognition (see Figure 2.3.1 for illustration).

Trajectories are also used to model human interactions. One of the earliest works using trajectories has been proposed to recognize violent interactions (aggressions) [123]. In [124], group trajectories of densely sampled key points are used to describe interactions as activity components related to body parts motion. The interactions between the different components are modeled through a spatiotemporal context kernel plugged in an SVM classifier. Similarly, Zhang *et al.* [125] grouped extracted dense trajectories on local motion patterns using a filtering algorithm, then *multiple instance learning* (MIL) is used to classify interactions. Finally, trajectories are used in multiple cameras setting to recognize human interaction by coupling body motion and proximity of each individual, then fusing information extracted of all views [126].

Human pose is another feature used for extracting descriptors for interaction recognition. Park *et al.* [127] proposed a hierarchical Bayesian network (HBN) for interaction recognition,

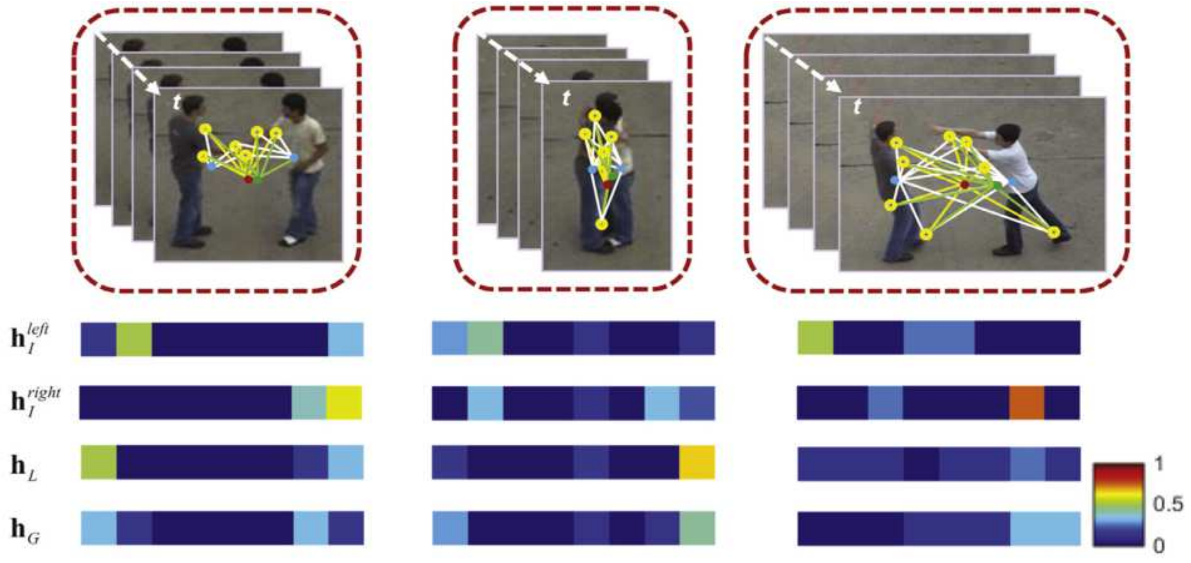


Figure 2.3.1 – Example of three movement CONtexts (MOCONs) extracted from three different interactions: hand-shake (left), hug (middle), and push (right). h_I^{left} and h_I^{right} capture the local movements of individuals, h_L capture locai interaction of people and h_G capture global interaction [13].

where low- and high- level nodes are used to represent body parts and overall body pose, respectively. Zhan and Chang [128] estimate pose by integrating a *pictorial structures model* (PSM) with a motion mask, then use hierarchical HMM to recognize human interactions. Meng *et al.* [129] used skeleton joints position for model human pose estimation, which are combined with appearance features for interaction description and classification using SVM. Gemeren *et al.* [14] (see Figure 2.3.2) used PSM to localize motion around joints of interest during interactions, then localized area is encoded using HOG and HOF for interaction representation and recognition using SVM.

Deep learning has been used recently for recognizing two-persons interaction in video sequences. For instance, Liang *et al.* [15] propose to represent interaction using high-level descriptors in a two-layer structure: spatiotemporal local trajectories and context features (see Figure 2.3.3 for illustration). The two representations are combined in SVM-based classicisation. Berlin et al. [130] considered videos with static background to extract regions of interest

(ROI) to detect key points using Harris corner detector. Extracted features are then fed as input to a deep learning network (DLN) to achieve dimensionality reduction and contrectation classification. Zhao *et al.* [131] proposed to use optical flow in each frame of the video as an input to residual network and resulted temporal features are classified into interactions using *long short-term memory* (LSTM) network.

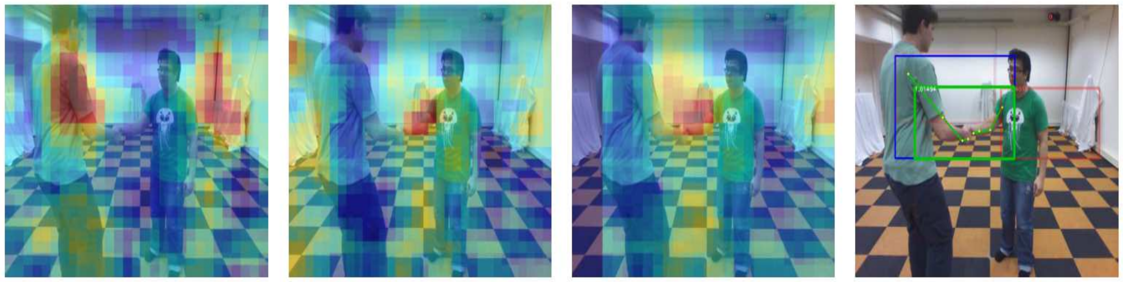


Figure 2.3.2 – Detection scores for two single person models (first and second image), the cumulative score (third image) and the final detection result (fourth image) [14].

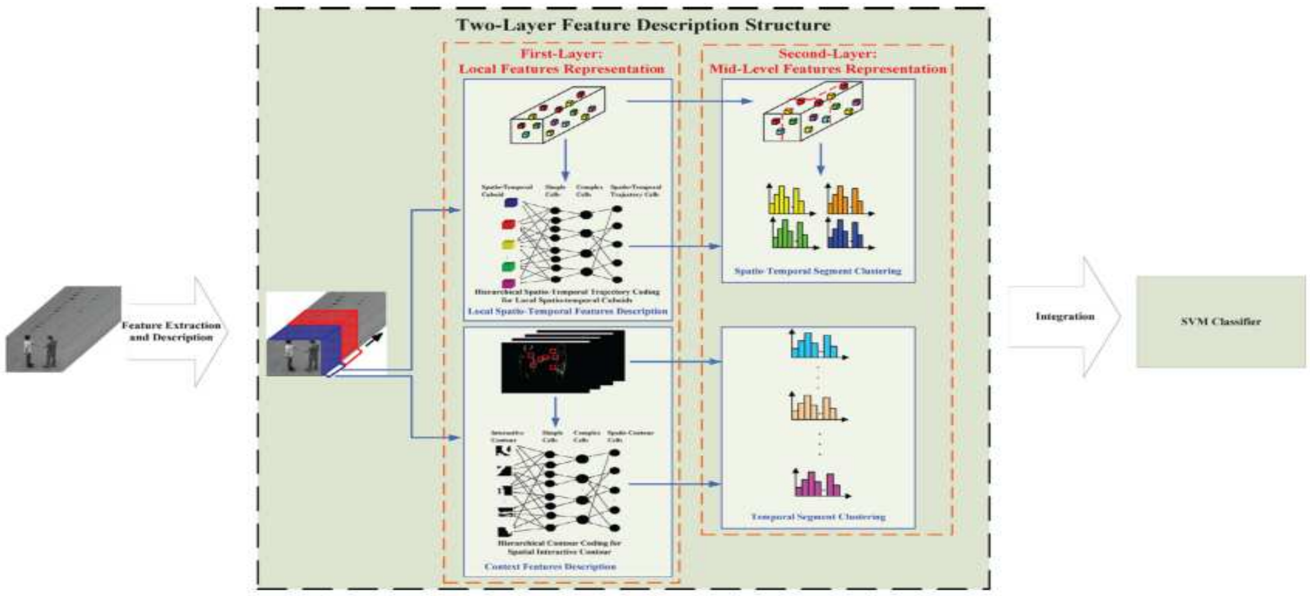

Feature Extraction and Description
Two-Layer Feature Description Structure
First-Layer: Local Features Representation
Second-Layer: Mid-Level Features Representation
Spatio-Temporal Cuboid
Simple Cells
Complex Cells
Spatio-Temporal Trajectory Cells
Hierarchical Spatio-Temporal Trajectory Coding for Local Spatio-temporal Cuboids
Local Spatio-Temporal Features Description
Spatio-Temporal Segment Clustering
Interactive Contour
Simple Cells
Complex Cells
Spatio-Contour Cells
Hierarchical Contour Coding for Spatial Interactive Contour
Context Features Description
Temporal Segment Clustering
Integration
SVM Classifier

Figure 2.3.3 – Two-layer structure for local trajectories and context features extraction of interactive activities [15].

2.4 Summary and comparison

We have provided an overview of state-of-art methods dealing action recognition involving single person or interactions between two persons. In a nutshell, action recognition systems are composed of two main parts: 1) action representation and 2) action classification. For action representation, several methods have been proposed to extract compact descriptors from raw video data, where mainly two types of representation exists: local representations (e.g., IP, HOG, trajectories) [33, 64, 73] and global representations (e.g., silhouette, optical flow, contour) [10, 9, 82]. Approaches based on local representations have the advantage of being invariant to view-point and appearance changes. However, they lack holistic description of body actions and are usually computationally intensive with large video sizes. Approaches based on global representations enable to describe the whole body action and provide detailed information about deformable objects. They are also less sensitive to noise. However, they require good algorithms to isolate object silhouettes, which can be reliably performed only on videos without background motion, captured by static cameras. Most of the above approaches are sensitive to intra-class variation since descriptors extracted, in dense fashion from the video,can vary drastically between different subjects performing the same action.

Statistical models for action recognition are usually based on supervised learning techniques, where labeled data are used to train classification models which are later used to predict action categories of newly-observed videos. Classification methods can be either discriminative (e.g., SVM, KNN, LR) or generative (e.g., HMM, DBN, SCFG). One must note, however, that most of used classifiers are more adapted to binary classification. Their performance decreases with increasing the number of classes and the dimensionality of data. On the other hand, feature vectors used for action representation are systematically extracted from raw video data and may contain noise and redundant information which can prevent classifiers from discriminating action categories. To mitigate this problem, an intermediate step is usually added to reduce the dimensionality of feature vectors by techniques such as PCA [44], LLE [47] and Isomap

[46]. However, in addition to increase the computation time, these techniques do not take into account discriminative information about classes, and can even destroy it through the reduction process [132]. This in turn can prevent any classifier from achieving optimal performance for action recognition [133]. Table 2.4.1 summarises the main existing approaches for single person action and two-persons interaction recognition (AR and IR) in the recent literature. We highlight also the type of information used to represent actions (local or global), as well as the classifiers used to discriminate between actions and whether a dimensionality reduction step is added.

To boost the performance of classifiers, some approaches have been proposed to directly integrate feature relevance in the classification models for action recognition. For example, Bregonzio *et al.* [134] have used the AdaBoost cascade classifiers for action recognition. Moreno *et al.* [135] have used the binary L_2 boost to select good linear models for reducing classification error. Zheng *et al.* [136] have used an entropy-based sub-modular function to select the best attributes for separating classes, which are then used in an SVM classifier to recognize actions. These methods are generally computationally efficient, but are not easy to generalize to multi-class problems since the build-in architecture are more adapted to binary classification [137]. Usually, these methods use the *one-vs-all* strategy to extent their approach to multi-class problem. This heuristic involves training a single classifier per class, with the samples of that class as positive samples and all other samples as negatives. It requires the base classifiers to produce a real-valued confidence score for its decision, rather than just a class label. This approach can suffer from multiple problems such as unbalanced class distributions since a classier will face a set of negatives that is much larger than the set of positives.

Table 2.4.1 – State-of -the-arts methods characteristics. For each method we mention what kind of representation and classification are used and if dimensionality reduction is applied.

Approach	References	Type of activities	Methods	Local	Global	Dimensionality reduction	Limitations
	[33]		3D SIFT + SVM	✓	×	✓	-
	[1]		Sparse STIP + KNN	✓	×	✓	-
	[2]		STIP cuboid + NN	✓	×	✓	-
	[66]		Cuboid + pLSA	✓	×	✓	-
	[67]	AR	Salient STIP + RVM	✓	×	✓	-
	[3]		moving parts IP+ NN	✓	✓	✓	-
	[4]		Salient IP + NN and KSVM	✓	✓	✓	- Do not capture the whole body action
	[68]		3D IP + KSVM	✓	✓	-	- Do not give details for deformable objects.
Interest points	[34]		HOG and HOF cuboid +KSVM	✓	×	-	- Computationally intensive with large video size.
	[42]		3D patches HOG + KSVM	✓	✓	-	- IP Sensitivity : lighting, noise and camera jitter.
	[70]		STP intensity + Distance matching	✓	✓	-	- HOG depend on human body size.
	[26]		STIP + SVM	✓	-	-	-
	[120]		3D-SIFT + KNN and SVM	✓	-	-	-
	[121]	IR	3D-HOG+ SVM	✓	✓	-	-
	[122]		HOG and HOF + MIL and SVM	✓	-	-	-
	[13]		STIP + SVM	✓	✓	-	-
	[71]		STP intensity + Distance matching	✓	✓	-	-
	[72]		SIFT trajectories+ MKL	✓	×	-	-
	[73]	AR	3D IP trajectories + NB	✓	×	✓	-
	[5]		Dense trajectories + KSVM	✓	✓	-	- False recognition due to the ambiguous
Trajectories	[74]		Dense trajectories + KSVM	✓	✓	✓	points trajectories caused by occlusion
	[43]		Dense trajectories + linear and kernel SVM	✓	✓	✓	or background clutter.
	[123]		Motion trajectory	-	✓	-	-
	[124]	IR	trajectories component + context kernel SVM	✓	-	-	-
	[125]		Dense trajectories + MIL	✓	-	-	-
	[6]		Binary silouhette + HMM	×	✓	-	-
	[7]		Binary silhouette + template matching	×	✓	-	-
	[75]		Hierarchical MHI + statistical matching	✓	✓	✓	-
Silhouette	[76]	AR	Shape(MEI, MHI) + KSVM	×	✓	-	-
	[8]		Optical flow + KNN	×	✓	-	-
	[78]		parts optical flow + SVM	×	✓	-	-
	[79]		Optical flow + edges+ KSVM	×	✓	-	- Require robust background subtraction.
	[10]		Space time shape + KNN	✓	✓	-	- Sensitive to intra-class variation.
	[109]		Silhouette and motion pattern + ANN	×	✓	-	- Limited application to video acquired by static camera.
	[108]		MEI, MHI and ITMI + ANN	×	✓	✓	-
	[80]		Chain code and shape context + ANN and KNN	×	✓	-	-
	[81]		3d shape context + points matching	×	✓	-	-
Contour	[82]		4D shape context + KNN	×	✓	-	-
	[36]	AR	silhouette + KSVM	×	✓	-	-
	[83]		STV + distance and epipolar geometry	×	✓	-	-

Approach	References	Type of activities	Methods	Local	Global	Dimensionality reduction	Limitations
Pose model	[117]	AR	body parts model + DBAN	×	✓	-	-
	[84]		joints locations and body parts model + KSVM	×	✓	-	- Limited application to video acquired by static camera.
	[88]		pose, ST-parts and parts + SVM	✓	✓	-	
	[127]	IR	Body parts + Hierarchical BN	-	✓	-	- Sensitivity to video quality.
	[128]		PSM and motion mask + HHMM	-	✓	-	Ambiguous joints detection caused by occlusion or background clutter.
	[129]		Human joints + SVM	-	✓	-	
	[14]		PSM + SVM	-	✓	-	-
deep learning	[11]	AR	3D CNN + SVM	-	-	-	-
	[12]		3D CNN+ RNN	-		-	-
	[95]		3D CNN+ SVM	-	-	-	-
	[96]		3D CNN+handcrafted+ SVM	-	-	-	-
	[97]		RGB and Optical flow + LTC-CNN	-	-	-	-Require large amount of video data to learn representation.
	[98]		trajectories+ 2D CNN and 3D CNN	-	-	-	
	[99]		2D CNN + SVM	-	-	-	- Boosting performance by handcrafted representation.
	[101]		two-stream Convnets(2D) + SVM	-	-	-	
	[102]		HTM (2D)+ KNN and SVM	-	-	-	-
	[15]	IR	Two-layer structure(trajectories and context features) + SVM	-	-	-	-
	[130]		IP + DLN	-	-	-	-
	[131]		Residual Network + LSTM	-	-	-	-

Chapter 3

Embedded feature relevance for multinomial kernel logistic regression

3.1 Introduction

Based on the presented review of the literature, we can conclude that achieving good success in action recognition requires adequate integration of action representation and classification steps. In most of past work, actions are usually represented using high-dimensional feature vectors extracted from raw video content, and action discrimination is performed using classification techniques such as SVM, KNN, etc. Given the sensitivity of these techniques to noise and outlying data, dimensionality reduction is often performed at an intermediate step. However, having this step operated independently from the classification process constitutes a drawback since it can destroy useful information for action discrimination [132]. In addition, dimensionality reduction methods incur additional computation time, especially with large volumes of data.

To overcome the shortcomings of dimensionality reduction algorithms, which operate independently from classifiers, another type of methods, coined *embedded methods*, has emerged in the literature with the aim to integrate feature selection directly in the classification learning process. These methods integrate directly feature selection (FS) in the classifiers which achieves model fitting and feature selection altogether to produce the so-called "sparse models" for classification [138, 139, 140]. Sparsity aims at assigning weights reflecting the contribution of all features in the classifiers in such a way that relevant features receive higher weights while non-relevant ones are inhibited. These models are also more computationally efficient than using separate steps for feature selection [141]. Sparse models can be roughly categorized into three main groups [142]:

1. *Forward-backward methods*: iteratively add/remove variables according to specific criteria such as the rate of change of an objective function [132, 143, 140, 144] or the sensitivity to the leave-one-out error [145].

2. *Scaling factor methods*: use hyper-parameters that are adjusted by model selection [146, 147] or by minimizing a generalization error bound [148].

3. *Direct optimization methods*: incorporate sparsity promoting terms based on the ℓ_0 and ℓ_1 norms that cause irrelevant feature weights to vanish [137]. Among methods in this category we can find the ℓ_1-regularized SVM [144, 148] and sparse multinomial logistic regression (SMLR) [149] proposed for linear and kernel-based classification.

In many domains, the number of features/predictors is getting larger than the number of sample observations (e.g., text, video) [150]. Optimization methods incorporating appropriate regularization terms are the most used approaches to produce sparse models and overcome overfitting. Regularized SVM [144] and penalized regression [52, 53] are the most commonly proposed methods implementing sparsity. However, SVM is designed for binary classification and is not easily generalizable to multi-class problems, especially when class boundaries are non-linear [137]. Furthermore, the optimization of SVM is sometimes hard to achieve when data are high-dimensional and classes are highly overlapping.

Logistic regression (LR) is a widely used classification method in data mining problems, especially in biomedical and epidemiological studies [137, 151]. Like support vector machines (SVM) [104], LR is known for its good performance for binary classification [137]. However, contrarily to LR, the extension of SVM to multi-class classification was carried out using one-versus-all [152], one-versus-one [153] and directed acyclic graph SVM (DAGSVM) [154] which bring the classification problem back to the binary case. This is partly because LR brings a natural notion of label probabilities which SVMs miss in its formulation [137]. Recently, the kernel trick used to make SVM applicable for non-linear classification problems has been successfully used for LR to build kernel logistic regression (KLR) [155]. The probabilistic formulation of LR and its kernelized version, KLR, makes it easy to extend to a Bayesian framework [155].

Recently, several methods have been proposed to build sparse models based on LR, where feature selection is embedded directly in the LR formulation. For example, regularized versions for logistic regression such as the *least absolute shrinkage and selection operator* (LASSO) [52] and its generalized version [51] have been proposed for enhancing LR by selecting the most relevant features for regression and classification. Group sparsity has been proposed as an extension to the LASSO method [54, 53]. Contrarily to LASSO which performs feature selection for individual features, group LASSO performs selection for entire groups of variables, where each group constitutes a separate explanatory factor. It remains, however, that most of proposed methods are limited to binary classification and linearly-sparable classes.

To deal with the multi-class problems, as is the case for video action recognition, we propose two new models for multi-class kernel logistic regression coined FR-MKLR and GFR-MKLR, respectively. Our models directly embed feature relevance in the kernel function of the MKLR with an appropriate regularization to reduce the effect of non-desirable features for classification. In addition to leading to sparse models for classification, they are easily extendable to multi-class problems where feature relevance is determined simultaneously for each class.

Before presenting the details of our models, we give an overview of the basic formulation of logistic regression and some other classification methods used in our later comparative study, namely kernel SVM (KSVM), sparse multinomial linear logistic regression (SMLR), LASSO and naive Bayes. Basic formulation of the multinomial kernel logistic regression (MKLR) will be presented as well, followed by the proposed models: FR-MKLR and GFR-MKLR.

3.2 Background

3.2.1 Naive Bayes (NB)

The naive Bayes classifier [156] is a simple probabilistic classifier based on the Bayes theorem. It assumes that predictive features are conditionally independent given the class. The Bayes rule used to compute the probability of each class label y given the vector of input features is:

$$p(y|\mathbf{x}_i) = \frac{p(y)p(\mathbf{x}_i|y)}{p(\mathbf{x}_i)}. \tag{3.2.1}$$

Given the assumption that each feature is independent of the others, given a class label y, we have:

$$p(y|\mathbf{x}_i) = \frac{p(y)\prod_{k=1}^{d} p(x_{ik}|y)}{p(\mathbf{x}_i)}. \tag{3.2.2}$$

The naive Bayes classifier which assign the class label y is then constructed using the maximum a posteriori (MAP) decision rule as follows:

$$y^* = \arg\max_y p(y)\prod_{k=1}^{d} p(x_{ik}|y), \tag{3.2.3}$$

with $y \in \{1, ..., m\}$.

3.2.2 Kernel support vector machines (KSVM)

The support vector machines (SVM) method is one of the most popular supervised binary classifiers in machine learning [157, 104, 158]. It was designed to find an optimal separating hyperplane with maximum margin between two classes (largest margin), and the data points in the margin delimiting hyperplanes are called *support vectors*. The SVM can be used to perform linear or non-linear classification using the kernel trick [159, 157]. The decision function for a kernel SVM is obtained by solving the dual optimization problem 3.2.4:

$$\max_{\alpha} \mathbf{W}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,l=1}^{n} \alpha_i \, \alpha_l \, y_i \, y_l \, \mathcal{K}(\mathbf{x}_i, \mathbf{x}_l), \tag{3.2.4}$$

where $\mathcal{K}$ is a non-linear function. The minimization of the function is subject to $\alpha_i \geq 0$, and $\sum_{i=1}^{n} \alpha_i \, y_i = 0$, and the hyperplane decision function can be written as:

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{n} y_i \, \alpha_i \, \mathcal{K}(\mathbf{x}, \mathbf{x}_i) + \beta_0\right). \tag{3.2.5}$$

For multi-class problems, many methods were suggested [160, 161, 162], among those methods we use one-versus-all KSVM in our experiments.

3.2.3 Logistic regression

Logistic regression (LR) is a discriminative classifier that models the class probabilities as a function of the linear combination of independent variables/predictors [163]. One advantage of LR is that it makes no assumption about the distribution of the independent variables as in the case of the Bayes classifier, for example [164]. In our case, the independent variables correspond to motion pattern features extracted form video sequences, whereas the dependent (class) variables are the actions categories. We note that LR has been successfully applied in the past for text categorization where the number of independent variables is large and usually exceeds the number of observations [165].

Let us have a set of n instances of training data $\mathbf{x}_i \in \mathbb{R}^d$, $i \in \{1, ..., n\}$, with d measured predictors, and a binary classification problem (positive and negative classes). To designate the class labels of data, we use a vector of outputs $\mathbf{y} = (y_1, ..., y_n)^T$, where $y_i = 1$ if the ith instance $\mathbf{x}_i$ belongs to the positive class and $y_i = 0$, otherwise. The conditional probability $p(y_i|\mathbf{x}_i)$ is modeled by a sigmoid function as follow $p(y_i = 1|\mathbf{x}_i) = 1/[1 + \exp(f(\mathbf{x}_i))]$ and $p(y_i = 0|\mathbf{x}_i) = \exp(f(\mathbf{x}_i))/[1 + \exp(f(\mathbf{x}_i))]$, where $f(\mathbf{x}_i)$ is linear function of the form:

$$f(\mathbf{x}_i) = \beta_0 + \sum_{k=1}^{d} \beta_k x_{ik} = \beta^T x_i, \tag{3.2.6}$$

The coefficients $\beta = (\beta_0, \beta_0, ..., \beta_d)$ of the logistic regression must be estimated from the training data $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$. This can be achieved using the maximum-likelihood estimation (MLE) method. The intuition of the MLE in logistic regression is that a search procedure seeks values for the coefficients β that minimize the error in the probabilities predicted by the model to those of the data (e.g. probability of 1 if the data is in the right class and 0 otherwise). This can be achieved by minimizing the negative log-likelihood (NLL) which is defined as follows [166]:

$$\mathcal{L}(\beta) = \sum_{i=1}^{n} y_i(\beta_0 + \sum_{k=1}^{d} \beta_k x_{ik}) - \ln \left[1 + \exp(\beta_0 + \sum_{k=1}^{d} \beta_k x_{ik}) \right] \tag{3.2.7}$$

3.2.4 Least absolute shrinkage and selection operator (LASSO)

LASSO [52] is a popular method for linear regression using ℓ_1-norm penalization to achieve sparse solution. The LASSO performs feature selection by shrinking some coefficients β_k to zero and retains the good features, the parameters $\hat{\beta}$ are estimated by solving the Equation 3.2.9

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{k=1}^{d} \beta_j x_{ik} \right)^2 . \tag{3.2.8}$$

subject to the constraint $\sum_{k=1}^{d} |\beta_k| \leq t$, with $t \geq 0$, this gives the Lagrangian form of the problem as follow:

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{k=1}^{d} \beta_k x_{ik} \right)^2 + \lambda \sum_{k=1}^{d} |\beta_k|. \tag{3.2.9}$$

where λ is the Lagrange multiplier.

3.2.5 Sparse multinomial logistic regression (SMLR)

Krishnapuram *et al.* [149] proposed a fast algorithm combining upper bound optimization and component-wise update procedure for learning sparse multinomial logistic regression using ℓ_1 penalty. The estimation of parameter w are done by the MAP estimate:

$$\hat{w} = \arg\max_{w} \sum_{i=1}^{n} \left(\left[\sum_{k=1}^{d} y_i^{(k)} w^{(k)^T} x_i - \log \sum_{k=1}^{d} \exp w^{(k)^T} x_i \right] + \log p(w) \right) \tag{3.2.10}$$

With, $p(w) \propto \exp(-\lambda \|w\|_1)$. Note that a kernel version using RBF kernel was also developed for SMLR. Note that both algorithms (LASSO and SMLR) use ℓ_1-norm penalization. However, the ℓ_1-norm present the drawback of selecting no more than $\min(n, d)$ features [167]. This may entail the elimination of features important for classification in the case of small database with high-dimensional feature vectors.

3.3 Feature relevance for kernel logistic regression

3.3.1 Multinomial kernel logistic regression

Multinomial kernel logistic regression (MKLR) is a supervised learning method that produces non-linear classification boundaries by transforming an input variable space into another space using a positive-definite kernel $\mathcal{K}(.,.)$. In the past, the relationship between SVM and regularized function estimation in the reproducing kernel Hilbert spaces (RKHS) has been established [159]. By replacing the hinge loss function of SVM with the negative log-likelihood (NLL) of the binomial distribution, the same relation can be established with MKLR [155].

More specifically, let us have n instances of training data $\mathbf{x}_i \in \mathbb{R}^d$, $i \in \{1, ..., n\}$, with d measured features for each instance. We suppose that these data have been generated from m classes ($m \geq 2$). Thus, we associate an encoding vector $\mathbf{y}_i = [y_i^{(1)}, y_i^{(2)}, ..., y_i^{(m)}]^T$ for each data point $\mathbf{x}_i$, such that $y_i^{(j)} = 1$ if $\mathbf{x}_i$ belongs to the class j and $y_i^{(j)} = 0$, otherwise. Here, $[\cdot]^T$ is the vector/matrix transpose operator. For binary classification ($m = 2$), we have $y_i \in \{0, 1\}$ and fitting a decision boundary is equivalent to searching a function f minimizing the NLL [155]:

In the two-class case ($m = 2$ and $y_i \in \{0, 1\}$), fitting a decision boundary is equivalent to searching a function f minimizing the following NLL [155]:

$$\mathcal{L}(f) = -\sum_{i=1}^{n} y_i f(\mathbf{x}_i) + \ln\left[1 + \exp(f(\mathbf{x}_i))\right] + \frac{\lambda}{2} \parallel f \parallel_{\mathcal{H}_K}^2, \tag{3.3.1}$$

where $\mathcal{H}_K$ is the RKHS generated by the kernel $\mathcal{K}(.,.)$ and λ controls the contribution of the regularization term $\parallel f \parallel_{\mathcal{H}_K}^2$ that smoothes f. Note that the NLL (3.3.1) is obtained by setting $p(y_i = 1|\mathbf{x}_i) = 1/[1 + \exp(f(\mathbf{x}_i))]$ and $p(y_i = 0|\mathbf{x}_i) = \exp(f(\mathbf{x}_i))/[1 + \exp(f(\mathbf{x}_i))]$. The optimal solution $f(\mathbf{x})$ for (3.3.1) has the form [168]:

$$f(\mathbf{x}) = \sum_{i=1}^{n} a_i \mathcal{K}(\mathbf{x}, \mathbf{x}_i), \tag{3.3.2}$$

where a_i, $i \in \{1, ..., n\}$, are real-valued coefficients. By defining the two vectors $\mathbf{a} = [a_1, ..., a_n]^T$ and $\mathbf{y} = [y_1, ..., y_n]^T$, and using the formulation (3.3.2), function (3.3.1) can be re-written in a compact form as follows [155]:

$$\mathcal{L}(\mathbf{a}) = -\mathbf{y}^T\mathbf{K}\mathbf{a} + \mathbf{1}^T \ln\left[1 + \exp(\mathbf{K}\mathbf{a})\right] + \frac{\lambda}{2}\mathbf{a}^T\mathbf{K}\mathbf{a}, \tag{3.3.3}$$

where $\mathbf{1} = [1, 1, ..., 1]^T$ is an n-dimensional vector of ones and $\mathbf{K}$ is a kernel matrix of dimension $n \times n$ with entries given by $\mathbf{K}_{r,s} = \mathcal{K}(\mathbf{x}_r, \mathbf{x}_s)$, $r, s \in \{1, ..., n\}$. Also, we have adopted the compact form $\ln[1 + \exp(\mathbf{K}\mathbf{a})]$ for $\left[\ln(1 + \exp(\mathbf{K}(1, .)\mathbf{a})), ..., \ln(1 + \exp(\mathbf{K}(n, .)\mathbf{a}))\right]^T$, with $\mathbf{K}(i, .)$, $i \in \{1, ..., n\}$, designating the ith row of $\mathbf{K}$.

In a multi-class setting ($m > 2$), MKLR gives the posterior probability of class j given an observation $\mathbf{x}_i$ which is written as $p_i^{(j)} = p(y_i^{(j)} = 1|\mathbf{x}_i)$. By defining a separate function $f_j(\mathbf{x})$ for each class j, the posterior probability of class j given $\mathbf{x}_i$ can be written as follows:

$$p(y_i^{(j)} = 1|\mathbf{x}_i) = \frac{\exp(f_j(\mathbf{x}_i))}{\sum_{h=1}^{m} \exp(f_h(\mathbf{x}_i))}, j = 1, ..., m, \tag{3.3.4}$$

where $f_j(\mathbf{x}_i) \in \mathcal{H}_K$ which is defined as:

$$f_j(\mathbf{x}) = \sum_{i=1}^{n} a_{ij}\mathcal{K}(\mathbf{x}, \mathbf{x}_i). \tag{3.3.5}$$

We put the coefficients of each function f_j into a separate vector $\mathbf{a}_j = [a_{1j}, ..., a_{nj}]^T$. Because $\sum_{j=1}^{m} p_i^{(j)} = 1$, we have $p_i^{(m)} = 1 - \sum_{j=1}^{m-1} p_i^{(j)}$. Thus, by setting $\mathbf{a}_m = \mathbf{0}$ as for linear logistic regression [149], only the parameters $\{\mathbf{a}_1, ..., \mathbf{a}_{m-1}\}$ are to be learned. For each data point $\mathbf{x}_i$, we associate a vector containing the class posterior probabilities $\mathbf{p}_i = [p_i^{(1)}, p_i^{(2)}, ..., p_i^{(m-1)}]^T$ defined as follows:

$$p_i^{(j)} = p(y_i^{(j)} = 1|\mathbf{x}_i) \;=\; \frac{\exp(f_j(\mathbf{x}_i))}{1 + \sum_{h=1}^{m-1}\exp(f_h(\mathbf{x}_i))}, j = 1, ..., m-1 \tag{3.3.6}$$

$$p_i^{(m)} = p(y_i^{(m)} = 1|\mathbf{x}_i) \;=\; \frac{1}{1 + \sum_{h=1}^{m-1}\exp(f_h(\mathbf{x}_i))}. \tag{3.3.7}$$

By putting $\mathbf{A} = [\mathbf{a}_1, ..., \mathbf{a}_{m-1}]$, the penalized multi-class NLL is given by (see Chapter 6 section 6.1):

$$\mathcal{L}(\mathbf{A}) \;=\; \sum_{j=1}^{m-1} -\mathbf{y}^{(j)^T}\mathbf{K}\mathbf{a}_j + \mathbf{1}^T \ln\left[1 + \sum_{h=1}^{m-1}\exp(\mathbf{K}\mathbf{a}_h)\right] + \frac{\lambda}{2}\sum_{j=1}^{m-1}\mathbf{a}_j^T\mathbf{K}\mathbf{a}_j, \tag{3.3.8}$$

where $\mathbf{y}^{(j)} = [y_1^{(j)}, y_2^{(j)}, ..., y_n^{(j)}]^T$.

3.3.2 Feature weighting for MKLR

Feature weighting is motivated by the fact that a good combination of features usually leads to better classification than using each feature individually [132]. To deal with a large number of features and improve the predictive performance of MKLR, we propose to directly weight features in the kernel of the radial basis function (RBF) of the MKLR. In other words, we use a weighted distance in the RBF where each feature is scaled according to its relevance to classification. Note that in contrast with [155], where selection of data instances is performed for better classification, our approach aims at achieving sparsity in terms of features. In what follows, we base our analysis on the Gaussian kernel defined as:

$$\mathcal{K}(\mathbf{x}_r, \mathbf{x}_s) = \exp\left(-\|(\mathbf{x}_r - \mathbf{x}_s)\|^2/(2\sigma^2)\right), \tag{3.3.9}$$

with $r, s \in \{1, ..., n\}$ and $\sigma > 0$ controls the width of the kernel. In what follows, we derive the formulation of fr-MKLR for binary classification and then propose a generalization to the multi-class case:

Feature weighting in case $(m = 2)$

We use a weighting vector $\Psi = [\psi_1, ..., \psi_d]^T$ of the same dimension as our feature space and we plug it into the RBF of the MKLR as follows:

$$\tilde{\mathcal{K}}(\mathbf{x}_r, \mathbf{x}_s) \;=\; \exp\left(\frac{-\|(\Psi^T(\mathbf{x}_r - \mathbf{x}_s)\|^2}{2}\right). \tag{3.3.10}$$

Note that in case all entries of Ψ are equal, the RBF is isotropic and the model boils down to the standard MKLR defined by function (3.3.3). If the entries of Ψ are not equal, Eq. (3.3.10) gives an anisotropic kernel which enables expressing the contribution of each feature using a weighted distance. As the weight of a feature decreases to zero, the feature's contribution to the distance calculation, and therefore to classification, will be decreased.

To encourage model sparsity, we add a regularization on the weight vector Ψ to the negative log-likelihood function (3.3.3) using the ℓ_0-"norm" [169]. The ℓ_0-"norm" of Ψ defined as $\|\Psi\|_0$ = card $\{k|\psi_k \neq 0, k = 1, ..., d\}$ which gives the number of non-zero entries of the vector Ψ. Note that, unlike ℓ_q norms with $q > 0$, $\|\cdot\|_0$ is not a strictly-defined norm. Since the ℓ_0-"norm" is not smooth, and therefore not differentiable, it is usually approximated using the following function (see Fig. 3.3.1 for illustration):

$$\|\Psi\|_0 \approx \sum_{k=1}^{d} \left[1 - \exp(-\beta\psi_k)\right], \tag{3.3.11}$$

where β is an approximation parameter that can be chosen experimentally or be tuned in order to increase the performance of the classifier as suggested by Bradely *et al.* [170]. Our aim by using this penalty term is decreasing weights and contribution of noisy features to classification. This can be achieved by substituting the kernel $\tilde{\mathbf{K}}$ to $\mathbf{K}$ in function (3.3.3) and minimizing the following penalized log-likelihood function:

$$\mathcal{L}(\mathbf{a}, \Psi) = -\mathbf{y}^T\tilde{\mathbf{K}}\mathbf{a} + \mathbf{1}^T \ln\left[1 + \exp(\tilde{\mathbf{K}}\mathbf{a})\right] + \frac{\lambda}{2}\mathbf{a}^T\tilde{\mathbf{K}}\mathbf{a} + \mu\sum_{k=1}^{d}\left[1 - \exp(-\beta\psi_k)\right], \tag{3.3.12}$$

where $\tilde{\mathbf{K}}$ is a kernel matrix of dimension $n \times n$ with entries $\tilde{\mathcal{K}}(\mathbf{x}_r, \mathbf{x}_s)$ given by Eq. (3.3.10) and μ is a regularization parameter. The parameter μ allows to control the level of features reduction. When $\mu = 0$, there is no sparsity in the model and its output is close to MKLR, whereas high values of this parameter can lead to eliminate important features, and thus reduce the performance of the classifier. The minimization of the minus log-likelihood function is performed through an iterative process alternating between two steps until convergence. In the first step, we estimate the entries of the vector $\mathbf{a}$ (for binary classification, only one vector $\mathbf{a}$ is estimated). In the second step, for a given solution $\mathbf{a}$, we minimize function (3.3.12) according to the weighting vector Ψ. We use the Newton-Raphson (N-R) method to estimate

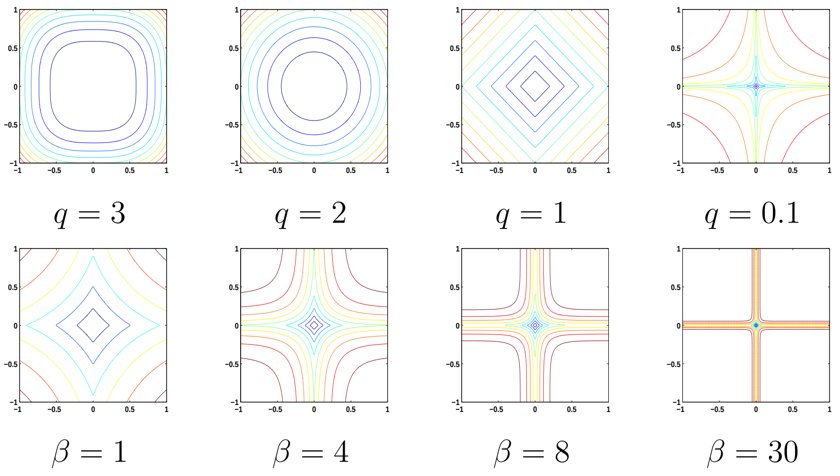


Figure 3.3.1 – Iso-contour plots of: (first row) ℓ_q-norm with different values of q, (second row) ℓ_0-norm approximation using Eq. (3.3.11) with different values of β.

the entries of $\mathbf{a}$ and Ψ. Note that the gradient of (3.3.12) with respect to $\mathbf{a}$ and Ψ are given as follows (see Chapter 6 section 6.2):

$$\begin{aligned} \partial\mathcal{L}/\partial\mathbf{a} &= \tilde{\mathbf{K}}\mathbf{c} && (3.3.13) \\ \partial\mathcal{L}/\partial\Psi &= [\mathbf{c}^T\mathbf{Q}_1\mathbf{a}, ..., \mathbf{c}^T\mathbf{Q}_d\mathbf{a}]^T + \mu\beta\text{exp}(-\beta\Psi), && (3.3.14) \end{aligned}$$

where $\mathbf{c} = \big(-\mathbf{y} + \mathbf{p} + \lambda\mathbf{a}\big)$ and $\mathbf{p} = [p_1, ..., p_n]^T$ where $p_i = p(y_i = 1|\mathbf{x}_i)$. We define the matrix $\mathbf{Q}_k$, $k \in \{1, ..., d\}$, as the following Hadamard product $\mathbf{Q}_k = \tilde{\mathbf{K}} \circ \mathbf{B}_k$, where $\mathbf{B}_k$ is an $n \times n$ dimension matrix with entries defined by $\mathbf{B}_k(r, s) = -\psi_k(x_{r,k} - x_{s,k})^2$, $r, s \in \{1, ..., n\}$. The final gradient vector of (3.3.12) is obtained by concatenating (3.3.13) and (3.3.14) which gives:

$$\tilde{\mathbf{g}} = \left(\begin{array}{c} \tilde{\mathbf{K}}\mathbf{c} \\ \left[\mathbf{c}^T\mathbf{Q}_1\mathbf{a}, ..., \mathbf{c}^T\mathbf{Q}_d\mathbf{a}\right]^T + \mu\beta\text{exp}(-\beta\Psi) \end{array} \right) \qquad (3.3.15)$$

To calculate the Hessian of function (3.3.12), note that the Hessian of (3.3.12) with respect to the entries of $\mathbf{a}$ is $\tilde{\mathbf{K}}\mathbf{W}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}}$, with $\mathbf{W} = \text{diag}[p_1(1-p_1), p_2(1-p_2), ..., p_n(1-p_n)]$. We define also two matrices $\mathbf{M}$ and $\mathbf{\Phi}$ with elements given by $\mathbf{\Phi}_{k\ell} = \frac{\partial^2 \mathcal{L}}{\partial\psi_k \partial\psi_\ell}$ and $\mathbf{M}_{ik} = \frac{\partial^2 \mathcal{L}}{\partial a_i \partial\psi_k}$, $\forall k, \ell \in \{1, ..., d\}$ and $i \in \{1, ..., n\}$. The full Hessian matrix with respect to the parameters $\mathbf{a}$ an $\mathbf{\Psi}$ is given by (see Chapter 6 section 6.3):

$$\tilde{\mathbf{H}} = \begin{pmatrix} \tilde{\mathbf{K}}\mathbf{W}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}} & \mathbf{M} \\ \mathbf{M}^T & \mathbf{\Phi} \end{pmatrix}, \tag{3.3.16}$$

Finally, the N-R update is done using the following iterative scheme [171]:

$$\begin{pmatrix} \mathbf{a}^{t+1} \\ \Psi^{t+1} \end{pmatrix} = \begin{pmatrix} \mathbf{a}^{t} \\ \Psi^{t} \end{pmatrix} - \tilde{\mathbf{H}}^{-1}\tilde{\mathbf{g}} \tag{3.3.17}$$

Feature weighting in case $(m > 2)$

We generalize (3.3.12) to the multi-class case by associating a feature relevance vector $\Psi_j = [\psi_{j1}, \psi_{j2}, ..., \psi_{jd}]^T$ for each class $j \in \{1, ..., m-1\}$. Thus, we associate a separate symmetric kernel $\tilde{\mathbf{K}}_j$ for each class j encoding the class feature relevance. The kernel entries for a class j are calculated as follows:

$$\tilde{\mathcal{K}}_j(\mathbf{x}_r, \mathbf{x}_s) = \exp\left(\frac{-\|(\Psi_j^T(\mathbf{x}_r - \mathbf{x}_s)\|^2}{2}\right). \tag{3.3.18}$$

The new posterior probabilities of the classes given an observation $\mathbf{x}_i$ will be similar to those given in Eq. (3.3.12) by substituting the kernel $\tilde{\mathbf{K}}_j$ to $\mathbf{K}$ for each class j. Using the ℓ_0-"norm" penalization, the new NLL is given as follows:

$$\mathcal{L}(\mathbf{A},\mathbf{\Psi}) = \sum_{j=1}^{m-1} -\mathbf{y}^{(j)^T}\tilde{\mathbf{K}}_j\mathbf{a}_j \quad + \quad \mathbf{1}^T \ln\left[1+\sum_{h=1}^{m-1}\exp(\tilde{\mathbf{K}}_h\mathbf{a}_h)\right] + \sum_{j=1}^{m-1}\left[\frac{\lambda}{2}\mathbf{a}_j^T\tilde{\mathbf{K}}_j\mathbf{a}_j + \mu\sum_{k=1}^{d}\left[1-\exp(-\beta\psi_{jk})\right]\right], \quad (3.3.19)$$

where $\mathbf{A} = [\mathbf{a}_1, ..., \mathbf{a}_{m-1}]$, $\mathbf{\Psi} = [\Psi_1, ..., \Psi_{m-1}]$. Similarly to Eqs. (3.3.13) and (3.3.14), we have $\forall j \in \{1, ..., m-1\}, \forall k \in \{1, ..., d\}$:

$$\partial\mathcal{L}/\partial\mathbf{a}_j = \tilde{\mathbf{K}}_j\mathbf{c}_j \quad (3.3.20)$$

$$\partial\mathcal{L}/\partial\Psi_j = [\mathbf{c}_j^T\mathbf{Q}_{j1}\mathbf{a}_j, ..., \mathbf{c}_j^T\mathbf{Q}_{jd}\mathbf{a}_j]^T + \mu\beta\exp(-\beta\Psi_j), \quad (3.3.21)$$

where we define $\mathbf{c}_j = \big(-\mathbf{y}^{(j)} + \mathbf{p}^{(j)} + \frac{\lambda}{2}\mathbf{a}_j\big)$ and $\mathbf{Q}_{jk} = \tilde{\mathbf{K}}_j \circ \mathbf{B}_{jk}$, with $\mathbf{B}_{jk}$ is an $n \times n$ matrix having entries defined by $\mathbf{B}_{jk}(r,s) = -\psi_{jk}(x_{r,k} - x_{s,k})^2$. It follows that the gradient of $\mathcal{L}$ with respect to the vectors $\mathbf{a}_j$'s and Ψ_j's will be given by:

$$\tilde{\mathbf{g}} = \begin{pmatrix} \tilde{\mathbf{K}}^*(\tilde{\mathbf{p}} - \tilde{\mathbf{y}} + \lambda\tilde{\mathbf{a}}) \\ \left[\mathbf{c}_1^T\mathbf{Q}_{11}\mathbf{a}_1, ..., \mathbf{c}_1^T\mathbf{Q}_{1d}\mathbf{a}_1\right]^T + \mu\beta\exp(-\beta\Psi_1), \\ \vdots \\ \left[\mathbf{c}_{m-1}^T\mathbf{Q}_{(m-1,1)}\mathbf{a}_{m-1}, ..., \mathbf{c}_{m-1}^T\mathbf{Q}_{(m-1,d)}\mathbf{a}_{m-1}\right]^T + \mu\beta\exp(-\beta\Psi_{m-1}), \end{pmatrix}, \quad (3.3.22)$$

where we define $\tilde{\mathbf{a}} = [\mathbf{a}_1^T, \mathbf{a}_2^T, ..., \mathbf{a}_{m-1}^T]^T$ and $\tilde{\mathbf{K}}^* = \text{diag}[\tilde{\mathbf{K}}_1, ..., \tilde{\mathbf{K}}_{m-1}]$. The operator diag$[\cdot]$ builds a matrix with diagonal blocks made of the elements of the arguments. To calculate the Hessian of function (3.3.19), note that the Hessian with respect to the elements of $\mathbf{A}$ is $\tilde{\mathbf{K}}^*\mathbf{W}^*\tilde{\mathbf{K}}^* + \lambda\tilde{\mathbf{K}}^*$ where we define the matrix $\mathbf{W}^*$ as follows:

$$\mathbf{W}^* = \begin{pmatrix} \mathbf{W}_{1,1} & \mathbf{W}_{1,2} & \dots & \mathbf{W}_{1,m-1} \\ \mathbf{W}_{2,1} & \mathbf{W}_{2,2} & \dots & \mathbf{W}_{2,m-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{W}_{m-1,1} & \mathbf{W}_{m-1,2} & \dots & \mathbf{W}_{m-1,m-1} \end{pmatrix}, \tag{3.3.23}$$

with:

$$\mathbf{W}_{j,\ell} = \begin{cases} \text{diag}[p_1^{(j)}(1-p_1^{(j)}), ..., p_n^{(j)}(1-p_n^{(j)})] & if \quad j = \ell. \\ \text{diag}[-p_1^{(j)}p_1^{(\ell)}, ..., -p_n^{(j)}p_n^{(\ell)}] & if \quad j \neq \ell. \end{cases} \tag{3.3.24}$$

We need also to calculate matrices $\mathbf{M}_j$ and $\mathbf{\Phi}_j$, $j = 1, ..., m-1$, with elements defined as follows: $\mathbf{\Phi}_j(k,\ell) = \frac{\partial^2 \mathcal{L}}{\partial \psi_{jk} \partial \psi_{j,\ell}}$ and $\mathbf{M}_j(i,k) = \frac{\partial^2 \mathcal{L}}{\partial a_{ji} \partial \psi_{jk}}$, for all $k, \ell \in \{1, ..., d\}$ and $i \in \{1, ..., n\}$. The full Hessian matrix with respect to all the parameters is given as follows:

$$\tilde{\mathbf{H}} = \begin{pmatrix} \tilde{\mathbf{K}}^* \mathbf{W}^* \tilde{\mathbf{K}}^* + \lambda \tilde{\mathbf{K}}^* & \mathbf{M}^* \\ \mathbf{M}^{*T} & \mathbf{\Phi}^* \end{pmatrix}, \tag{3.3.25}$$

where we have $\mathbf{M}^* = \text{diag}[\mathbf{M}_1, ..., \mathbf{M}_{m-1}]^T$ and $\mathbf{\Phi}^* = \text{diag}[\mathbf{\Phi}_1, ..., \mathbf{\Phi}_{m-1}]^T$. Finally, the N-R update is done using:

$$\begin{pmatrix} \tilde{\mathbf{a}}^{t+1} \\ \tilde{\Psi}^{t+1} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{a}}^{t} \\ \tilde{\Psi}^{t} \end{pmatrix} - \tilde{\mathbf{H}}^{-1} \tilde{\mathbf{g}}. \tag{3.3.26}$$

where $\tilde{\mathbf{\Psi}} = [\mathbf{\Psi}_1^T, \mathbf{\Psi}_2^T, ..., \mathbf{\Psi}_{m-1}^T]^T$. Finally, Algorithm 2 shows the steps for estimating the parameters of our model. The algorithm ends when the estimation reaches a certain precision ϵ or a maximum number of iterations MAXITER.

Algorithm 1 Parameter estimation for fr-MKLR method.

```
Inputs: - Data set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)\}$.
Output: - Parameter vectors $(\mathbf{a}_j, \Psi_j)$, $j = 1, ..., m-1$.
  $\Psi_j \leftarrow \Psi_j^{(0)}$; $\mathbf{a}_j \leftarrow \mathbf{a}_j^{(0)}$;
  $t \leftarrow 1$;
  repeat
    $\mathcal{E} \leftarrow 0$;
    for $j = 1 \to m-1$ do
      Compute the gradient $\partial\mathcal{L}/\partial\mathbf{a}_j$ using Eq. (3.6.10);
      Compute the gradient $\partial\mathcal{L}/\partial\Psi_j$ using Eq. (3.6.11);
      Compute the Hessian using using Eq. (3.6.15);
      Update the entries of $\mathbf{a}_j^{(t)}$ and $\Psi_j^{(t)}$ using Eq. (3.6.16);
      $\mathcal{E} \leftarrow (\mathcal{E} + \|\mathbf{a}_j^{(t+1)} - \mathbf{a}_j^{(t)}\| + \|\Psi_j^{(t+1)} - \Psi_j^{(t)}\|)$;
    end for
    $t \leftarrow t + 1$;
  until ($\mathcal{E} < \epsilon$ OR $t >$ MAXITER)
```

3.4 Tests on simulated and UCI data

We have evaluated our method using simulated and real-world datasets. In each of our experiments, we have used Holdout method for assessing methods performance, where for each dataset we randomly generated 5 groups for learning and 5 groups for testing. The classification accuracy (CA) is measured by averaging the values of CA among the 5 groups. Let N_l and N_t be the size of a learning and testing groups in a dataset containing N data points ($N = N_l + N_t$). The CA is calculated as $1 - \frac{1}{5}\sum_{i=1}^{5}\frac{n_l^{(i)}}{N_l}$ (for training) and $1 - \frac{1}{5}\sum_{i=1}^{5}\frac{n_t^{(i)}}{N_t}$ (for testing), where $n_l^{(i)}$ and $n_t^{(i)}$ are the numbers of badly classified points in the learning and testing sets generated in the *i*th validation split, respectively. Obtained results using our method fr-MKLR are compared with MKLR, KSVM, LASSO, naive Bayes and sparse multinomial logistic regression (SMLR) [149] classifiers. For the purpose of comparison we develop fr-MKLR using the ℓ_1-norm and the ℓ_2-norm. Following the same reasoning as for fr-MKLR, the penalized multi-class NLL are given by equations 3.4.1 and 3.4.2

$$\mathcal{L}(\mathbf{a}, \Psi)_{\ell_1} = -\mathbf{y}^T\tilde{\mathbf{K}}\mathbf{a} + \mathbf{1}^T \ln\left[1 + \exp(\tilde{\mathbf{K}}\mathbf{a})\right] + \frac{\lambda}{2}\mathbf{a}^T\tilde{\mathbf{K}}\mathbf{a} + \mu\|\psi_k\|_1, \tag{3.4.1}$$

$$\mathcal{L}(\mathbf{a}, \Psi)_{\ell_2} = -\mathbf{y}^T\tilde{\mathbf{K}}\mathbf{a} + \mathbf{1}^T \ln\left[1 + \exp(\tilde{\mathbf{K}}\mathbf{a})\right] + \frac{\lambda}{2}\mathbf{a}^T\tilde{\mathbf{K}}\mathbf{a} + \mu\|\psi_k\|_2, \tag{3.4.2}$$

3.4.1 Tests on simulated data

To show the ability of fr-MKLR to select the best features for classification, we have conducted first tests using 7 simulated datasets. In these datasets, some features are purposefully set to have no discrimination between classes. We have used finite Gaussian mixture models (GMMs) to generate our datasets. Thus, each class data distribution is a GMM model of the following general form (see Table 3.4.1):

$$p(\mathbf{x}|y^{(j)} = 1) = \sum_{k=1}^{L_j} \pi_{j,k} p(\mathbf{x}|\mu_{j,k}, \mathbf{\Sigma}_{j,k}), j \in \{1, ..., m\}, \tag{3.4.3}$$

where L_j is the number of components of the GMM generating class j, $\pi_{j,k}$, $\mu_{j,k}$ and $\mathbf{\Sigma}_{j,k}$ are the a priori probability, the mean vector and covariance matrix of the *k*th component of the mixture, $k \in \{1, ..., L_j\}$.

Case of binary classification ($m = 2$)

We have conducted three tests for binary classification (Test I to III) with GMMs parameters given in Table 3.4.1. The tests aim to show the generalization capability of the algorithm with different scenarios of class data:

1. Test I (*overlapping classes*): data contain two overlapping classes where only one dimension is relevant for classification. Each class data have been generated using a bi-

Table 3.4.1 – Used GMM Parameters for generating the datasets of Tests I to VI, respectively.

Test	**Class**	**L$_j$**	**GMMs parameters**
I $N_l = 160$ $N_t = 2000$	$j = 1$	1	$\mu_{1,1} = [1, 2]^T$, $\mathbf{\Sigma}_{1,1} = \text{diag}([0.08, 0.1])$.
	$j = 2$	1	$\mu_{2,1} = [1.75 + \tau\delta, 2]^T$, $\mathbf{\Sigma}_{2,1} = \text{diag}([0.2, 0.08])$.
II $N_l = 20$ to 200 $N_t = 2000$	$j = 1$	4	$\mu_{1,1} = [1.2, 4]^T$, $\mu_{1,2} = [5.2, 4]^T$, $\mu_{1,3} = [5.2, 1.5]^T$, $\mu_{1,4} = [0.8, 1.5]^T$, $\mathbf{\Sigma}_{1,1} = \text{diag}([0.15, 0.23])$, $\mathbf{\Sigma}_{1,2} = \mathbf{\Sigma}_{1,3} = \mathbf{\Sigma}_{1,4} = \text{diag}([0.28, 0.28])$, $\pi_{1,1} = \pi_{1,2} = \pi_{1,3} = \pi_{1,4} = 0.25$.
	$j = 2$	2	$\mu_{2,1} = [3, 2]^T$, $\mu_{2,2} = [3, 4.2]^T$, $\mathbf{\Sigma}_{2,1} = \text{diag}([0.18, 0.33])$,$\mathbf{\Sigma}_{2,2} = \text{diag}([0.11, 0.15])$, $\pi_{2,1} = 0.90$, $\pi_{2,2} = 0.10$.
III $N_l = 80$ $N_t = 1200$	$j = 1$	2	$\mu_{1,1} = [1, 5.5]^T$, $\mu_{1,2} = [3, 5.5]^T$. $\mathbf{\Sigma}_{1,1} = \begin{pmatrix} 1 & 0.75 \\ 0.75 & 1 \end{pmatrix}$, $\mathbf{\Sigma}_{1,2} = \begin{pmatrix} 1 & -0.75 \\ -0.75 & 1 \end{pmatrix}$ $\pi_{1,1} = \pi_{1,2} = 0.5$.
	$j = 2$	2	$\mu_{2,1} = [1, 8]^T$, $\mu_{2,2} = [3, 8]^T$. $\mathbf{\Sigma}_{2,1} = \mathbf{\Sigma}_{1,1}$, $\mathbf{\Sigma}_{2,2} = \mathbf{\Sigma}_{1,2}$ $\pi_{2,1} = \pi_{2,2} = 0.5$.
IV $N_l = 80$ $N_t = 3000$	$j = 1$	1	$\mu_{1,1} = [1, 2]^T$, $\mathbf{\Sigma}_{1,1} = \text{diag}([0.5, 0.5])$.
	$j = 2$	1	$\mu_{2,1} = [1, 4]^T$, $\mathbf{\Sigma}_{2,1} = \text{diag}([0.5, 0.5])$.
	$j = 3$	1	$\mu_{3,1} = [4, 1]^T$, $\mathbf{\Sigma}_{3,1} = \text{diag}([0.5, 0.5])$.
V $N_l = 250$ $N_t = 3000$	$j = 1$	1	$\mu_{1,19} = [4, 4]^T$, $\mathbf{\Sigma}_{1,1} = \text{diag}([10, 0.1])$.
	$j = 2$	1	$\mu_{2,1} = [1, -2]^T$, $\mathbf{\Sigma}_{2,1} = \mathbf{\Sigma}_{1,1}$.
	$j = 3$	2	$\mu_{3,1} = [1, 8.5]^T$, $\mu_{3,2} = \mu_{3,1}$. $\mathbf{\Sigma}_{3,1} = \begin{pmatrix} 4 & 3.75 \\ 3.75 & 4 \end{pmatrix}$, $\mathbf{\Sigma}_{3,2} = \begin{pmatrix} 4 & -3.75 \\ -3.75 & 4 \end{pmatrix}$
	$j = 4$	1	$\mu_{4,1} = [-10, 1.8.5]^T$, $\mathbf{\Sigma}_{4,1} = \text{diag}([0.1, 0.8])$.
	$j = 5$	1	$\mu_{5,1} = [15, 8.5]^T$, $\mathbf{\Sigma}_{5,1} = \text{diag}([0.1, 0.8])$.
VI $N_l = 50$ to 500 $N_t = 1250$	$j = 1$	1	$\mu_{1,1} = [6, 1, 1, 1]^T$, $\mathbf{\Sigma}_{1,1} = \text{diag}([0.5, 0.5, 0.5, 0.5])$.
	$j = 2$	1	$\mu_{2,1} = [6, 1.5, 3.5, 1]^T$, $\mathbf{\Sigma}_{2,1} = \mathbf{\Sigma}_{1,1}$.
	$j = 3$	1	$\mu_{3,1} = [6, 1.5, 1, 3.5]^T$, $\mathbf{\Sigma}_{3,1} = \mathbf{\Sigma}_{1,1})$.
	$j = 4$	2	$\mu_{4,1} = [1, 1, 1, 1]^T$, $\mu_{4,2} = [1, 3.25, 1, 1]^T$. $\mathbf{\Sigma}_{4,1} = \mathbf{\Sigma}_{4,2} = \mathbf{\Sigma}_{1,1}$, $\mathbf{\Sigma}_{1,2}$ $\pi_{4,1} = \pi_{4,2} = 0.5$.

variate Gaussian. We vary the amount of class overlapping by shifting the mean of the first class in one dimension by increments $\tau \in \{1, ..., 10\}$ of a step $\delta = 0.25$.

2. Test II (*scarce learning data*): shows the generalization capability of our algorithm when learning data are scarce. Each class has been generated using a mixture of bivariate Gaussians. We vary the number of learning data per Gaussian N_g from 10 to 100.

3. Test III (*multi-modal classes*): classes are multimodal and separated by non-linear boundaries. Each class has been generated using a mixture of bivariate Gaussians.

In Figs. 3.4.1 and 3.4.2, the first, second and third rows show class boundaries obtained in Test I to Test III using MKRL, fr-MKLR and KSVM methods, respectively. Clearly, fr-MKLR has succeeded for both tests in selecting the best separating feature which led to better generalization than the other methods. Fig. 3.4.3 shows CA values obtained for both tests (by varying class overlapping for Test I and the number of training data in Test II). For the obtained CA values using the learning data, we can observe that MKLR, and KSVM have yielded sensibly the same performance. By contrast, fr-MKLR outperforms all the other methods when using the testing data. For Test III, the results are reported in Table 3.4.2. We can note that fr-MKLR outperforms the other methods on testing data. Therefore, we can conclude that our approach has better generalization capability than the other methods.

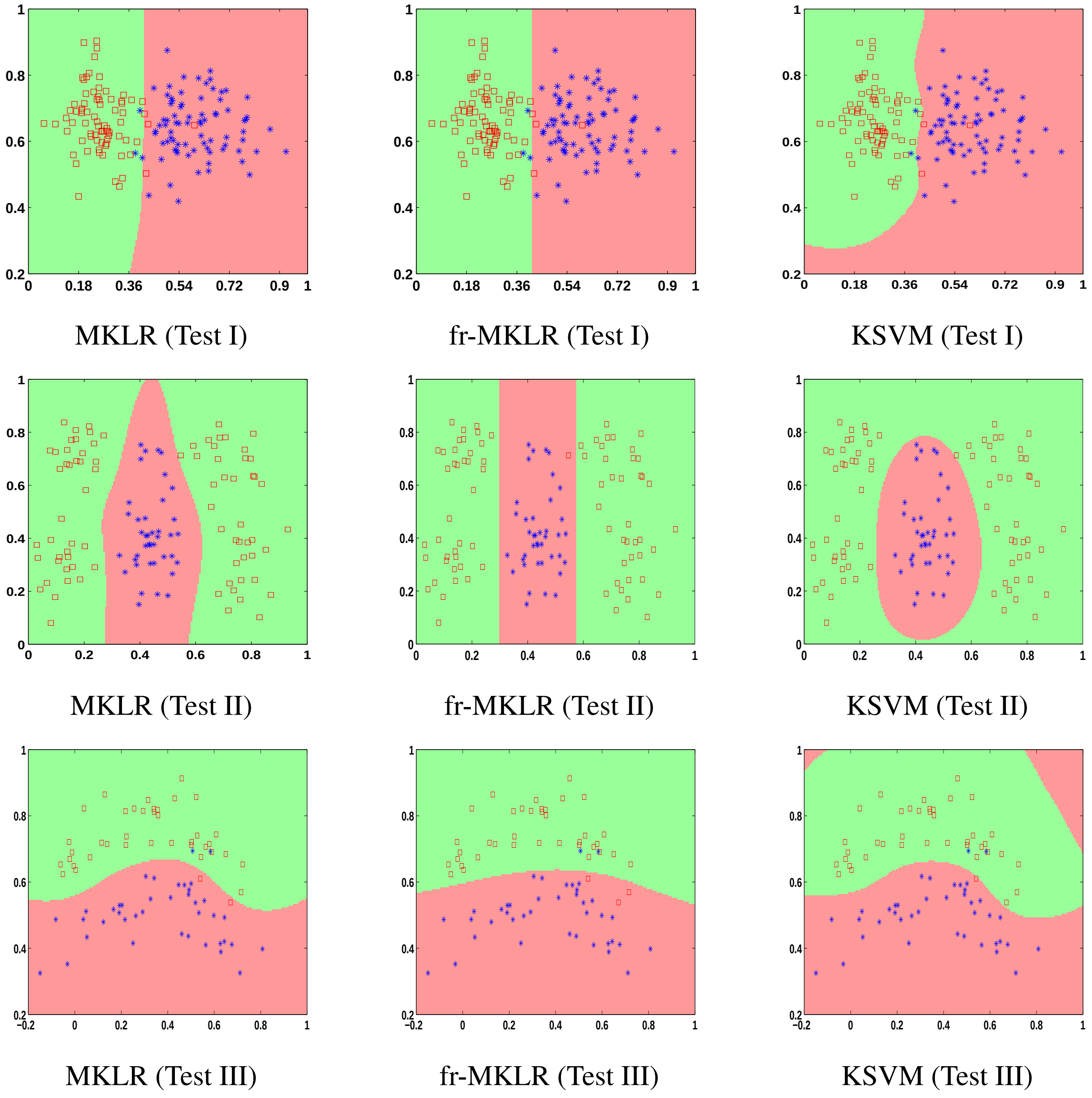
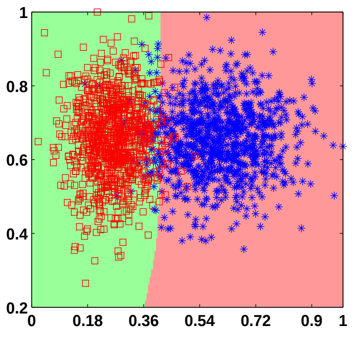
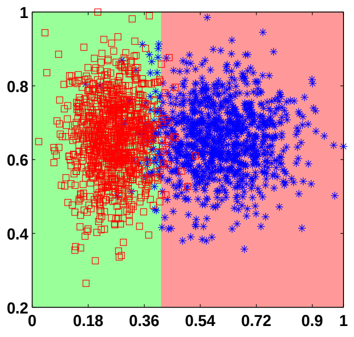
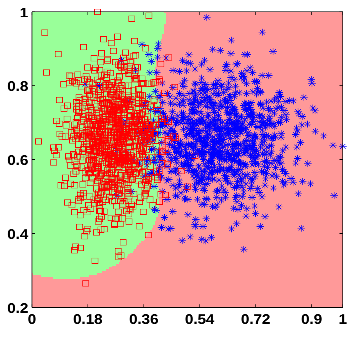
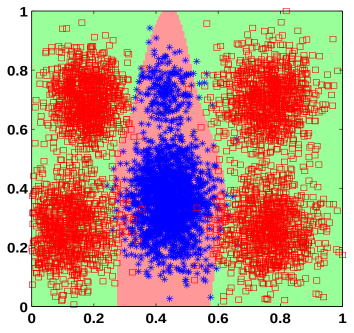


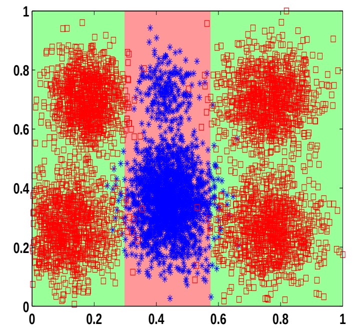
Figure 3.4.1 – Examples illustrating classification boundaries obtained by MKLR, fr-MKLR and KSVM for Test I data (first row), Test II data (second row) and Test III data (third row), respectively. For each method, a 2D scatter is shown using learning data.

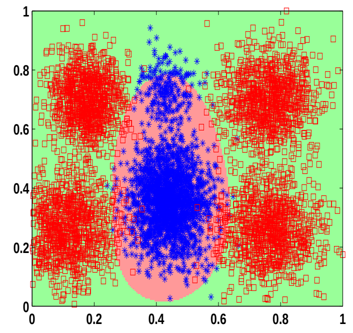

MKLR (Test I)

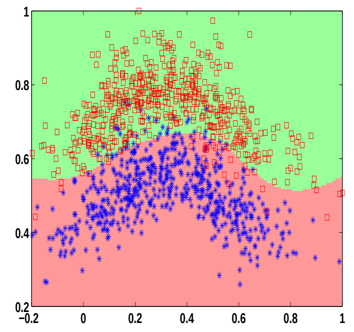

fr-MKLR (Test I)

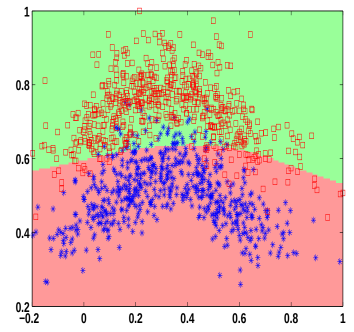

KSVM (Test I)


MKLR (Test II)


fr-MKLR (Test II)


KSVM (Test II)


MKLR (Test III)


fr-MKLR (Test III)

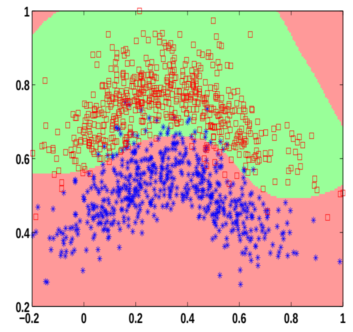
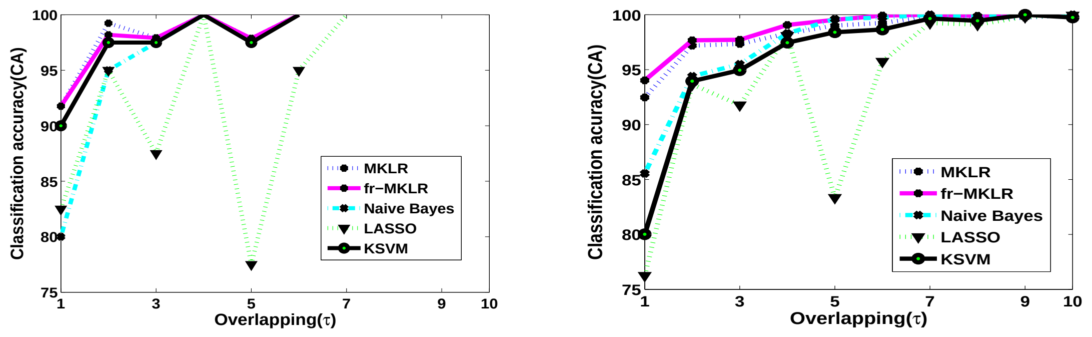

KSVM (Test III)

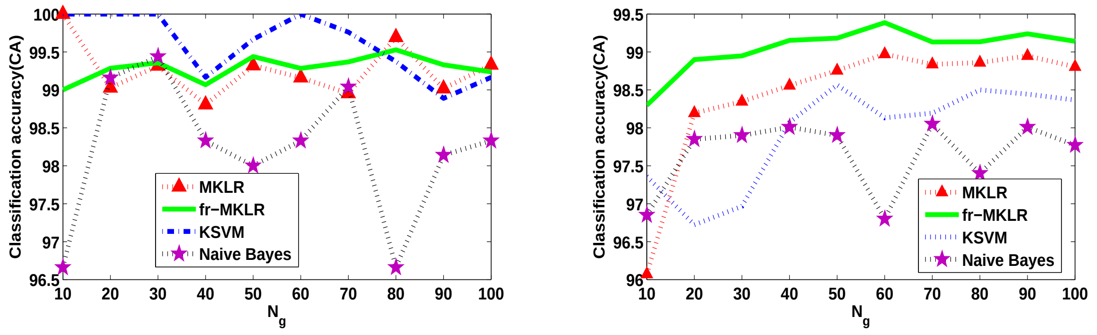
Figure 3.4.2 – Examples illustrating classification boundaries obtained by MKLR, fr-MKLR and KSVM for Test I data (first row), Test II data (second row) and Test III data (third row), respectively. For each method, a 2D scatter is shown using testing data.

Table 3.4.2 – Average classification accuracy obtained by the compared methods.

	Methods	Learning accuracy (%)	Testing accuracy(%)
Test III	MKLR	97.12	95.10
	fr-MKLR (ℓ_0)	95.85	95.60
	- (ℓ_1)	95.25	93.75
	- (ℓ_2)	95	93.50
	KSVM	96.25	93.33
	Naive Bayes	92.5	89.19
	LASSO	87.5	83.33
Test IV	MKLR	99.14	94.89
	fr-MKLR (ℓ_0)	98.8	97.26
	-(ℓ_1)	96	93.40
	- (ℓ_2)	93.33	92.87
	KSVM	95.74	91.19
	Naive Bayes	94.12	94,3
	LASSO	67.56	66.56
Test V	MKLR	100	93.23
	fr-MKLR (ℓ_0)	100	98.91
	- (ℓ_1)	99.58	98.60
	-(ℓ_2)	99.55	98.43
	KSVM	100	98.87
	Naive Bayes	100	100
	LASSO	83.25	83.31



Test I



Test II

Figure 3.4.3 – Classification accuracy (CA) obtained for Tests I and II. For each test, we show CA using (left) learning data and (right) testing data.

Case of multi-class classification ($m > 2$)

We conducted three tests (Tests IV to VI) using multi-class data. Tests IV and V use data with number of classes $m = 3$ and $m = 5$, respectively. The data of each class have been generated using mixtures of bivariate Gaussians. Test VI uses data with $m = 4$ and $d = 4$. Projection of the data on one and two dimensions are shown in Figs. 3.4.6 and 3.4.7, respectively. The GMM parameters used for generating the data of these tests are given in Table 3.4.1.

The resulting classification boundaries obtained using MKRL, fr-MKLR and KSVM are shown in Figs. 3.4.4 and 3.4.5 for Tests IV and V and corresponding CA are given in Table 3.4.2. Clearly, fr-MKLR has succeeded in selecting the best separating features for each class, which led to a better generalization than the other methods. For Test VI, CA using learning and testing data are given in Fig. 3.4.8 by varying N_l from 50 to 500. We can note that using the learning data, MKLR and KSVM had better performance than the other methods. When using the testing data, fr-MKLR and Naive Bayes lead to better performance then the other methods. Moreover, for $N_l \geq 100$, fr-MKLR has performed better than Naive Bayes. This shows that in the presence of high dimension and scarce learning data, fr-MKLR has tendency to provide better generalization capability.

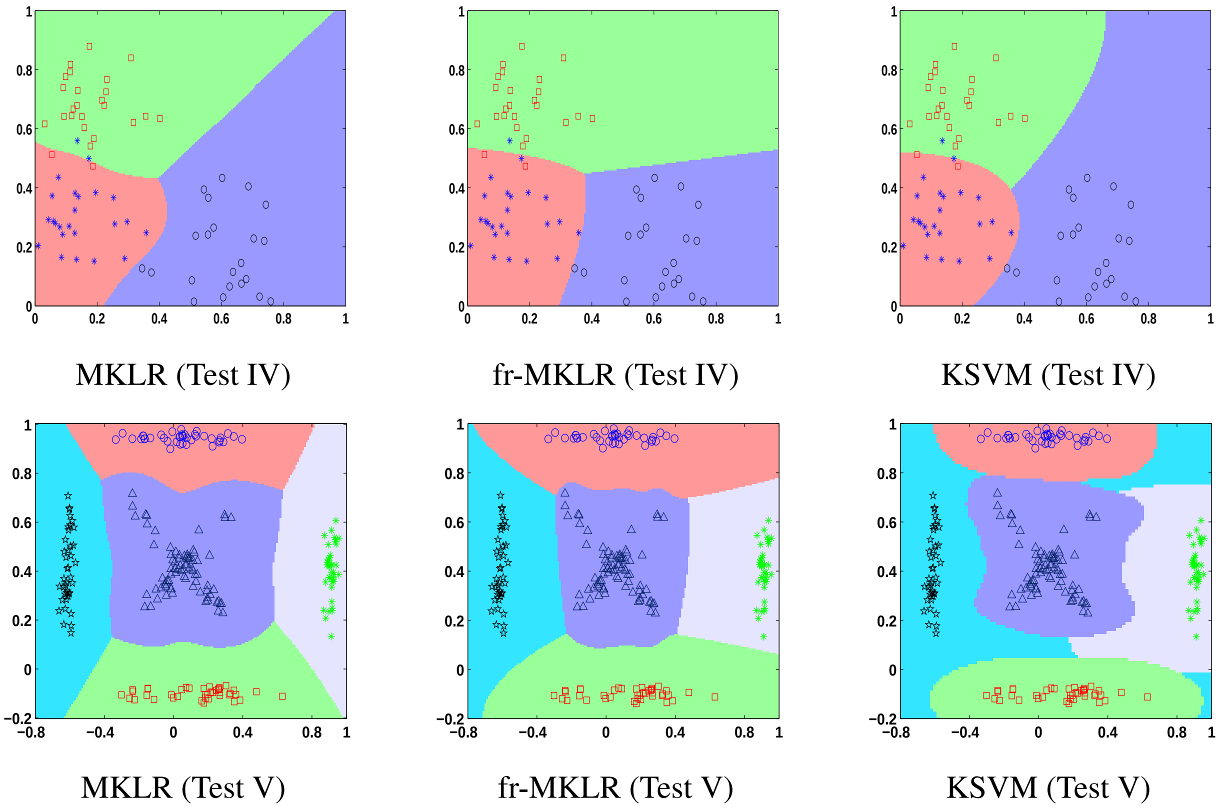


Figure 3.4.4 – Examples illustrating classification boundaries obtained by MKLR, fr-MKLR and KSVM for Test IV data (first row) and Test V data (second row) respectively. For each method, a 2D scatter is shown using learning data.

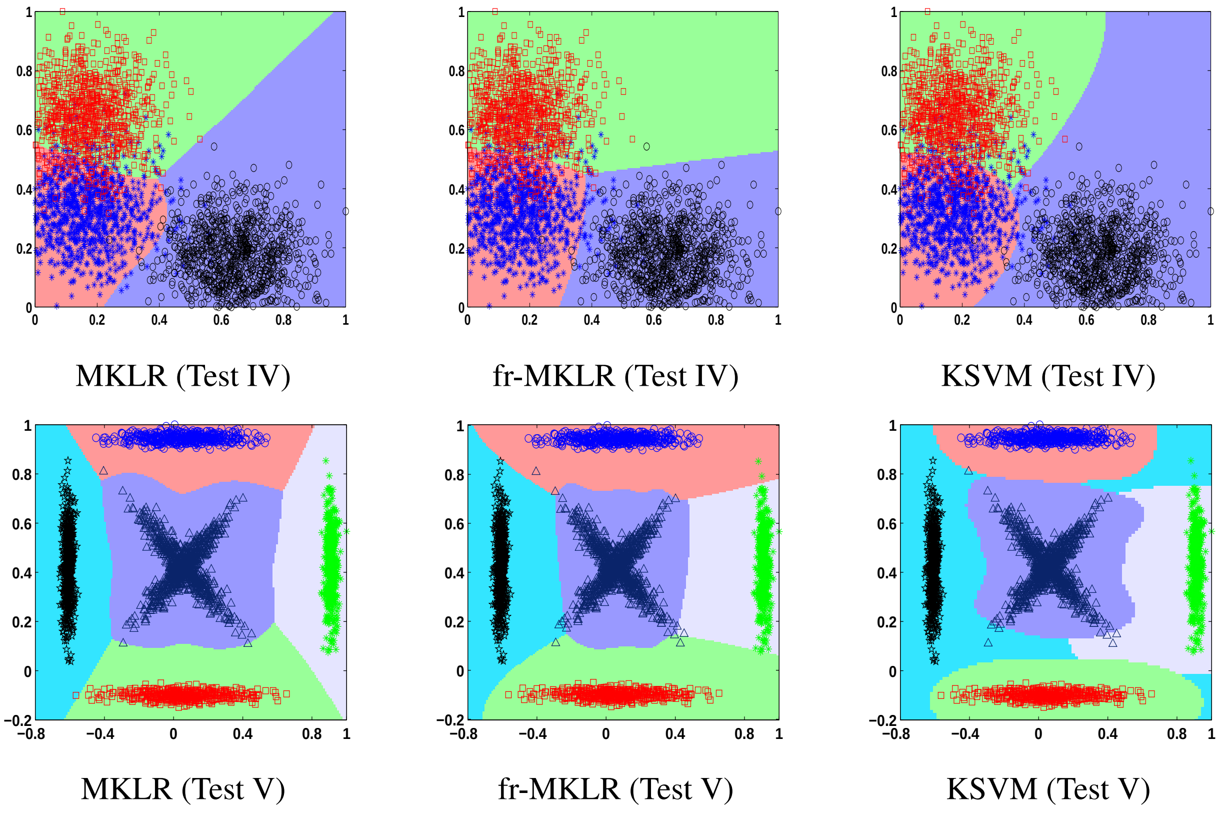


Figure 3.4.5 – Examples illustrating classification boundaries obtained by MKLR, fr-MKLR and KSVM for Test IV data (first row) and Test V data (second row) respectively. For each method, a 2D scatter is shown using testing data.

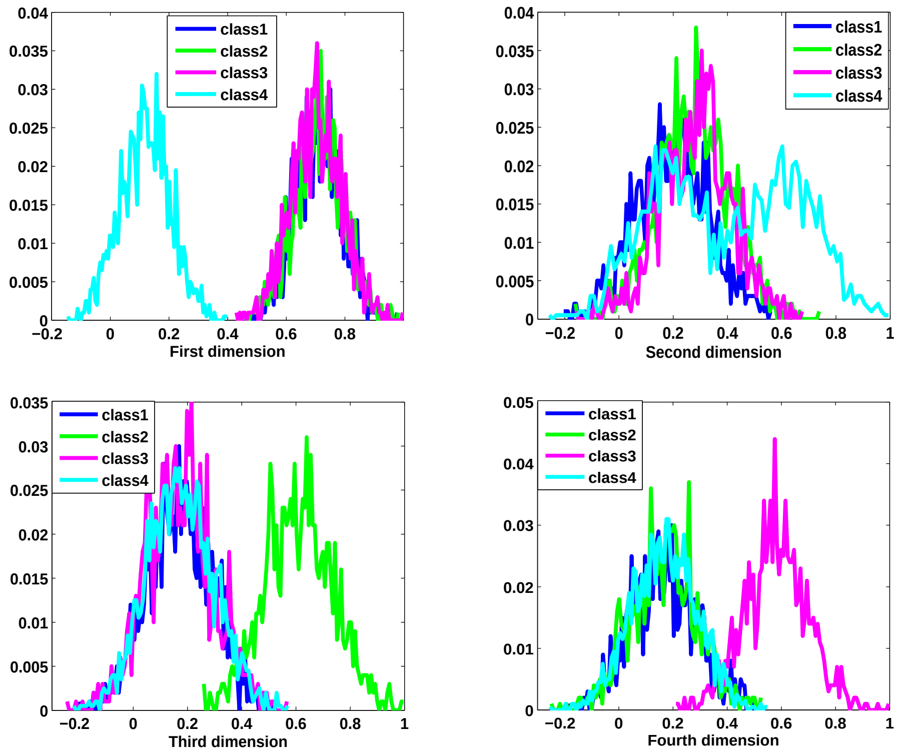


Figure 3.4.6 – 1D projection of data of Test VI on axis x_1, x_2, x_3 and x_4, respectively.

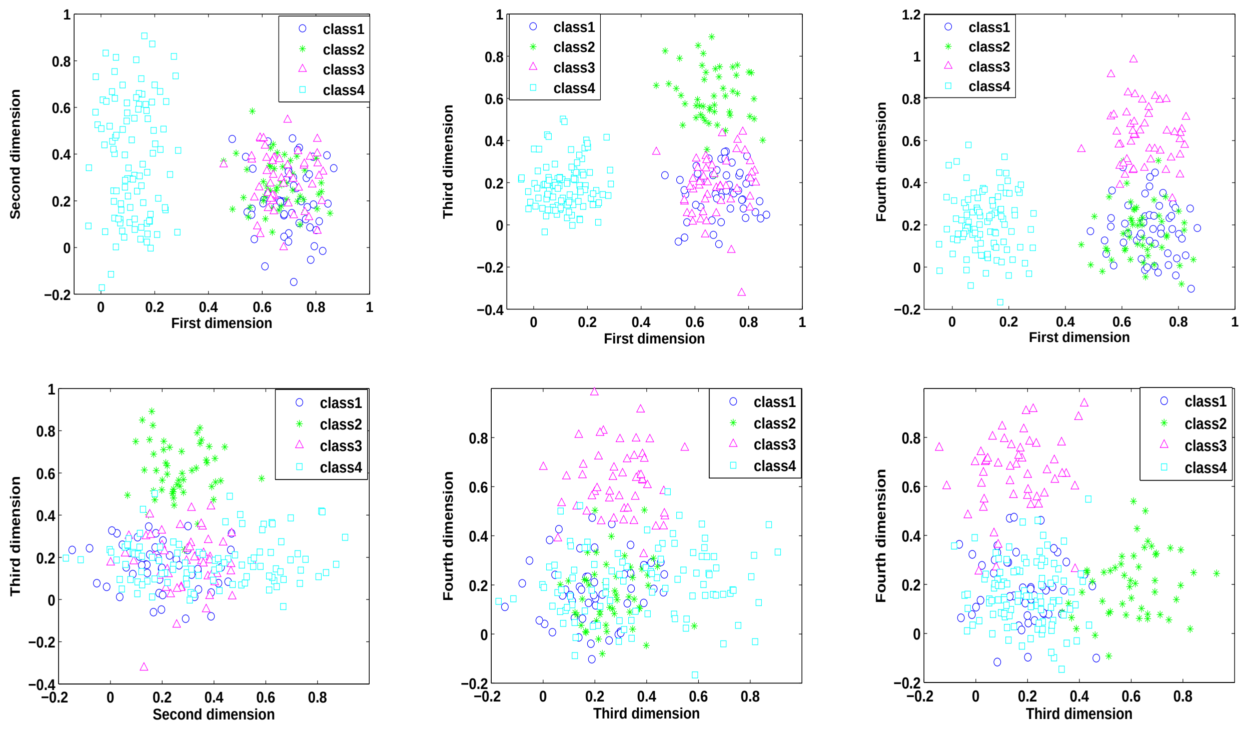
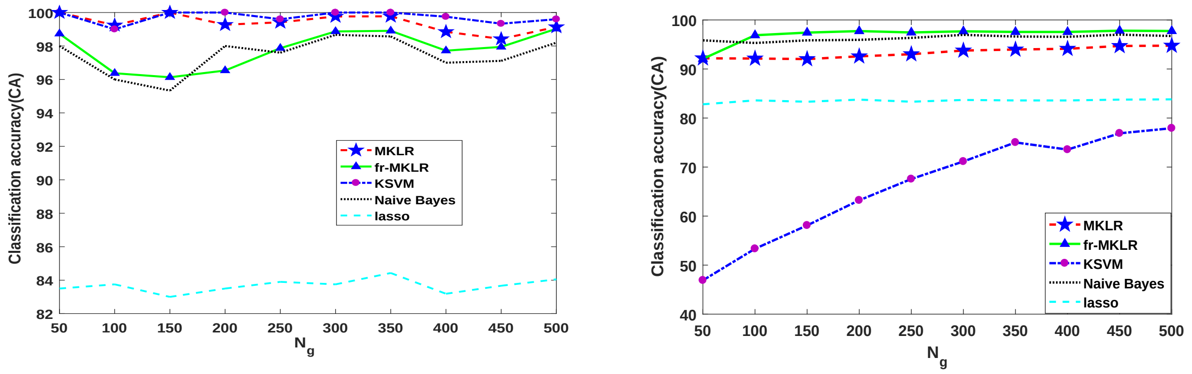
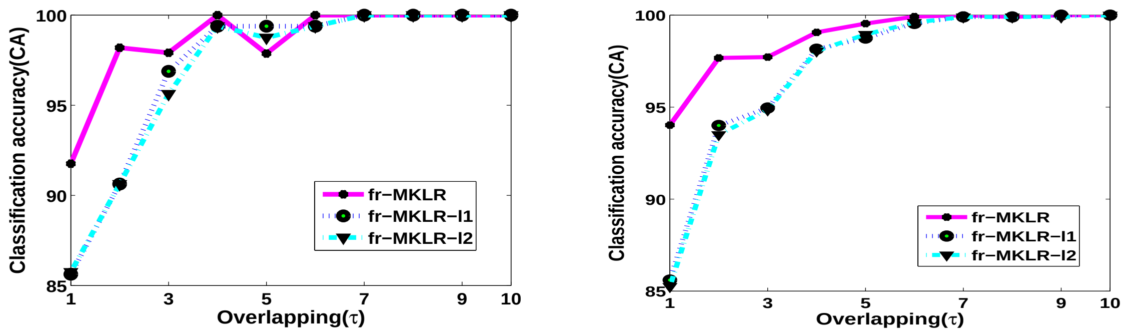


Figure 3.4.7 – 2D projection of data of Test VI.

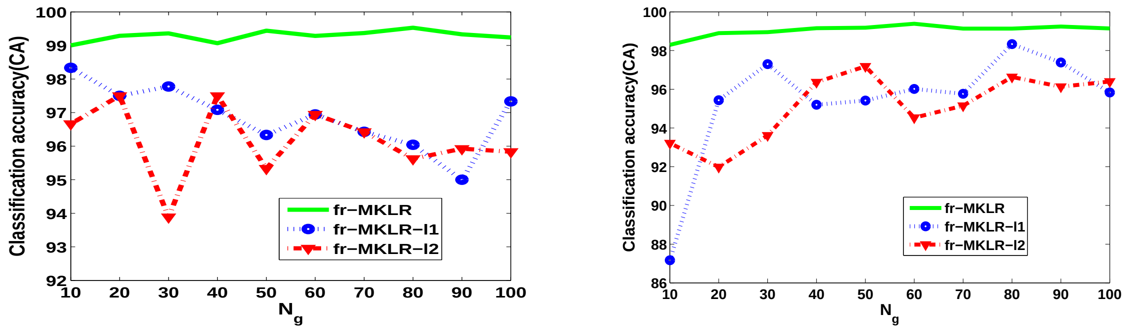


Test VI

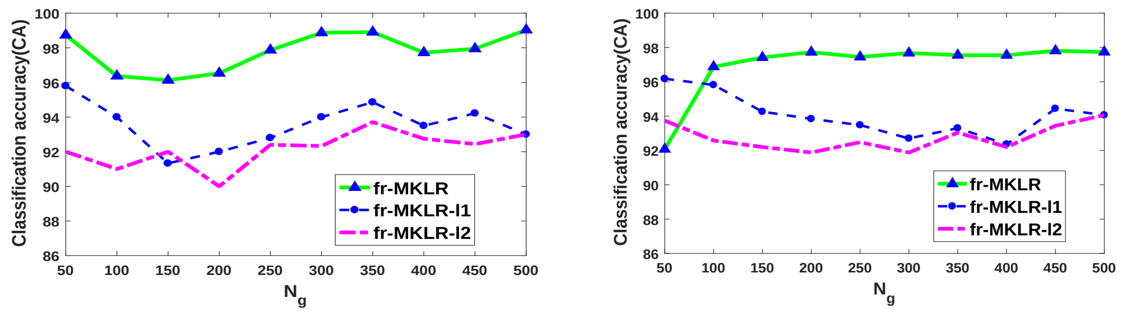
Figure 3.4.8 – Classification accuracy obtained using datasets of Test VI . For each dataset, we show CA using (left) learning data and (right) testing data.



Test I



Test II



Test VI

Figure 3.4.9 – Classification accuracy (CA) obtained in Tests I, II and VI using fr-MKLR with ℓ_1 and ℓ_2 norms, respectively.

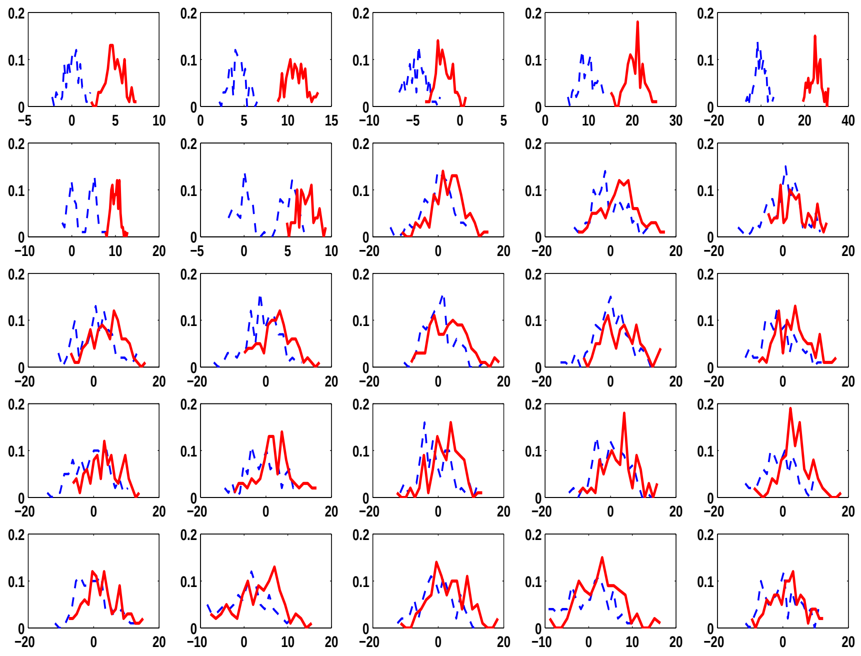


Figure 3.4.10 – Histograms of the 25 features used in Test VII: red (continuous) and blue (dashed) lines show feature distributions in classes $j = 1$ and $j = 2$, respectively.

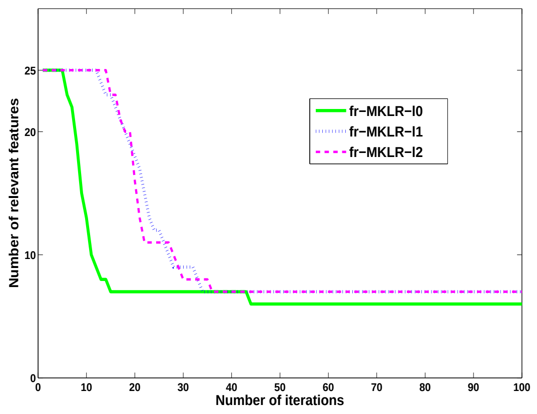


Figure 3.4.11 – Graph showing the number of retained relevant features for Test VII as a function of number of iterations.

To compare the effectiveness of using the ℓ_0 norm instead of ℓ_1 and ℓ_2 norms for obtaining sparse models for MKLR, we implemented two versions of the fr-MKLR approach using the ℓ_1 and ℓ_2 norms, respectively. Obtained results are shown for Tests I to VI in Fig. 3.4.9 and Table 3.4.2, respectively. Also, results obtained for the UCI tested datasets (presented in the next section) are shown in Table 3.4.4. We can clearly note that the ℓ_0 norm outperforms the other norms in all the tests for testing data. These results support the motivation of using the ℓ_0 norm for penalization since it provides sparse models that yield good classification generalization.

Finally, we conducted an (illustrative) test of binary classification using a simulated dataset of 25 features and $N = 200$ data, where only 6 features have a clear discrimination between the two classes (see Fig. 3.4.10). Fig. 3.4.11 presents a graph showing the number of retained relevant features (i.e., with weight $\psi_k > 0$) as a function of the number of iterations. Clearly, ℓ_0 norm has allowed to capture the exact number of relevant features and with a smaller number of iterations than ℓ_1 and ℓ_2. This test demonstrates the effectiveness of our method for obtaining good sparse models for classification.

Table 3.4.3 – Description of used UCI datasets in our experiments.

Dataset	*EMG physical action*	*image segmentation*	*wine quality*	*Ecoli*
Number of instances (N)	10^4	2310	178	336
Number of attributes (d)	8	19	13	8
Number of classes (m)	20	7	3	8

3.4.2 Tests on UCI Repository data

The datasets used for this experiments are taken from the UCI machine learning repository [60]. Tested datasets include *EMG physical action*, *wine quality*, *Ecoli* and *Image segmentation*. The description of the four datasets is given in Table 3.4.3. Note that, unlike the experiments with synthetic data where we assumed Gaussianity of class data, we can not make such assumption for the chosen UCI datasets. Values of CA calculated on the learning and testing data are shown in Table 3.4.4 and compared with MKLR, SMLR, KSVM, NB and LASSO. We also show results for different versions of fr-MKLR using different norms inducing sparsity and values for the sparsity coefficient μ. For the sparsity coefficient, three values are tested $\mu \in \{0.05, 2.5, 4.5\}$, among which the value $\mu = 2.5$ has been obtained by cross-validation. Note that when $\mu = 0.05$, almost the same results as MKLR are obtained for all datasets. When $\mu = 4.5$, the performance of fr-MKLR decreases as shown in the table. Clearly, our model fr-MKLR, with $\mu = 2.5$ obtained by cross-validation, has yielded the best results compared to other methods. These results also demonstrate the ability of our method to perform well when dealing with non-Gaussian data.

Table 3.4.4 – Average classification accuracy obtained by the compared methods for UCI datasets repository (Standard deviation in brackets).

	EMG action(M>2)	EMG action(M=2)	Wine quality	Ecoli	Image segmentation
MKLR	68.65(2.1)	65.74 (2.2)	94.44 (1.6)	82.94 (2.6)	84.35 (2.2)
fr-MKLR($\mu = 0.05$)	68.38(2.5)	65.36 (2.2)	*94.44(1.6)*	83 (2.9)	84.35 (2.3)
- ($\mu = 2.5$)	*72.52(0.8)*	**96.23 (1.6)**	**96.83 (0.9)**	**84.71 (2.5)**	**86.39 (1.6)**
- ($\mu = 4.5$)	68.20(2.5)	91.35 (2.1)	93.52 (1.8)	82.9 (3.3)	83.67(2.8)
- (ℓ_1)	63.50 (8.8)	*95.62 (1.4)*	94.44 (2.4)	71.55 (3.4)	*85.71 (1.9)*
- (ℓ_2)	62.77 (7.5)	94.53 (5.8)	92.86 (2)	70.71 (20)	83.67 (3.8)
SMLR	68.45(0.9)	90.03 (0.9)	96.56 (4)	*83.52 (8.6)*	83.54 (8.4)
KSVM	68.98 (2.5)	95.50 (0.7)	94.44 (1.8)	80.33 (2.5)	72.79 (3.8)
Naive Bayes	**85.90 (3.3)**	93.67 (1.2)	93.65 (2.4)	66.10 (7.6)	78.91 (3.7)
LASSO	62,64 (1.6)	-	62.17 (8.6)	54.81 (5.9)	64.24 (3.5)

3.5 Computational analysis

Since most of the time is taken by model training, we discuss the computational time induced by the training step. Having N_l data points, distance calculation for each kernel matrix will require $N_l(N_l - 1)/2$ steps which can be performed in parallel. The calculation of gradient and Hessian terms using Eq. (3.6.11) and (3.6.15) has a linear computational complexity $\sim O(mN_l)$. The NLL minimization is performed iteratively using Eq. (3.6.16). Therefore, the computational complexity induced by a single iteration is approximately $\sim O([m(d+N_l)]^{2.8})$ since it involves matrix inversion.

Knowing that we deal usually with scarce learning data, computational time of the above steps can be significantly reduced using newly-developed computer hardware. For example, matrix inversion can be reduced to nearly linear complexity using the method proposed in [172]. We have used the MATLAB platform on a PC with Intel(R) core(TM) i7-3920XM at 2.9GHz CPU to run our experiments and compared average execution time including both learning and testing phases. Obtained values are dressed in Table 3.5.1. We can note that, fr-MKLR and MKLR have almost similar execution times even if fr-MKLR has an additional calculation step. LASSO and Naive Bayes require lesser computation time because of use simple probability calculation. KSVM is the slowest algorithm because of the one versus all process used to deal with the multi-class case. This allows us to conclude that our approach

Table 3.5.1 – comparison of execution time for learning and testing phases in second (s).

	Learning time	testing time
fr-MKLR	1.61	0.07
MKLR	1.10	0.07
LASSO	0.51	0.002
Naive Bayes	0.279	≈ 0
KSVM	37.56	1.90

does not add much burden to classification in term of computation time.

3.6 Group feature relevance for MKLR (GFR-MKLR)

Models with sparsity constraint on solutions plays a central role in many high-dimensional classification problems [54, 52]. Sparse models usually prevent over-fitting and lead to more interpretable solutions in high-dimensional classification problems [54, 173]. In some cases, explanatory variables can be grouped together into separate factors influencing prediction of classes [173, 174]. This is the case, for example, in real world human activities captured in videos where a single activity can be decomposed into co-occurring actions performed by different persons (e.g., hand shaking, hugging, meeting, etc.) [118]. Each action, performed by a single person, incurs the motion of different body parts related to the gestures performed by the person [24]. Therefore, having sparse classification models selecting features at the gesture level is an important issue for recognizing activities involving multiple persons.

Group sparsity has been proposed in the past mainly as an extension to the *least absolute shrinkage and selection operator* (LASSO) method [52, 53]. Contrarily to LASSO which performs feature selection for individual features, group LASSO performs selection for entire groups of variables, where each group constitutes a separate explanatory factor [54]. In particular, it can be assumed that the optimal sparsity will tend to involve clusters or groups of coefficients, corresponding to preexisting groups of features [53]. While the form of the groups can be a priori known (e.g., in activity recognition, a group can correspond to all features associated with a part of the body performing a gesture), the subset of groups that is relevant to the classification task at hand can be unknown. Recently, group LASSO methods have enjoyed a tremendous success in high-dimensional classification problems [175, 176, 5]. It remains, however, that most of the proposed methods are limited to binary classification and they are usually based on linear models.

Building on the success of FR-MKLR, we propose to extend FR-MKLR to take into account group of features relevance in classification. We assume that the features are divided into non-overlapping groups, with each group playing the role of a separate factor explaining the categories of a classification problem. Therefore, each group will be assigned a separate weight shared among all features of the group. Our group sparsity model, coined GFR-MKLR, uses the ℓ_0 norm to regularize the log-likelihood function where group weights are determined according to their discrimination for classification. In what follows, we present the formulation of group sparsity in the case of binary and multiclass classification using kernel logistic regression.

Assume that we have n instances of training data $\mathbf{x}_i \in \mathbb{R}^d$, $i \in \{1, ..., n\}$, with d measured features for each instance. The features are partitioned in G groups $\mathbf{x}_i^{(g)} \in \mathbb{R}^{d_g}$, $g \in \{1, ..., G\}$ and $d_g = d/G$ and we can rewrite the full vector $\mathbf{x}_i$ as: $\mathbf{x}_i = (\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, ..., \mathbf{x}_i^{(G)})$. Therefore, instead of calculating relevance at the level of each feature, we calculate relevance for each group of features g in the multinomial kernel logistic regression. Suppose that the data are generated from m classes ($m \geq 2$). We associate an encoding vector $\mathbf{y}_i = [y_i^{(1)}, y_i^{(2)}, ..., y_i^{(m)}]^T$ for each data point $\mathbf{x}_i$.

Group of feature weighting in case $(m = 2)$
We use a weighting vector $\Psi = [\psi_1, ..., \psi_G]^T$ of the G dimension corresponding to the number of groups and we plug it into the RBF of the MKLR and we obtain a kernel of each group of features as follow:

$$\tilde{\mathcal{K}}(\mathbf{x}_r, \mathbf{x}_s) = \exp\left(-\frac{1}{2}\sum_{g=1}^{G}\psi_g\|\mathbf{x}_r^{(g)} - \mathbf{x}_s^{(g)}\|^2\right), \tag{3.6.1}$$

with $r, s \in \{1, ..., n\}$ and $\mathbf{x}_r$. Our aim by using this penalty term is decreasing weights and contribution of noisy groups of features to classification. This can be achieved by substituting the kernel $\tilde{\mathbf{K}}$ to $\mathbf{K}$ in likelihood function:

$$\mathcal{L}(\mathbf{a}, \Psi) = -\mathbf{y}^T\tilde{\mathbf{K}}\mathbf{a} + \mathbf{1}^T \ln\left[1 + \exp(\tilde{\mathbf{K}}\mathbf{a})\right] + \frac{\lambda}{2}\mathbf{a}^T\tilde{\mathbf{K}}\mathbf{a} + \mu\sum_{g=1}^{G}\left[1 - \exp(-\beta\psi_g)\right], \tag{3.6.2}$$

where $\tilde{\mathbf{K}}$ is a kernel matrix of dimension $n \times n$ with entries $\tilde{\mathcal{K}}(\mathbf{x}_r, \mathbf{x}_s)$ given by Eq. (3.6.1) and μ is a regularization parameter. This minimization can be performed through an iterative process that alternates between two steps until convergence. In the first step, we estimate the entries of the vector $\mathbf{a}$ (for binary classification, only one vector $\mathbf{a}$ is estimated). In the second step, for a given solution $\mathbf{a}$, we minimize function (3.6.2) according to the weighting vector Ψ. We use the Newton-Raphson (N-R) method to estimate the entries of $\mathbf{a}$ and Ψ. Note that the gradient of (3.6.2) with respect to $\mathbf{a}$ and Ψ are given as follows:

$$\partial\mathcal{L}/\partial\mathbf{a} = \tilde{\mathbf{K}}\mathbf{c} \tag{3.6.3}$$

$$\partial\mathcal{L}/\partial\Psi = [\mathbf{c}^T\mathbf{Q}_1\mathbf{a}, ..., \mathbf{c}^T\mathbf{Q}_G\mathbf{a}]^T + \mu\beta\text{exp}(-\beta\Psi), \tag{3.6.4}$$

where $\mathbf{c} = \big(-\mathbf{y}+\mathbf{p}+\lambda\mathbf{a}\big)$ and $\mathbf{p} = [p_1, ..., p_n]^T$ where $p_i = p(y_i = 1|\mathbf{x}_i)$. We define the matrix $\mathbf{Q}_g$, $g \in \{1, ..., G\}$, as the following Hadamard product $\mathbf{Q}_g = \tilde{\mathbf{K}} \circ \mathbf{B}_g$, where $\mathbf{B}_g$ is an $n \times n$ dimension matrix with entries defined by $\mathbf{B}_g(r, s) = -\psi_k\|\mathbf{x}_r^{(g)} - \mathbf{x}_s^{(g)}\|$, $r, s \in \{1, ..., n\}$. The final gradient vector of (3.6.2) is obtained by concatenating (3.6.3) and (3.6.4) which gives:

$$\tilde{\mathbf{g}} = \begin{pmatrix} \tilde{\mathbf{K}}\mathbf{c} \\ \left[\mathbf{c}^T\mathbf{Q}_1\mathbf{a}, ..., \mathbf{c}^T\mathbf{Q}_G\mathbf{a}\right]^T + \mu\beta \exp(-\beta\Psi) \end{pmatrix} \tag{3.6.5}$$

To calculate the Hessian of function (3.6.2), note that the Hessian of (3.6.2) with respect to

the entries of $\mathbf{a}$ is $\tilde{\mathbf{K}}\mathbf{W}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}}$, with $\mathbf{W} = \text{diag}[p_1(1-p_1), p_2(1-p_2), ..., p_n(1-p_n)]$. We define also two matrices $\mathbf{M}$ and $\mathbf{\Phi}$ with elements given by $\mathbf{\Phi}_{k\ell} = \frac{\partial^2 \mathcal{L}}{\partial\psi_k \partial\psi_\ell}$ and $\mathbf{M}_{ik} = \frac{\partial^2 \mathcal{L}}{\partial a_i \partial\psi_k}$, $\forall k, \ell \in \{1, ..., G\}$ and $i \in \{1, ..., n\}$. The full Hessian matrix with respect to the parameters $\mathbf{a}$ an $\mathbf{\Psi}$ is given by :

$$\tilde{\mathbf{H}} = \begin{pmatrix} \tilde{\mathbf{K}}\mathbf{W}\tilde{\mathbf{K}} + \lambda\tilde{\mathbf{K}} & \mathbf{M} \\ \mathbf{M}^T & \mathbf{\Phi} \end{pmatrix}, \tag{3.6.6}$$

Finally, the N-R update is done using the following iterative scheme

$$\begin{pmatrix} \mathbf{a}^{t+1} \\ \Psi^{t+1} \end{pmatrix} = \begin{pmatrix} \mathbf{a}^{t} \\ \Psi^{t} \end{pmatrix} - \tilde{\mathbf{H}}^{-1}\tilde{\mathbf{g}} \tag{3.6.7}$$

Group of feature weighting in case $(m > 2)$

We associate a feature relevance vector $\Psi^{(j)} = [\psi_1^{(j)}, \psi_2^{(j)}, ..., \psi_D^{(j)}]^T$ for each class $j \in \{1, ..., m-1\}$. Thus, we associate a separate symmetric kernel $\tilde{\mathbf{K}}^{(j)}$ for each class j encoding the class feature relevance. The kernel entries for a class j are calculated as follows:

$$\tilde{\mathcal{K}}^{(j)}(\mathbf{x}_r, \mathbf{x}_s) = \exp\left(-\frac{1}{2}\sum_{g=1}^{G} \psi_g^{(j)} \|\mathbf{x}_r^{(g)} - \mathbf{x}_s^{(g)}\|^2\right) \tag{3.6.8}$$

The new posterior probabilities of the classes given an observation $\mathbf{x}_i$ will be similar to those

given in Eq. (3.6.2) by substituting the kernel $\tilde{\mathbf{K}}^{(j)}$ to $\mathbf{K}$ for each class j. Using the ℓ_0-"norm" penalization, the new NLL is given as follows:

$$\begin{aligned}\mathcal{L}(\mathbf{A},\mathbf{\Psi}) = \sum_{j=1}^{m-1} -\mathbf{y}^{(j)^T}\tilde{\mathbf{K}}^{(j)}\mathbf{a}^{(j)} \quad &+ \quad \mathbf{1}^T \ln\left[1+\sum_{h=1}^{m-1}\exp(\tilde{\mathbf{K}}^{(h)}\mathbf{a}^{(h)})\right] \\ &+ \quad \sum_{j=1}^{m-1}\left[\frac{\lambda}{2}\mathbf{a}^{(j)^T}\tilde{\mathbf{K}}^{(j)}\mathbf{a}^{(j)} + \mu\sum_{g=1}^{G}\left[1-\exp(-\beta\psi_g^{(j)})\right]\right], \quad (3.6.9)\end{aligned}$$

where $\mathbf{A} = [\mathbf{a}^{(1)}, ..., \mathbf{a}^{(m-1)}]$ and $\mathbf{\Psi} = [\Psi^{(1)}, ..., \Psi^{(m-1)}]$. Similarly to Eqs. (3.6.3) and (3.6.4), we have $\forall j \in \{1, ..., m-1\}, \forall g \in \{1, ..., G\}$:

$$\partial\mathcal{L}/\partial\mathbf{a}^{(j)} = \tilde{\mathbf{K}}^{(j)}\mathbf{c}^{(j)} \quad (3.6.10)$$

$$\partial\mathcal{L}/\partial\Psi^{(j)} = [\mathbf{c}^{(j)^T}\mathbf{Q}_1^{(j)}\mathbf{a}^{(j)}, ..., \mathbf{c}^{(j)^T}\mathbf{Q}_G^{(j)}\mathbf{a}^{(j)}]^T + \mu\beta\text{exp}(-\beta\Psi^{(j)}), \quad (3.6.11)$$

where we define $\mathbf{c}^{(j)} = \big(-\mathbf{y}^{(j)} + \mathbf{p}^{(j)} + \frac{\lambda}{2}\mathbf{a}^{(j)}\big)$ and $\mathbf{Q}_g^{(j)} = \tilde{\mathbf{K}}^{(j)} \circ \mathbf{B}_g^{(j)}$, with $\mathbf{B}_g^{(j)}$ is an $n \times n$ matrix having entries defined by $\mathbf{B}_g^{(j)}(r,s) = -\psi_g^{(j)}\|\mathbf{x}_r^{(g)} - \mathbf{x}_s^{(g)}\|$. It follows that the gradient of $\mathcal{L}$ with respect to the vectors $\mathbf{a}^{(j)}$'s and $\Psi^{(j)}$'s will be given by:

$$\tilde{\mathbf{g}} = \begin{pmatrix} \tilde{\mathbf{K}}^*(\tilde{\mathbf{p}} - \tilde{\mathbf{y}} + \lambda\tilde{\mathbf{a}}) \\ \left[\mathbf{c}^{(1)^T}\mathbf{Q}_1^{(1)}\mathbf{a}^{(1)}, ..., \mathbf{c}^{(1)^T}\mathbf{Q}_G^{(1)}\mathbf{a}^{(1)}\right]^T + \mu\beta\text{exp}(-\beta\Psi^{(1)}), \\ \vdots \\ \left[\mathbf{c}^{(m-1)^T}\mathbf{Q}_1^{(m-1)}\mathbf{a}^{(m-1)}, ..., \mathbf{c}^{(m-1)^T}\mathbf{Q}_G^{(m-1)}\mathbf{a}^{(m-1)}\right]^T + \mu\beta\text{exp}(-\beta\Psi^{(m-1)}), \end{pmatrix}, \quad (3.6.12)$$

where we have defined $\tilde{\mathbf{a}} = [\mathbf{a}^{(1)^T}, \mathbf{a}^{(2)^T}, ..., \mathbf{a}^{(m-1)^T}]^T$ and $\tilde{\mathbf{K}}^* = \text{diag}[\tilde{\mathbf{K}}^{(1)}, ..., \tilde{\mathbf{K}}^{(m-1)}]$. The operator diag$[\cdot]$ builds a matrix with diagonal blocks made of the elements of the arguments. To calculate the Hessian of function (3.6.9), note that the Hessian with respect to the elements of $\mathcal{A}$ is given by the matrix $\tilde{\mathbf{K}}^*\mathbf{W}^*\tilde{\mathbf{K}}^* + \lambda\tilde{\mathbf{K}}^*$, where we define $\tilde{\mathbf{K}}^* = \text{diag}[\tilde{\mathbf{K}}^{(1)}, ..., \tilde{\mathbf{K}}^{(m-1)}]$. The operator diag$[\cdot]$ builds a matrix with diagonal blocks made of the elements of the arguments. We define also the matrix $\mathbf{W}^*$ as follows:

$$\mathbf{W}^* = \begin{pmatrix} \mathbf{W}_{1,1} & \mathbf{W}_{1,2} & \cdots & \mathbf{W}_{1,m-1} \\ \mathbf{W}_{2,1} & \mathbf{W}_{2,2} & \cdots & \mathbf{W}_{2,m-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{W}_{m-1,1} & \mathbf{W}_{m-1,2} & \cdots & \mathbf{W}_{m-1,m-1} \end{pmatrix}, \tag{3.6.13}$$

with:

$$\mathbf{W}_{j,\ell} = \begin{cases} \text{diag}[p_1^{(j)}(1-p_1^{(j)}), ..., p_n^{(j)}(1-p_n^{(j)})] & if \quad j = \ell. \\ \text{diag}[-p_1^{(j)}p_1^{(\ell)}, ..., -p_n^{(j)}p_n^{(\ell)}] & if \quad j \neq \ell. \end{cases} \tag{3.6.14}$$

Similarly to the case of binary clarification, we need also to calculate matrices $\mathbf{T}^{(j)}$ and $\mathbf{M}^{(j)}$ for each class j, $j \in \{1, ..., m-1\}$, with elements defined as follows: $\mathbf{T}^{(j)} = \frac{\partial^2 \mathcal{L}}{\partial \Psi^{(j)} \partial \Psi^{(j)^T}}$ and $\mathbf{M}^{(j)} = \frac{\partial^2 \mathcal{L}}{\partial \mathbf{a}^{(j)} \partial \Psi^{(j)^T}}$. The full Hessian matrix with respect to all the parameters is given as follows:

$$\tilde{\mathbf{H}} = \begin{pmatrix} \tilde{\mathbf{K}}^* \mathbf{W}^* \tilde{\mathbf{K}}^* + \lambda \tilde{\mathbf{K}}^* & \mathbf{M}^* \\ \mathbf{M}^{*T} & \mathbf{T}^* \end{pmatrix}, \tag{3.6.15}$$

where we have $\mathbf{M}^* = \text{diag}[\mathbf{M}^{(1)}, ..., \mathbf{M}^{(m-1)}]^T$ and $\mathbf{T}^* = \text{diag}[\mathbf{T}^{(1)}, ..., \mathbf{T}^{(m-1)}]^T$. Finally, the N-R update consists of the following iterative formula:

$$\begin{pmatrix} \tilde{\mathbf{a}}_{(t+1)} \\ \tilde{\mathbf{\Psi}}_{(t+1)} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{a}}_{(t)} \\ \tilde{\mathbf{\Psi}}_{(t)} \end{pmatrix} - \tilde{\mathbf{H}}^{-1} \tilde{\mathbf{g}}. \tag{3.6.16}$$

where $\tilde{\mathbf{a}} = [\mathbf{a}^{(1)^T}, \mathbf{a}^{(2)^T}, ..., \mathbf{a}^{(m-1)^T}]^T$ and $\tilde{\mathbf{\Psi}} = [\Psi^{(1)^T}, \Psi^{(2)^T}, ..., \Psi^{(m-1)^T}]^T$. Algorithm 2 shows the steps for estimating the parameters of our model. The algorithm ends when the estimation reaches a certain precision ϵ or a maximum number of iterations MAXITER.

Algorithm 2 Parameter estimation for GRF-MKLR method.

```
Inputs: - Data set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)\}$.
Output: - Parameter vectors $(\mathbf{a}^{(j)}, \Psi^{(j)})$, $j \in \{1, ..., m-1\}$.
  $\Psi^{(j)} \leftarrow \Psi^{(j)}{}_{(0)}$; $\mathbf{a}^{(j)} \leftarrow \mathbf{a}^{(j)}{}_{(0)}$;
  $t \leftarrow 1$;
  repeat
      $\mathcal{E} \leftarrow 0$;
      for $j = 1 \to m-1$ do
          Compute the gradient $\partial\mathcal{L}/\partial\mathbf{a}^{(j)}$ using Eq. (3.6.10);
          Compute the gradient $\partial\mathcal{L}/\partial\Psi^{(j)}$ using Eq. (3.6.11);
          Compute the Hessian using using Eq. (3.6.15);
          Update the entries of $\mathbf{a}^{(j)}$ and $\Psi^{(j)}$ using Eq. (3.6.16);
          $\mathcal{E} \leftarrow (\mathcal{E} + \|\mathbf{a}^{(j)}{}_{(t-1)} - \mathbf{a}^{(j)}{}_{(t)}\| + \|\Psi^{(j)}{}_{(t-1)} - \Psi^{(j)}{}_{(t)}\|)$;
      end for
      $t \leftarrow t + 1$;
  until ($\mathcal{E} < \epsilon$ OR $t >$ MAXITER)
```

3.6.1 Tests of GFR-MKLR on simulated data

To demonstrate the efficiency of our algorithm to select the most discriminative groups of features for classification, we conducted two tests on simulated data composed of $m = 2$ and $m = 3$ classes, respectively. The details of these tests are given as follows:

Binary case $(m = 2)$

For this test, we generated $n = 200$ simulated samples for two classes C_1 and C_2. Each data point has 25 features grouped into 5 predefined groups, and each group has 5 features. Figure 3.6.1 shows the histograms of features in each group for the two classes. Figure 3.6.1 shows for each class the mean value of each feature in its originating group. Notice the values of features in group $g = 2$ and $g = 5$ in class C_1 and $g = 3$ in class C_2 are set to zero, and features in groups $g = 1$ and $g = 4$ are overlapping between the two classes. Since we are dealing with a binary case, we have one vector of weights corresponding to the $G = 5$ feature groups. Figure 3.6.3 shows the obtained weights for the different groups. We can note that the weight values are proportional to the discrimination of groups between the two classes. When the group features are overlapping between classes (e.g., $g = 1$ and $g = 3$), the obtained weights are small, whereas the groups having high discrimination were assigned higher weights.

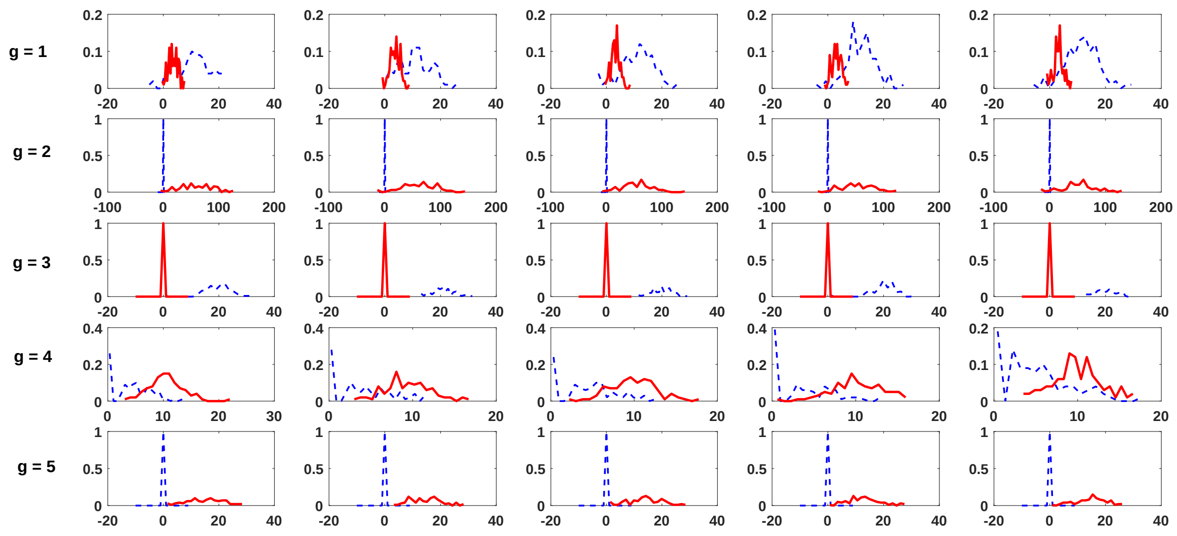


Figure 3.6.1 – Histograms of the features in each group and for each class: red (continuous) line correspond to C_1 and blue (dashed) line correspond to C_2 data.

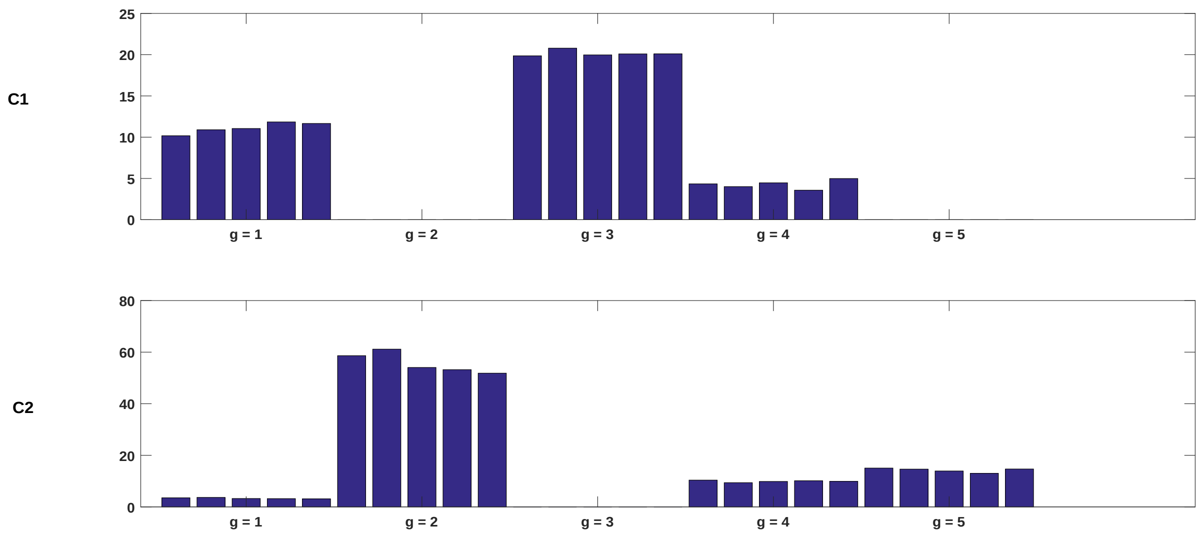


Figure 3.6.2 – Representation of the mean value of each feature in each group for classes C_1 and C_2, respectively.

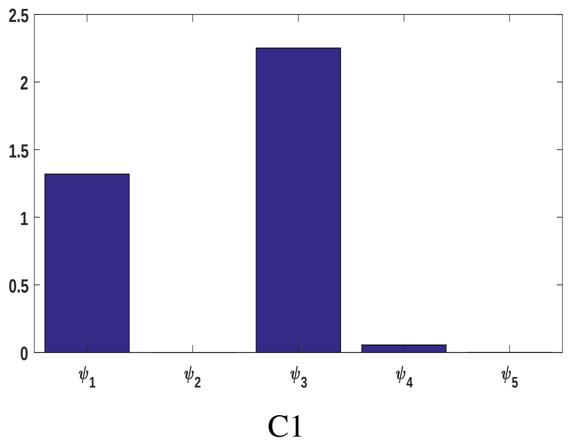


Figure 3.6.3 – Obtained weights $(\psi)$for class C_1.

Multi-class case $(m = 3)$

For this test, we generated $n = 300$ simulated samples for three classes C_1, C_2 and C_3. Figure 3.6.4) shows for each class the mean value of each feature in its originating group. Notice the values of features in groups $g = 2$ and $g = 5$ in class C_1, and $g = 3$ in class C_2, which are set to zero, whereas feasters in groups $g = 1$ and $g = 4$ in each class are overlapping. Groups in class C_3 have non-zero values and almost all groups have some overlapping between the other classes.

The obtained weight vectors were assigned to classes C_1 and C_2 and the dimension of the each vector is G. Figure 3.6.6 shows the weights of the different groups ifor $C1$ and $C2$. The assigned values are proportional to the discrimination between classes. For class C_1, $g = 3$ and $g = 5$ are the most discriminative compared to the same groups in C_2 and C_3, hence the assignment of large weight values to ψ_3 and ψ_5. Since the groups $g = 1$ and $g = 4$ are overlapping between the three class, they were assigned small values. Note that $g = 2$ is the most discriminative group for class C_2, which explains the largest value assigned to this group in class C_2. From these results, we can conclude that the weights are assigned according to their separability between classes.

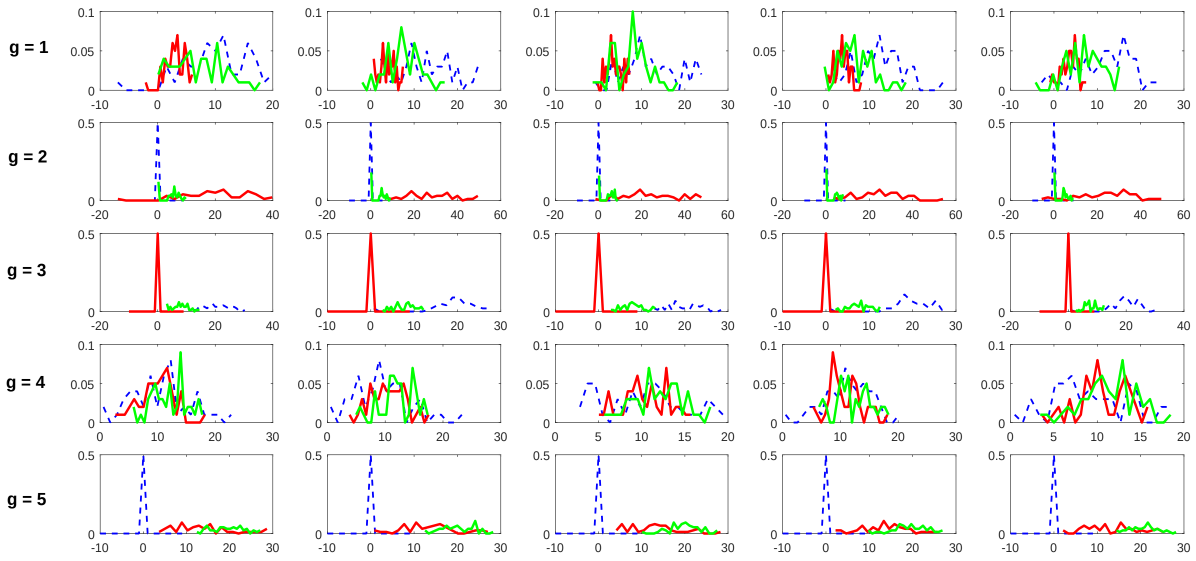


Figure 3.6.4 – Histograms of the features in each group and for each class: red (continuous) line correspond to C_1, blue (dashed) line correspond to C_2 and green (continuous) line correspond to C_3.

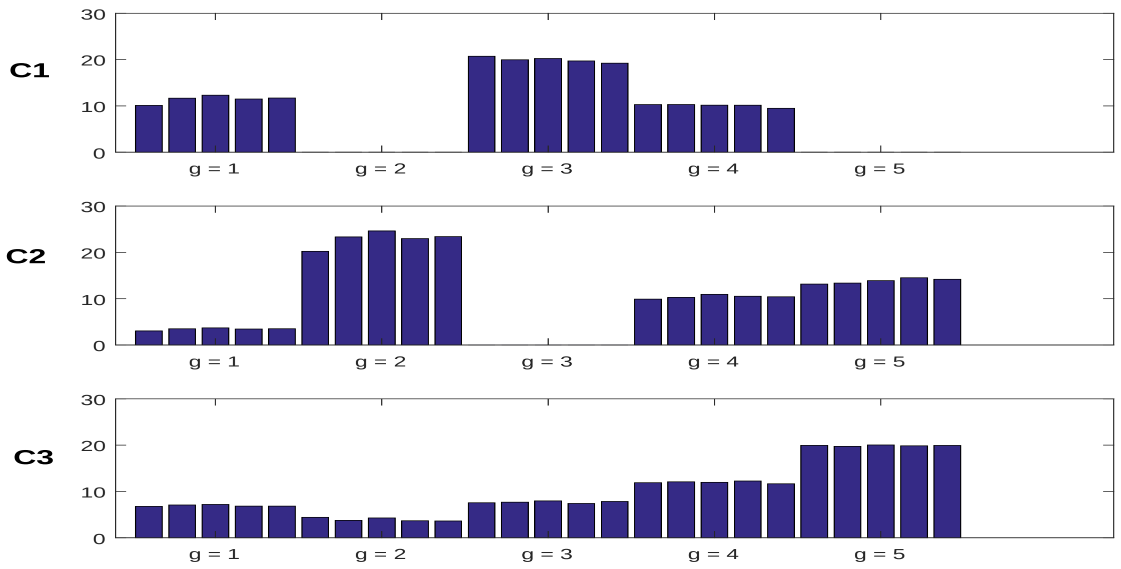
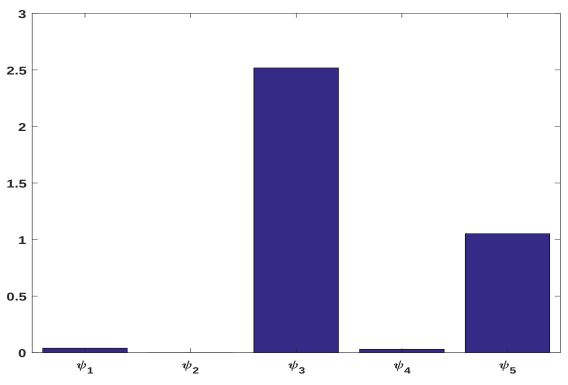
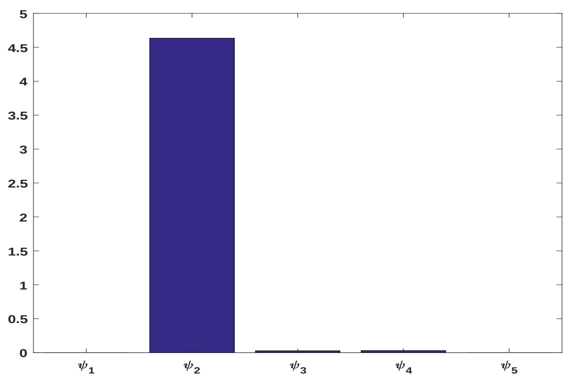


Figure 3.6.5 – Representation of the mean value of each feature in each group for C_1, C_2 and C_3, respectively.



C1



C2

Figure 3.6.6 – Obtained weights (ψ) for class C_1 and C_2, respectively.

3.7 Conclusion

In this chapter, a method for features weighting and group of features weighting embedded in multinomial kernel logistic regression (FR-MKLR and GFR-MKLR) are presented. A direct optimization method based on ℓ_0 norm is used to produce sparse model and reduce the effect of irrelevant features or group of features which leads to increase the performance of the classifier. Results obtained using FR-MKLR in synthetic and UCI datasets encouraged us to develop group of features weighting GFR-MKLR and we provide an application example to show the effectiveness of GFR-MKLR to assign the correct weights to the groups. In the the next chapter, we propose to apply FR-MKLR and GFR-MKLR recognizing single action and two-person interaction, respectively.

Chapter 4

Single action and interaction recognition

4.1 Introduction

Motivated by the success of FR-MKLR and GFR-MKLR demonstrated in the previous chapter, we propose to use our models to improve the performance of human action recognition involving single persons or interactions between two persons captured in videos. In fact, most activity descriptors proposed in the literature (e.g., HOG, SIFT, etc.) are generally high-dimensional, whereas most relevant information for discrimination may lie in a handful of dimensions. For example, having information about the body top parts is sufficient for distinguishing between 'punching' and 'greeting' actions, whereas 'running' and 'walking' can be distinguished merely by the motion of low body parts. Therefore, having sparse models for classification that take into account feature relevance can be very useful for enhancing the accuracy of action recognition. In this chapter, we treat single action and two persons interaction recognition. In both cases, we follow three steps: 1) preprocessing, 2) extracting feature descriptors and 3) recognition of actions and interactions by applying FR-MKLR and GFR-MKLR, respectively, to the descriptors.

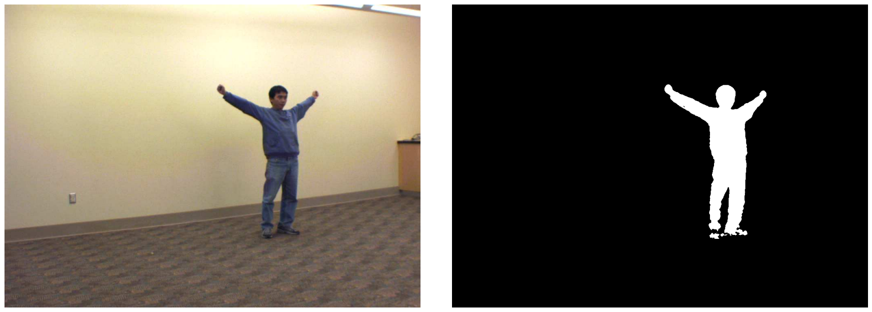
4.2 Single action recognition

Usually, action descriptors (e.g., HOG, SIFT, etc.) are extracted on raw video data and are very high-dimensional [24]. This can cause them to contain a huge amount of non-informative information for action recognition. For example, HOG describe very local spatial details about objects which may be irrelevant for action description [35, 42]. In addition, since these descriptors are calculated directly from raw video data, they can be very sensitive to occlusions and background instabilities due to noise, camera jitter and lighting changes [58]. Bravo *et al.* [177] have shown in several experiments, for example, that information used for recognition depends on the object's background. They concluded that models of recognition that have been developed for isolated objects may not generalize well to objects in dense clutter.

To enhance the accuracy of action recognition, we adopt the strategy consisting of isolating first human silhouettes from the background before performing action description. Since the human vision system is capable of recognizing several types of objects and activities based only on object silhouettes, several works proposed to perform object and simple action recognition based on silhouettes [178, 179, 180]. This is encouraged also by the advent of more efficient background subtraction techniques [181, 58]. Elgammal *et al.* [178] have proposed to track human poses through manifold learning. There method is capable of describing periodic actions such as walking. However, the method is very sensitive to action variations. Wang et al. [179] have successfully used silhouettes to recognize gaits for human identification. Finally, Wu *et al.* [180] have used human history images (MHI) calculated frond silhouettes to build a representation for simple actions. To reduce the dimension of data, PCA and K-Means clustering have been used before computing the final descriptors. However, as stated in Chapter 3, since PCA is agnostic to class labels, useful data for classification can be destroyed though the dimension reduction process.

4.2.1 Preprocessing: Background subtraction (BS)

To extract human silhouettes for single action recognition (see Figure 4.2.1 for illustration), we use BS algorithm proposed in [58]. In this approach, temporal and spatial information are combined in a probabilistic framework to detect moving objects. Temporal information is represented using mixture of generalized Gaussians (MoGG) and co-occurrence analysis between successive frames, whereas spatial information is extracted by multi-scale correlation analysis between a reference image and each frame of the video sequence. This approach has shown robustness to various complex environments such as camera jitter, self shadows, illumination changes, harsh weather conditions and background motion.



(a) (b)

Figure 4.2.1 – An example of BS for stretching action. (a): original frame, (b): BS frame.

4.2.2 Representation based on action shape context

In this section, we propose a new action representation based on shape analysis of silhouettes. The proposed method extends the shape context (SC) method [16] originally proposed for object recognition to action recognition in videos. This extension called *action shape context* (ASC) uses the variation of human shape over time to extract motion information of body parts. In addition, the ASC is made simpler by using only one reference point (i.e., center of gravity) instead of calculating a histogram for each point on the silhouette contour. The ASC representation is, therefore, invariant to object scale changes, translation and small rotations. The proposed ASC naturally carries *local* and *global* information about periodic actions performed by single individuals. *Local* information is extracted by analyzing local ASC bins over time, whereas *global* information is obtained by combining the local information in a grid maintaining the spatial and temporal structure of the action. Finally, using FR-MKLR to classify actions allows to identify the most relevant dimensions in the ASC discriminating each action category.

The input to our action representation consists of binary video frames containing the human silhouettes. Consider a video sequence **V** consisting of *N* frames $\{f_1, f_2, ..., f_N\}$. In each frame f_i, a human silhouette S_i is extracted, and we extract the silhouette boundary by combining several morphological operations such as dilations and erosions. We finally perform a uniform sampling to extract a set of *n* contour points $\mathbb{P} = \{p_1, ..., p_n\}$ representing the silhouette. These steps are necessary to eliminate noise and very small details that are meaningless for action classification. It allows also to reduce the computation time of the algorithm.

Shape context was proposed by Belongie *et al.* [16] and resulted in a good performance on the MNIST handwritten digit set and on a variety of 3D objects recognition. The basic idea of SC is illustrated in Fig 4.2.2. In its original form, SC use the context of each contour point $p_j = (x_j, y_j)$ of the object to calculate a shape histogram h_j. To each point p_j, its histogram h_j captures the distribution of the relative position of remaining $n-1$ points which are put in bins as follows:

$$h_j = \#\{q \neq p_j : (q - p_j) \in bin\}. \tag{4.2.1}$$

Since an histogram is generalized for each point, the original SC results in a very high-dimensional descriptor and is computationally prohibitive. This is not suitable for videos since we have multiple frames and high number of boundary points for each silhouette. To deal with this problem, we use one reference point corresponding to the gravity center C_i of a silhouette in frame f_i to create the SC. To We first position the polar coordinate system according to C_i of the silhouette of moving subject [182]. We binned the coordinate system using $n_r = 4$

radial bins and $n_\theta = 16$ angular bins. The choice of these values is made experimentally by varying n_r from 2 to 12 and n_θ from 4 to 24. When $n_r \leq 4$ and $n_\theta \leq 16$, we obtain a coarse description of the action, and when $n_r \geq 4$ and $n_\theta \geq 16$ we obtain a more refined description that can confuse several actions, According to Table.4.2.1, the best classification accuracy is obtained with the values $n_r = 4$ and$n_\theta = 16$. This gives 64 dimensions for the ASC descriptor, denoted by b_{ki}, $k \in \{1, ..., 64\}$, for the ith frame. Finally, to make the shape description invariant to person size, we use the Mahalanobis instead of the Euclidian distance to measure the radial distance of points relative to center of gravity. This is defined by Eq.4.2.2:

$$D_m(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}. \tag{4.2.2}$$

with μ and Σ are the mean and covariance matrix of the point positions. The use of Mahalanobis distance allows ASC to adapt the descriptor to the shape of the silhouettes extracted from video frames. To differentiate this representation with the original SC version,we call it *action shape context* (ASC). The ASC has the following properties:

- *Invariance to the affine transformation:* this is due to the positioning of the coordinate system in the gravity center of the moving silhouette. Therefore, any displacement in the plane has no effect on the ASC. Distance are normalized relative to the mean distance which makes ASC scale invariant. The use of Mahalanobis distance enables the ASC to adapt to different forms of silhouettes (human with different size) and express well the repartition of contour points relative to the gravity center corresponding to the reference point.

- *Dimensionality reduction*: Compared to the original SC [16], the ASC reduces considerably the dimension of the descriptor. This has a direct impact on reducing the processing time and the complexity the classification algorithm. Even with fewer dimensions than the SC, the ASC allows to capture global and local features of human actions described by silhouettes. Local body part motion is detected by bin analysis across the frames of the sequence, while the global aspect of motion is captured by combining the local features in a structured way.

Table 4.2.1 – Average classification accuracy obtained by varying n_r and n_θ.

n_r \ n_θ	4	8	12	16	20	24
2	57.41	65.93	74.81	80.37	77.04	78.15
4	69.26	75.93	82.59	**93.34**	88.52	79.63
6	77.78	83.33	89.63	88.89	88.15	83.33
8	80	87.04	87.41	88.52	90.74	88.89
10	81.85	84.81	87.78	87.78	89.63	*91.85*
12	84.81	81.48	86.67	86.30	91.48	90.37

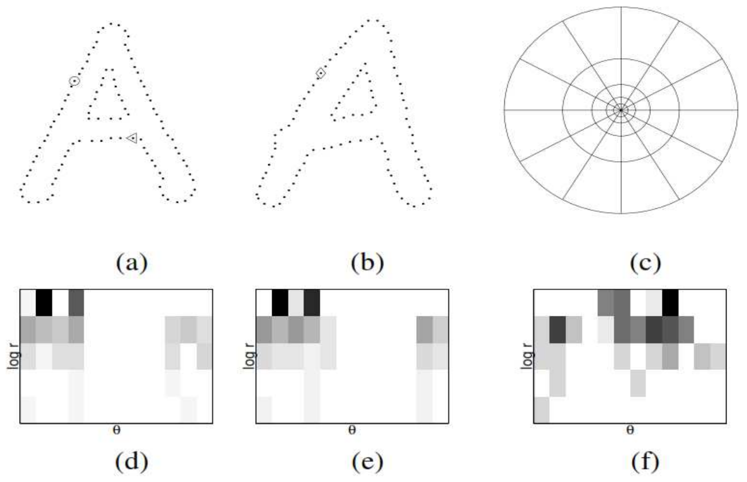


Figure 4.2.2 – Shape contexts. (a,b) Sampled edge points of two shapes. (c) Diagram of log-polar histogram bins used in computing the shape contexts. 5 bins for log r and 12 bins for θ were used. (d-f) Example shape contexts for reference samples marked by ○, ◇, ◁ in (a,b). Each shape context is a log-polar histogram of the coordinates of the rest of the point set measured using the reference point as the origin. (Dark=large value.) Note the visual similarity of the shape contexts for ○ and ◇, which were computed for relatively similar points on the two shapes. By contrast, the shape context for ◁ is quite different [16].

To extract motion information, we use a binary code (0/1) for each bin b_k in the ASC descriptor, meaning the presence (1) or absence (0) of edge points inside the bin. Local motion information at the kth bin is quantified by analyzing the presence of points at the bin for a number of frames in the video sequence. Let $\mathbf{Lb}_{ki}$ denote an activation variable which relates if the bin contains contour points or not. This is defined as follows:

$$\mathbf{Lb}_{ki} = \begin{cases} 1 & if \quad |b_{ki}| > 0. \\ 0 & \text{otherwise.} \end{cases} \tag{4.2.3}$$

where $|b_{ki}|$ designates the number of contour points contained inside the bin b_{ki}. We finally calculate the mean $\mathbf{mLb}_k$ and the variance $\mathbf{vLb}_k$ of $\mathbf{Lb}_{ki}$ across the frames as follows:

$$\mathbf{mLb}_k = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{Lb}_{ki}). \tag{4.2.4}$$

$$\mathbf{vLb}_k = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{Lb}_{ki} - \mathbf{mLb}_k)^2. \tag{4.2.5}$$

Note that $(\mathbf{mLb}_k, \mathbf{vLb}_k) \in [0, 1]$. The combination of the different $\mathbf{mLb}_k$ and $\mathbf{vLb}_k$ in grid manner leads to a global action descriptor *Gb* defined as follows:

$$\mathbf{Gb} = (\mathbf{mLb}_1, \mathbf{mLb}_2, ..., \mathbf{mLb}_{n_r \times n_\theta}, \mathbf{vLb}_1, \mathbf{vLb}_2, ..., \mathbf{vLb}_{n_r \times n_\theta}). \tag{4.2.6}$$

Knowing that we have 64 bins, we obtain a 128 (2×64) dimensional vector for the action description. The concatenation of $\mathbf{mLb}_k$ and $\mathbf{vLb}_k$ in a certain order starting from the small radius and going in the opposite direction of clockwise will lead to $\mathbf{Gb}$. Fig 4.2.3 shows an example of active bins in a *hand-waving* action video frame. Active bins are colored in white which means the presence of edge points, and dark bins mean the absence of edge points (inactive bins).

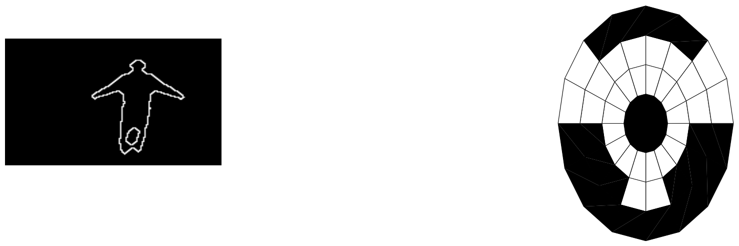


Figure 4.2.3 – An example of ASC for hand wave action. (a) current frame, (b) ASC display.

Figs. 4.2.4 and 4.2.5 show two different examples of actions and their calculated ASC. Fig. 4.2.4 shows ASC for in-place actions *hand-clapping* and *hand-waving* carried out by different subjects but starting at the same pose. We can notice the evolution of ASC content as the action goes through. Fig. 4.2.4 shows ASC for actions with displacement, namely *walking* and *running*. The final ASC for the action examples in *hand waving*, *hand clapping*, *walking* and *running* calculated by Eq. (4.2.4) are shown in Fig.4.2.6. Each bin contains the mean variation for each action though the video sequence.

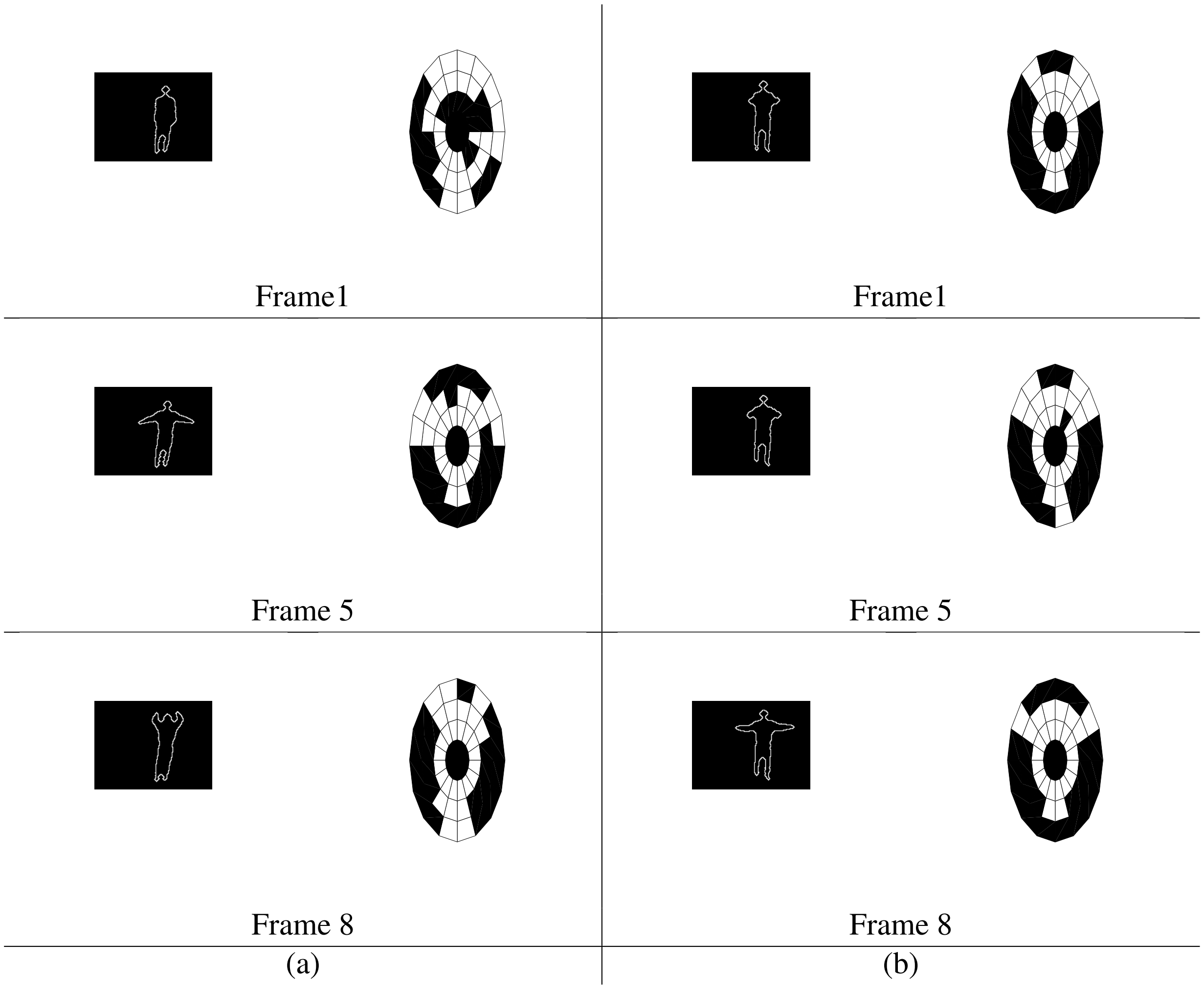


Figure 4.2.4 – Action Frames examples and their corresponding ASC for frames 1, 5 and 8 for actions: (a): hand waving and (b) hand clapping.

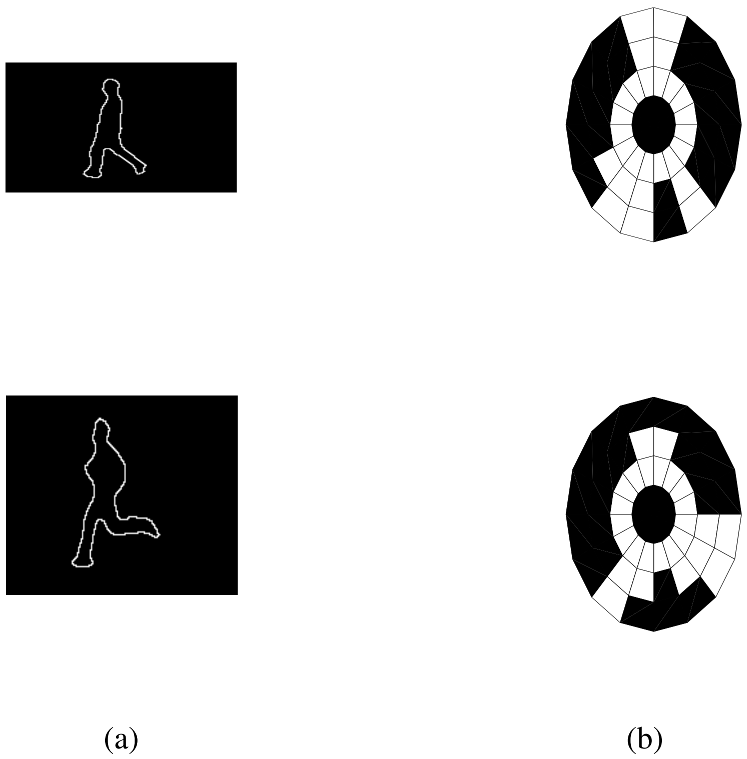


Figure 4.2.5 – Action frames examples (a) and their corresponding ASC (b). From top to down, frames correspond to walk and run actions, respectively.

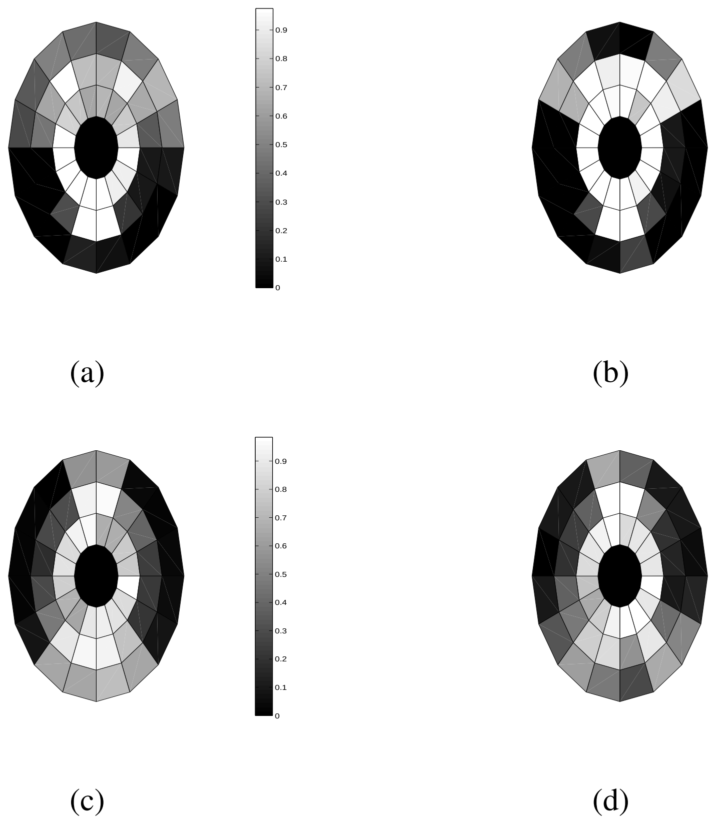


Figure 4.2.6 – Mean of ASC for the video sequences in actions: (a) hand waving, (b) hand clapping, (c) walking and (d) running.

ASC refinenment using restricted Boltzmann machines

To ensure optimal high-level representation of actions, we added a layer of dimensionality reduction using restricted Boltzmann machines (RBM) [183]. We recall that RBM [183] is a generative stochastic artificial neural network that learns a probability distribution over its input data with the restriction that there is no within-layer connections. It contains a set of visible units $v \in \{0,1\}^D$, and a set of hidden units $h \in \{0,1\}^P$. An energy function E is defined for each joint configuration (v,h) of the visible and hidden units as follows:

$$E(v,h) = -\sum_{i \in data} b_i v_i - \sum_{j \in features} b_j h_j - \sum_{i,j} v_i h_j w_{ij}. \tag{4.2.7}$$

b_i and b_j represent the biases of visible and hidden units, respectively, and $w_{i,j}$ represents the weight between them. RBM can be used as a non-linear dimensionality reduction algorithm [184]. It transforms features from the input data into a lower dimensional space by capturing the dependencies between the different features. In our case, the RBM uses 64 hidden units to process the 128 dimensions of ASC descriptor. Then, we use the output of RBM to action classification using the FR-MKLR model.

4.2.3 Experimental results

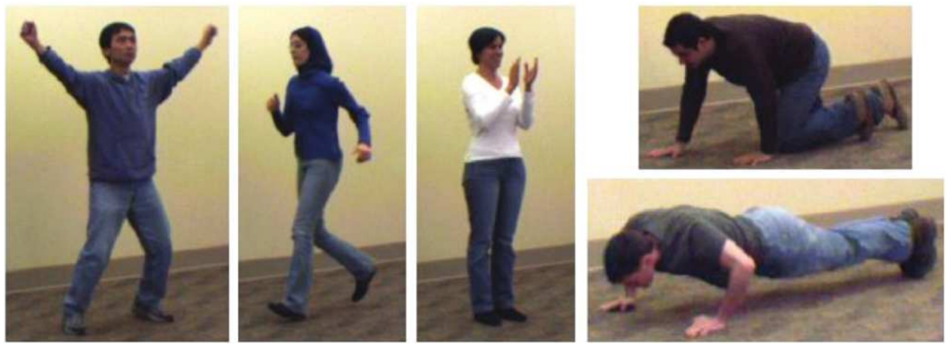
In this experiment, we use three datasets to evaluate our method for action recognition on three standard datasets, nemaly KTH [34], UIUC [49] and the I3DPost [61]. Each dataset contains a single person performing basic actions (examples of actions are shown in Fig.4.2.7). The description of each dataset is as follows:

- KTH contains 6 type of actions (*walking*, *jogging*, *running*, *boxing*, *hand waving (HW)*, *hand clapping (HC)*) performed several times by 25 subjects in 4 different scenarios: *outdoors*, *outdoors with scale variation*, *outdoors with different clothes*, and *indoors*, the total of 600 videos. For our experiments we use 180 videos with 30 videos per class.

- UIUC consists of 14 actions (*walking*, *running*, *jumping*, *waving*, *jumping jacks* (J_J), *clapping*, *jumping from sit up (JS)*, *raising one hand (RH)*, *stretching out*, *turning*, *sitting to standing (SS1)*, *crawling*, *pushing up*, *standing to sitting (SS)*) performed by 8 subjects in total it contain 532 videos. For this dataset, we use 24 examples for each class.

- I3DPost contains multi-view actions of 768 videos captured with 8 cameras and performed by 8 subjects (2 females and 6 males) for 12 actions (*bend*, *hand wave (HW)*, *jump*, *jump in place (JP)*, *run*, *walk*, *run-fall (RF)*, *run-jump-walk (RJW)*, *sit-stand-up (SS)*, *walk-sit (WS)*, *handshake*, *pull*). Note that since we aim to classify only single person action, we removed the last two actions (*handshake*, and *pull*) from our tests and we use one view for each action and a total of 8 examples per class.

(a) KTH datset


(b) UIUC dataset

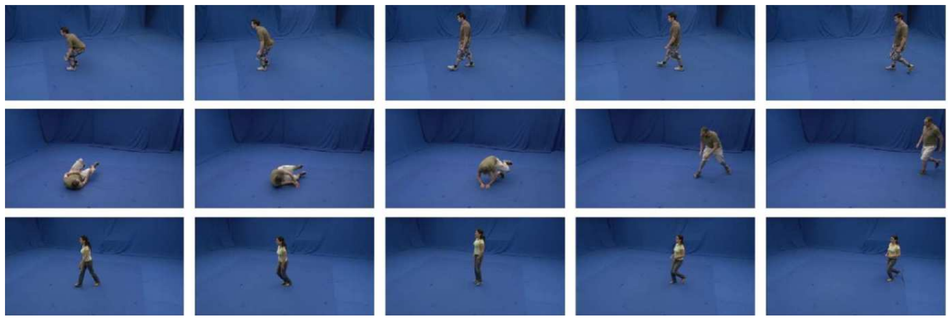
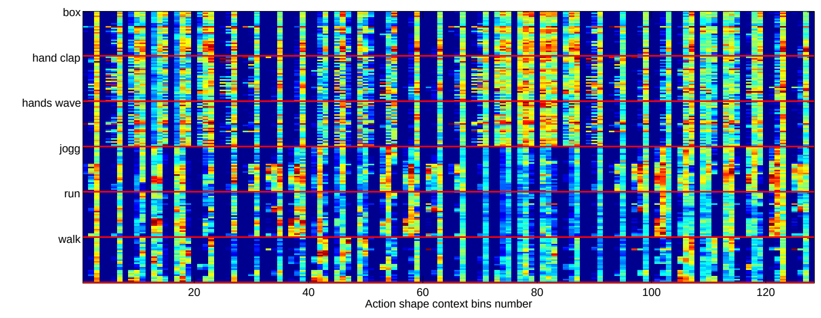

(c) i3dPOST dataset

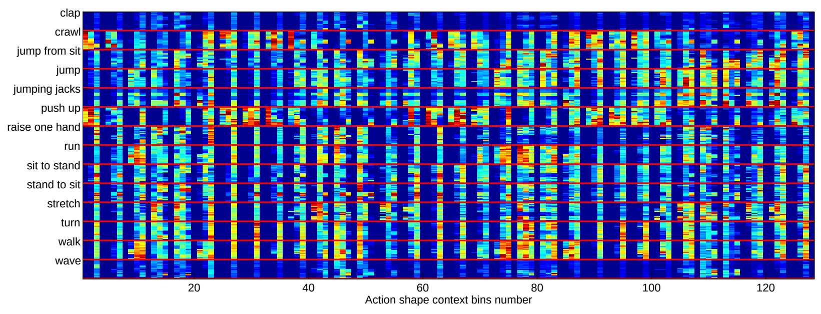
Figure 4.2.7 – Examples of action datasets.

We first extract ASC features from the videos in each dataset. Examples of resulting descriptors are shown in Fig.4.2.8 where the blue color represents the zero or very low values of the descriptor (i.e., absence or small variation of motion in the bin), and colors ranging from yellow to red represent the most important values of descriptor (i.e., presence of motion in the considered bin). This allows to visualize the variability and the sparseness of the action descriptor.

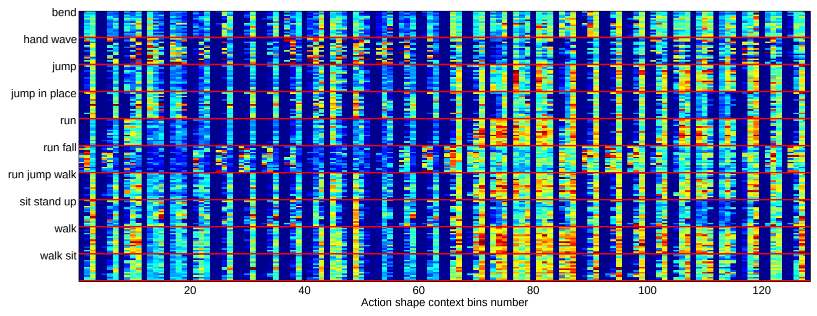
For evaluation, we follow the same procedure as for simulated data by using holdout method for assessing methods performance. We use cross-validation for fixing the hyper-parameters of the FR-MKLR model. Then for each dataset, we randomly generated 5 groups for learning the model and 5 groups for testing and we average the values of CA among the 5 groups as follows: $1-\frac{1}{5}\sum_{i=1}^{5}\frac{n_l^{(i)}}{N_l}$ (for training) and $1-\frac{1}{5}\sum_{i=1}^{5}\frac{n_t^{(i)}}{N_t}$ (for testing), where N_l and N_t are the size of a learning and testing groups in a datasets containing N data points ($N = N_l + N_t$) and $n_l^{(i)}$ and $n_t^{(i)}$ are the number of badly classified points in the learning and testing sets generated in the *i*the split, respectively.



(a) KTH



(b) UIUC



(c) i3dPOST

Figure 4.2.8 – Action Shape Context (ASC) obtained for KTH, UIUC and I3Dpost datasets.

Table 4.2.2 gives average values of CA obtained using the three datasets for the compared methods. These include fr-MKLR, MKLR, KSVM, and LASSO using our action description based on SCD and the baseline and recent methods described in Chapter 2 section 2.1 for each dataset, namely: interest points + KSVM [34], interest points + pLSA and LDA [66], space-time features + SVM [32], Optical flow + KNN [185], local space-time features

+ part-based model + multi-task learning [50], parameterized representation + discriminative classifiers [186], dense trajectories + SVM [187] and recent results using convolutional neural networks [102, 101, 99, 100, 11] for KTH. Optical flow + Correlated Topic Model (CTM) [188], spatiotemporal volumes + KNN [49], dense trajectories + motion boundary histogram + SVM [5] and depth map + skeleton structure + multi-kernel learning [189] for UIUC. and For i3DPost multi-view dataset, comparison is made with reported results in [190] using 3D motion context (3D-MC) and harmonic motion context (HMC)+ normalized correlation, motion context (MC) and transform surface (RT) + SVM [191] and discrete Fourier transform (DFT) + cosine similarity [61] using only 5 among the 12 actions of the dataset.

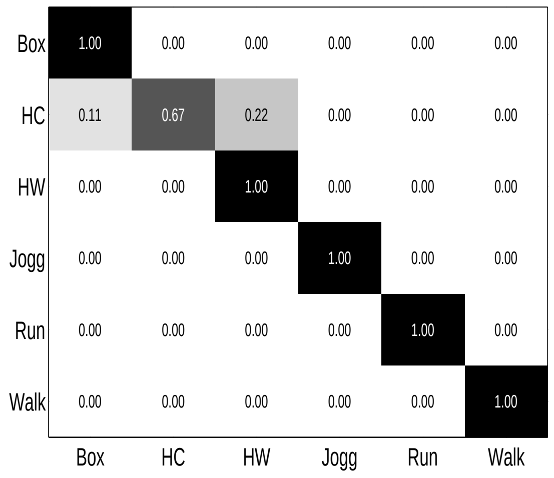
We can note that fr-MKLR outperforms almost all methods for the KTH and UIUC datasets. For KTH, authors in [187, 50, 186] have improved the video descriptors to boost the performance of action recognition. Therefore, they obtained better performance than fr-MKLR. However, by applying the RBM, we obtained the best performance among all methods. The same performance has been obtained for the UIUC dataset. For i3DPost dataset, [191] has obtained higher performance than our method. This is partly due to the quality of the features used in [191] which take into account the multi-view setting of the dataset. Indeed, 3D motion context (MC) using all views information in [190] and MC with discriminative views in [191] perform well than using the same features for all the views. Nonetheless, the application of RBM has improved considerably our results. Finally, we must put on emphasis the performance gap between RBM+MKLR and RBM+fr-MKLR. Indeed, the majority of deep learning methods apply softmax functions (e.g., MKLR) on the top layer of the network for classification. Given the obtained performance by fr-MKLR over MKLR, our method constitutes a good alternative for softmax functions in classification problems using DNNs.

Table 4.2.2 – Average classification accuracy obtained by compared methods for three data base: KTH(6 classes), UIUC(14 classes) and i3dpost(10 classes), LOO means Leave one Out and (-) means values not reported in the original papers.

Dataset	Methods	Evaluation method	Learning Accuracy (%)	Testing Accuracy (%)
KTH	*Banerjee et al.* [100]	random splits	-	90
	Charalampous et al.[102]	random splits	-	91.99
	Ji et al.[11]	random splits	-	90.2
	Kai et al. [101]	random splits	-	88.76
	Laptev et al. [34]	random splits	-	91.8
	Li et al.[187]	random splits	-	97.6
	Liu et al.[50]	random splits	-	94.3
	Niebles et al. [66]	LOO	-	83.33
	Ravanbakhsh et al.[99]	LOO	-	94.1
	Schuldt et al.[32]	random splits	-	71.7
	Yang et al.[185]	one clip	-	75
	Yuan et al.[186]	random splits	-	96.3
	LASSO	random splits	74.69	40
	KSVM	random splits	94.44	76.67
	MKLR	random splits	*97.14*	87.77
	RBM +**MKLR**	random splits	100	98.52
	fr-MKLR	random splits	**100**	93.34
	RBM +**fr-MKLR**	random splits	**100**	**98.59**
UIUC	*Hong et al.*[188]	random splits	-	93.3
	Lin et al.[189]	LOO	-	98.7
	Liu et al.[49]	LOO	-	93.5
	Wang et al.[5]	random splits	-	97.1
	LASSO	random splits	86.16	61
	KSVM	random splits	*95.98*	92.26
	MKLR	random splits	94.05	95.60
	RBM +**MKLR**	random splits	95.71	96.25
	fr-MKLR	random splits	**97.02**	*98.84*
	RBM +**fr-MKLR**	random splits	*96.61*	**99.46**
I3d actions	*Holte et al.*[190] 3D−MC	LOO	-	80
	– 3D−MC−mean	LOO	-	77.50
	– HMC	LOO	-	76.25
	– HMC−mean	LOO	-	68.75
	Spurlock et al.[191] RT	2-fold cross-validation	-	73.75
	– MC	2-fold cross-validation	-	**96.25**
	LASSO	random splits	77.56	67.20
	KSVM	random splits	93.64	72
	MKLR	random splits	80.91	68.80
	RBM +**MKLR**	random splits	94.40	88.68
	fr-MKLR	random splits	**98.91**	77.60
	RBM +**fr-MKLR**	random splits	94.40	90

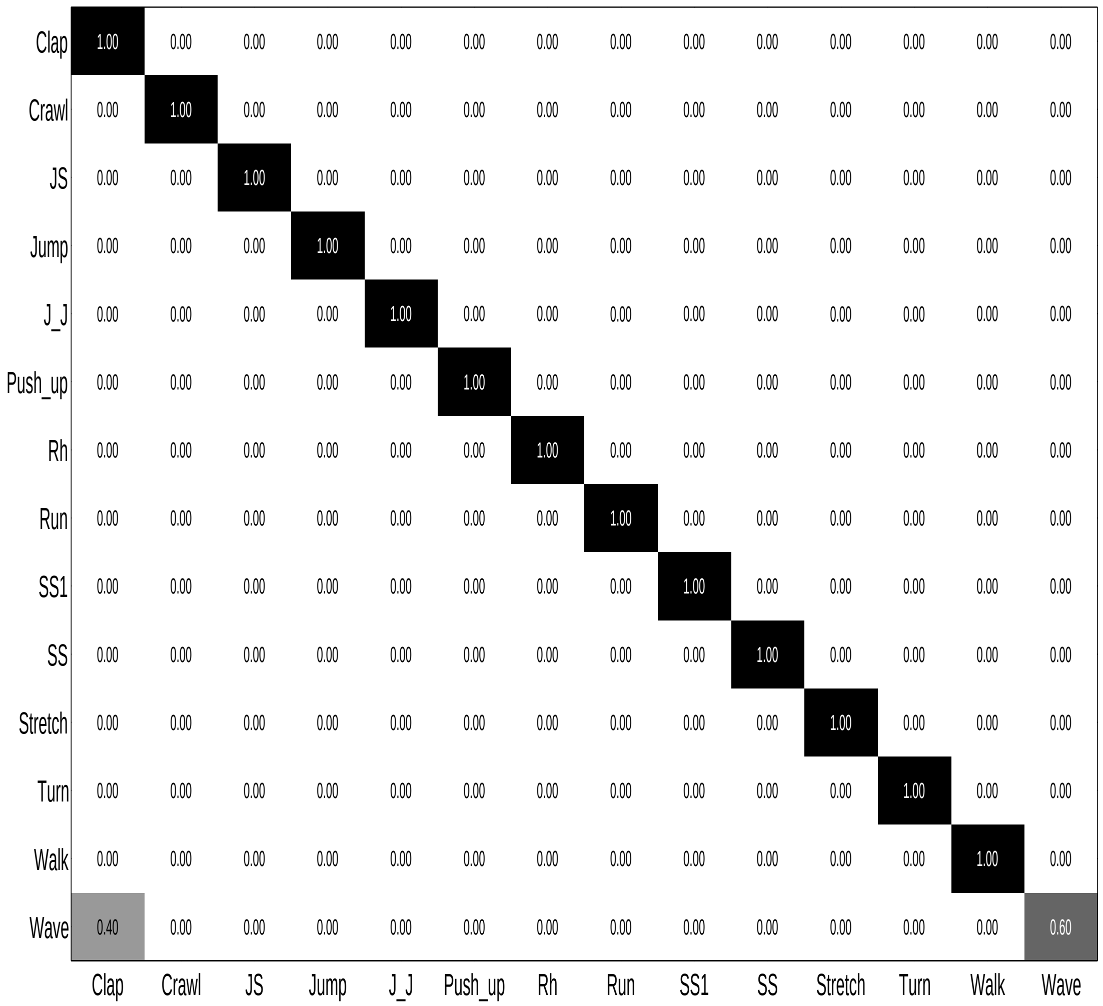
Figures 4.2.9, 4.2.10 and 4.2.11 show the best results obtained using ASC+fr-MKLR on the

KTH, UIUC and I3DPost datasets respectively. However, we can observe some confusion between actions in each confusion matrix. For example confusion between hand clap and hand wave on KTH and UIUC confusion matrix, and confusion between sit-stand-up and bend on I3DPost confusion matrix. This similarity can be explained by the similarity in pose variation when performing the action by different agents. Similarities between run-jump-walk, walk and walk-site and between run and run-full are due to the shared action between the composed action. Therefore, including these confusions we achieved a classification accuracies of 94.44% , 97.14% and 86% for KTH, UIUC and I3DPost respectively



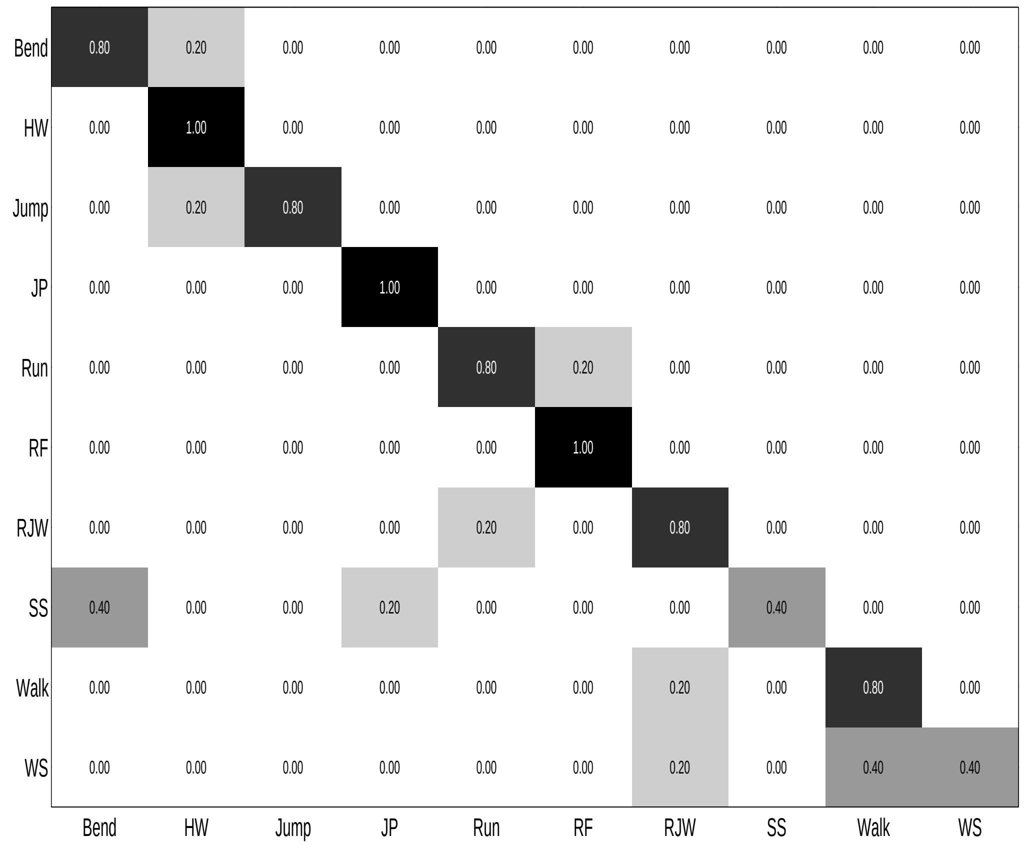
	Box	HC	HW	Jogg	Run	Walk
Box	1.00	0.00	0.00	0.00	0.00	0.00
HC	0.11	0.67	0.22	0.00	0.00	0.00
HW	0.00	0.00	1.00	0.00	0.00	0.00
Jogg	0.00	0.00	0.00	1.00	0.00	0.00
Run	0.00	0.00	0.00	0.00	1.00	0.00
Walk	0.00	0.00	0.00	0.00	0.00	1.00

Figure 4.2.9 – Confusion matrix showing results on the KTH dataset using ASC+fr-MKLR



	Clap	Crawl	JS	Jump	J_J	Push_up	Rh	Run	SS1	SS	Stretch	Turn	Walk	Wave
Clap	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Crawl	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
JS	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Jump	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
J_J	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Push_up	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rh	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Run	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
SS1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
SS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Stretch	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
Turn	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
Walk	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Wave	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.60

Figure 4.2.10 – Confusion matrix showing results on the UIUC dataset using ASC+fr-MKLR



	Bend	HW	Jump	JP	Run	RF	RJW	SS	Walk	WS
Bend	0.80	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
HW	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Jump	0.00	0.20	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00
JP	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Run	0.00	0.00	0.00	0.00	0.80	0.20	0.00	0.00	0.00	0.00
RF	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
RJW	0.00	0.00	0.00	0.00	0.20	0.00	0.80	0.00	0.00	0.00
SS	0.40	0.00	0.00	0.20	0.00	0.00	0.00	0.40	0.00	0.00
Walk	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.80	0.00
WS	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.40	0.40

Figure 4.2.11 – Confusion matrix showing results on the I3DPost dataset using ASC+fr-MKLR

4.3 Two-person interaction recognition

Daily activity do not constitute only in performing single actions, we are constantly interacting with objects and peoples. In group activity recognition, one must take into account all persons present in the scene and study individually their actions and their interactions with the environment (other persons and/or objects) [118, 119]. Many methods were suggested to describe two persons interaction, those descriptors are calculated from spatiotemporal interest points, SIFT, human pose, skeleton joints position and dense trajectories [120, 13, 127, 14, 124]. In this section we propose to recognize two persons interaction by analyzing gestures generated by each person over time. Since gestures are related to local motion of body parts, we build a representation based on body parts trajectories. These are extracted by tracking over time body joints position using the method proposed in [59]. We then combine shape and motion descriptors to build a feature group corresponding to each trajectory. Note that, contrarily to action recognition, no periodicity of movement is required here. Finally, based on the proposed interaction representation, we use GFR-MKLR to identify relevant groups of features, in this case the relevant trajectories, that discriminate between different interaction types. The different steps required for our interaction recognition approach are shown in Figure 4.3.1.

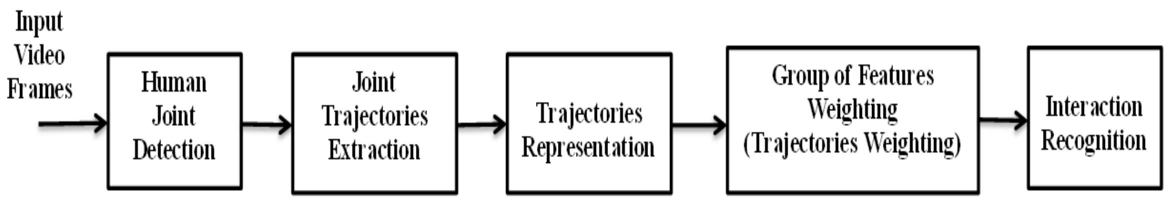

Input Video Frames
Human Joint Detection
Joint Trajectories Extraction
Trajectories Representation
Group of Features Weighting (Trajectories Weighting)
Interaction Recognition

Figure 4.3.1 – Our framework of human interaction recognition.

4.3.1 Preprocessing: Human joints extraction

This step is required in interaction recognition between two persons. We extract trajectories of body parts by tracking human joints over video frames using the algorithm proposed in [59].

In that algorithm, human pose estimation is performed using a representation based on the deformable part model [192]. To address problems related to limbs appearance variations due clothing, body chape, viewpoint and foreshortening, the authors propose using a mixture of non-oriented pictorial structures. This flexible mixture model jointly captures spatial relations between part locations and co-occurrence relations between parts, thus augmenting pictorial structure models that encode spatial relations. Examples of pose estimation are shown in Figure 4.3.2.

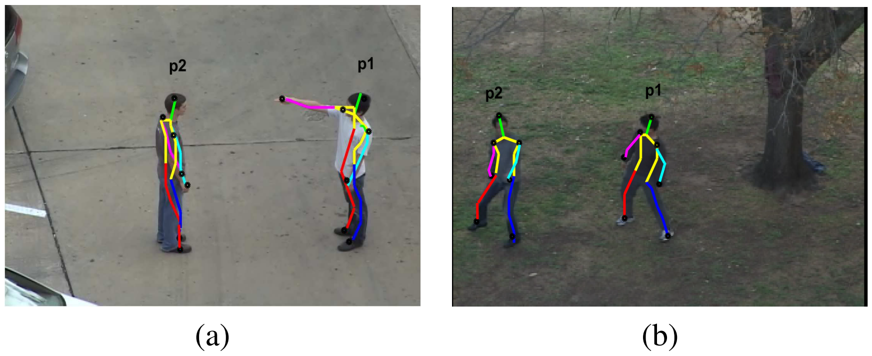


Figure 4.3.2 – An example of human pose estimation for hand wave and run actions. (a): point, (b): kick.

4.3.2 Trajectory extraction and representation

Instead of using dense trajectories, we propose to recognize two persons interaction by analyzing trajectories generated by the motion of human joints over time in two cases: 7 and 14 key joints per person in the video which gives 14 and 28 trajectories per interaction respectively.

To do this, for an input video V with T frames, we extract the trajectory for detected key points corresponding to a set of human joints $\mathbf{J}$. For this purpose, we track each joint over video frames $\mathbf{F}_t$, $t \in \{1, ..., T\}$ using the algorithm proposed in *Yang et al.*. In the case of 7 detected joints per person in the video, tracked joints are ordered as follow: *head (H), right shoulder (RS),right hand (RH), right foot (RF) ,left shoulder (LS), left hand (LH) and left* *foot (LF)*. Each detected joint is tracked over the video frames. Since we are dealing with two persons interaction we have a total of 14 joints at each frame, the concatenation of joints locations $(l_1, l_2, ..., l_T)$ with $l_t = (x_t, y_t)$ form the interaction trajectories $\mathbf{tr}_{J_i}$, $i \in \{1, ..., 14\}$ (7 trajectories per person).

To better study the influence of different body parts trajectories for discrimination between different interactions, we increase the number of joints by adding neck (N), right and left elbows (RE,LE), right and left hip (RHi, LHi), and knee joints for each person. This gives 28 trajectories in each video frame ((14 trajectories per person)) in the following order: *head (H), neck (N), right shoulder (RS),right elbow (RE), right hand (RH), right hip (RHi), right knee (RK), right foot (RF) ,left shoulder (LS), left elbow (LE) left hand (LH), left Hip (LHi), left knee (LK) and left foot (LF)*. In order to eliminate false joints detections over frames, we use a median filter to smooth the resulting trajectories. The static joints which give points or small trajectories are retained since the goal is to prove the contribution or not of a joint movement to discriminate between different interactions.

In Figure 4.3.3 we show examples of extracted trajectories for *punch*, *kick* and *point* interactions in both cases 14 and 28 trajectories per interaction.

From each trajectory $\mathbf{tr}_{J_i}$, two features are computed to describe their shape and motion. Given a trajectory of length $\mathbf{T}$, its shape is described by computing the displacement according x and y coordinate: $(\Delta l_1, \Delta l_2, ..., \Delta l_{T-1})$, with $\Delta l_t = (\Delta l_{x_t}, \Delta l_{y_t}) = (x_{t+1} - x_t, y_{t+1} - y_t)$. The final displacement vectors according to the coordinates x and y are normalized as follow:

$$D_{x,y} = \frac{(\Delta l_1, \Delta l_2, ..., \Delta l_{T-1})}{\sum_{j=1}^{T-1} \|\Delta l_j\|}, \tag{4.3.1}$$

The motion of each trajectory is described by a local curvature in space and time coordinates, respectively, $\mathbf{x}, \mathbf{y}$ **and** $\mathbf{t}$ [193]. The curvature $\mathbf{C_t}$ at each frame t is defined in Eq. (4.3.2)

$$\mathbf{C_t} = \frac{x_t^{'} y_t^{''} - y_t^{'} x_t^{''}}{(x_t^{'2} + y_t^{'2} + 1)^{3/2}}, \tag{4.3.2}$$

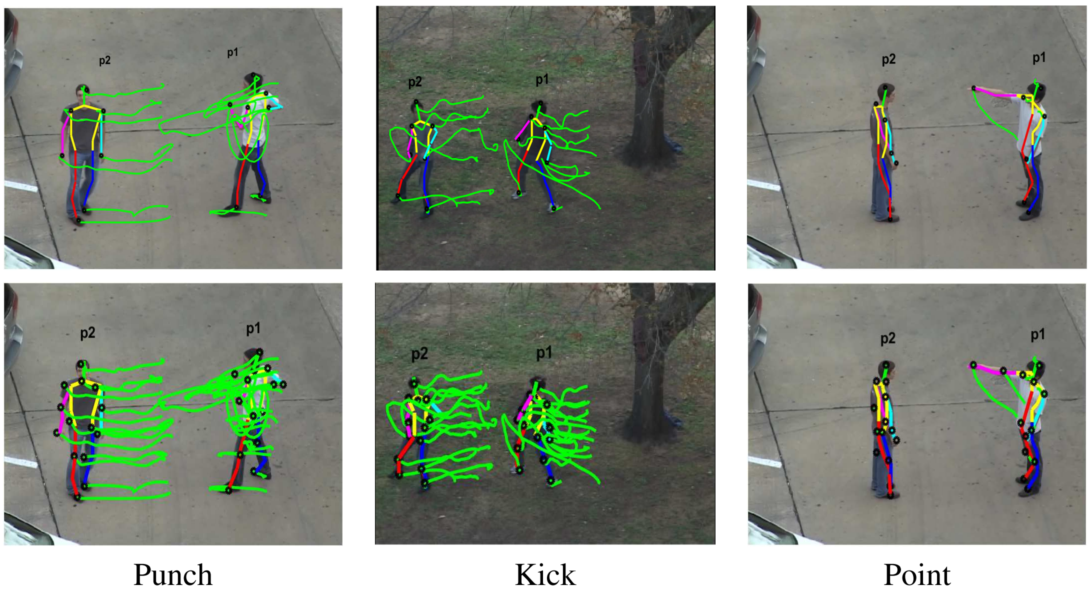


Figure 4.3.3 – Examples of extracted trajectories: 14 trajectories (first row) and 28 trajectories (second row).

with, $x_t^{'}, y_t^{'}, x_t^{''} and \;\; y_t^{''}$ are the first and second order temporal derivatives of the trajectory position, with: $x_t^{'} = \Delta l_{x_t}, y_t^{'} = \Delta l_{y_t}, x_t^{''} = \Delta x_t^{'}, y_t^{''} = \Delta y_t^{'} \; and \; \Delta t = 1$ knowing that the trajectories are extracted over successive frames.

For both shape and motion descriptors, histograms are generated by binning each vector into 10 bins. The normalized histogram of displacement HOD is then obtained by concatenating the histograms of D_x and D_y as follow $HOD = [HOD_x, HOD_y]$. The shape and the motion of the given trajectory is then described by a concatenation of normalized histogram of displacement and curvature to form a group of features: $[HOD, HOC]$. Finally, for each video two descriptors per trajectory are concatenated following a certain spatial order starting from right to left and from up to bottom to form the final interaction representation which is used to classify interactions. The final vector has the following structure:

$$[(HOD_{tr_H}, HOC_{tr_H})(p1), (HOD_{tr_N}, HOC_{tr_N})(p1), ..., (HOD_{tr_{LF}}, HOC_{tr_{LF}})(p1),$$
$$HOD_{(tr_H}, HOC_{tr_H})(p2), ..., (HOD_{tr_{LF}}, HOC_{tr_{LF}})(p2)].$$

p1 and p2 refers to person 1 and person 2 in a current video frame. Knowing that we have 10 bins per histogram, this give a description vector of 30 dimensions per trajectory (dimension of each group), therefore the final interaction vector is for 420 and 840 dimensions for 14 and 28 trajectories per interaction respectively.

4.3.3 Experimental results

To evaluate the performance of our method, we conducted experiments on the UT-interaction dataset [26] which has two sets of video data set I and set II. The video sequences in set I are taken in the parking with slightly different zoom rate and static background and Set II is slightly more challenging with some camera motion. this dataset contains six different classes of human-human interactions: *punch, kick, hand-shake, hug, points* and *push* (see Figure 4.3.4). As proposed in [26], for two interacting persons we use for each activity the first four sequences from set I and the first three sequences from set II. This gives 24 and 18 instances from sets I and set II, respectively, with an average number of 40 frames per video sequence. For each set, we follow the same procedure as for action recognition by randomly generated 5 groups for learning and 5 groups for testing and we average the obtained classification accuracy values.

Figure 4.3.4 – Examples of UT-interaction from set I (first row) and set II (second row).

The obtained average values of CA using UT-interaction dataset are given in Table 4.3.1 for compared methods. These include GFR-MKLR, MKLR and recent methods described in Chapter 2 section 2.3, namely: IP + DLN (deep learning network) [130], human joints + SVM [129], trajectories + pairwise kernels [126], STIP + SVM [26], HOG and HOF + MIL and SVM [122], 3D HOG + SVM [121], 3D SIFT + KNN and SVM [120], treajectories component + context kernel SVM [124], residual networks + LSTM [131] and dense trajectories + MIL [125].

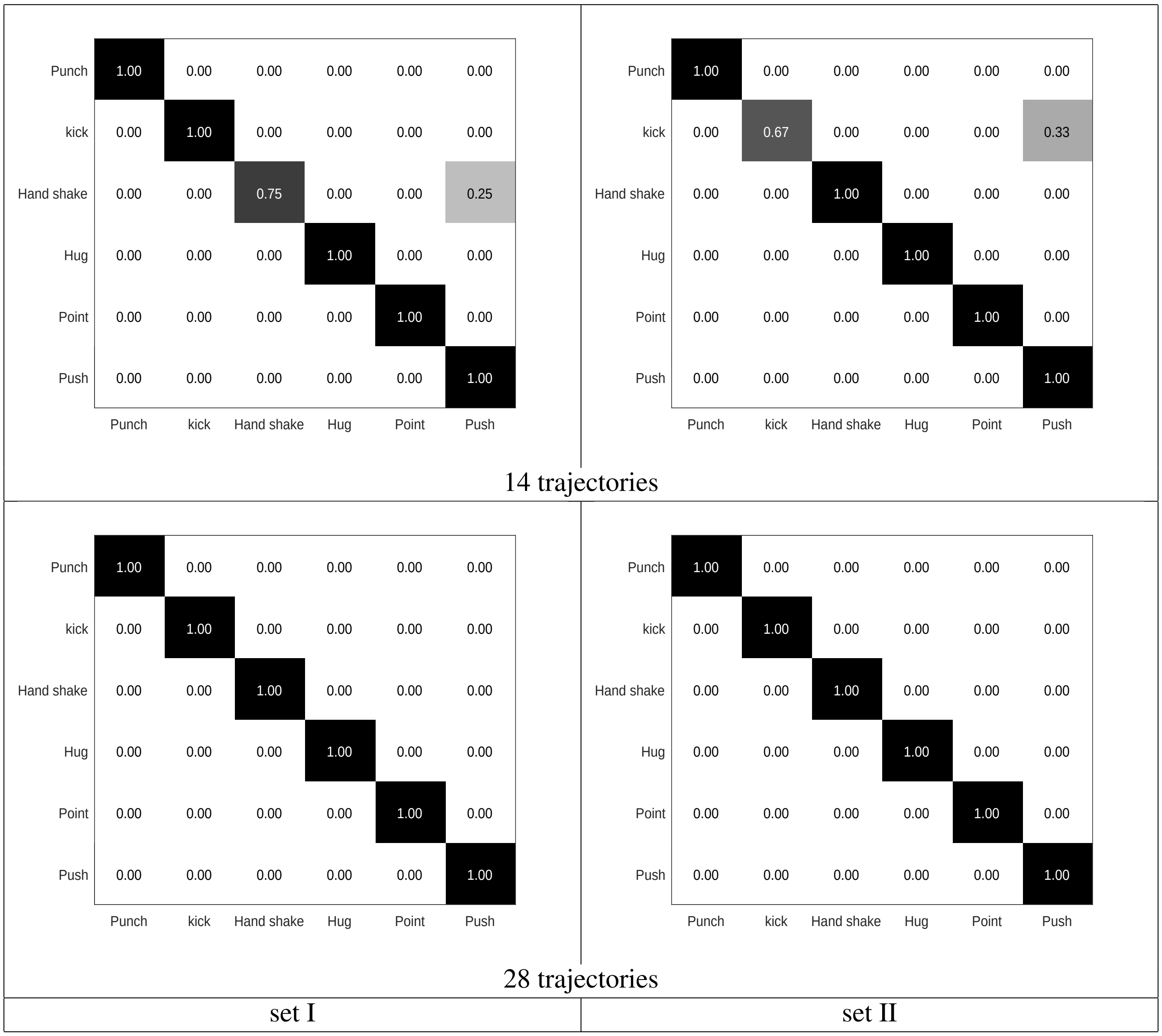
From the average values of CA, we can note that for set I if we take only the case of 14 trajectories, [131] obtained the highest performance and our method come in the second position. This is partially due to the high level features incorporating spatial and temporal features extracted using residual network and LSTM, however, besides result in [131] our method outperforms the sited methods for 14 trajectories even if we use simple representation but the use of GFR-MKLR increase the classification accuracy for set I and set II respectively. When we increase the number of trajectories per interaction, this add more relevant information to the descriptor and increase the discrimination between interaction classes, we obtain 100% accuracy, for both sets.

Figure 4.3.5 shows the best results obtained using our method on the UT-dataset. We can observe some classification errors between actions in each confusion matrix: between *hand* *shake* and *push* on Set1 confusion matrix, and confusion between *kick* and *push* on Set II confusion matrix when we use 14 trajectories to describe interaction. Those similarities can be explained by the similarity of body parts motion when acting and reaction by different persons in the sequences.

Since UT-interaction dataset contain 6 classes, the final weights vectors for 14 and 28 trajectories are of size of 14×5 and 28×5 respectively, obtained weights for *punch, kick, hand-shake, hug, points* in the two cases are represented in Figure 4.3.6 . From this figure we can note that our method has been able to attribute weights for the trajectory according to their discrimination between interactions.

Table 4.3.1 – Average classification accuracy obtained by compared methods for set I and set II from UT-dataset, (-) means values not reported in the original papers.

Methods	Evaluation method	Set I	Set II
Berlin and John [130]	-	95	88
Meng *et al.* [129]	10-fold LOOCV	91.81	83.6
Motiian *et al.* [126]	LOOCV	95.08	89.39
Ryoo *et al* [26]	percentage split	91.1	–
Sener *et al.* [122]	10-fold LOOCV	95	91.67
Sefidgar *et al.* [121]	10-fold LOOCV	93.3	91
Slimani *et al.* [120]	5-fold CV	40.63	66.67
Yuan *et al.* [124]	10-fold LOOCV	78.3	68.2
Zhao *et al.* [131]	random split	*98.33*	–
Zhang *et al.* [125]	LOOCV	76	78
Kernel logistic regression	random split	87.5	83.33
Our method (14 trajectories)	random split	95.8	*94.44*
Our method (28 trajectories)	random split	**100**	**100**



14 trajectories, set I

	Punch	kick	Hand shake	Hug	Point	Push
Punch	1.00	0.00	0.00	0.00	0.00	0.00
kick	0.00	1.00	0.00	0.00	0.00	0.00
Hand shake	0.00	0.00	0.75	0.00	0.00	0.25
Hug	0.00	0.00	0.00	1.00	0.00	0.00
Point	0.00	0.00	0.00	0.00	1.00	0.00
Push	0.00	0.00	0.00	0.00	0.00	1.00

14 trajectories, set II

	Punch	kick	Hand shake	Hug	Point	Push
Punch	1.00	0.00	0.00	0.00	0.00	0.00
kick	0.00	0.67	0.00	0.00	0.00	0.33
Hand shake	0.00	0.00	1.00	0.00	0.00	0.00
Hug	0.00	0.00	0.00	1.00	0.00	0.00
Point	0.00	0.00	0.00	0.00	1.00	0.00
Push	0.00	0.00	0.00	0.00	0.00	1.00

28 trajectories, set I

	Punch	kick	Hand shake	Hug	Point	Push
Punch	1.00	0.00	0.00	0.00	0.00	0.00
kick	0.00	1.00	0.00	0.00	0.00	0.00
Hand shake	0.00	0.00	1.00	0.00	0.00	0.00
Hug	0.00	0.00	0.00	1.00	0.00	0.00
Point	0.00	0.00	0.00	0.00	1.00	0.00
Push	0.00	0.00	0.00	0.00	0.00	1.00

28 trajectories, set II

	Punch	kick	Hand shake	Hug	Point	Push
Punch	1.00	0.00	0.00	0.00	0.00	0.00
kick	0.00	1.00	0.00	0.00	0.00	0.00
Hand shake	0.00	0.00	1.00	0.00	0.00	0.00
Hug	0.00	0.00	0.00	1.00	0.00	0.00
Point	0.00	0.00	0.00	0.00	1.00	0.00
Push	0.00	0.00	0.00	0.00	0.00	1.00

Figure 4.3.5 – Confusion matrix showing results in set I and set II for 14 and 28 trajectories from UT-interaction dataset.

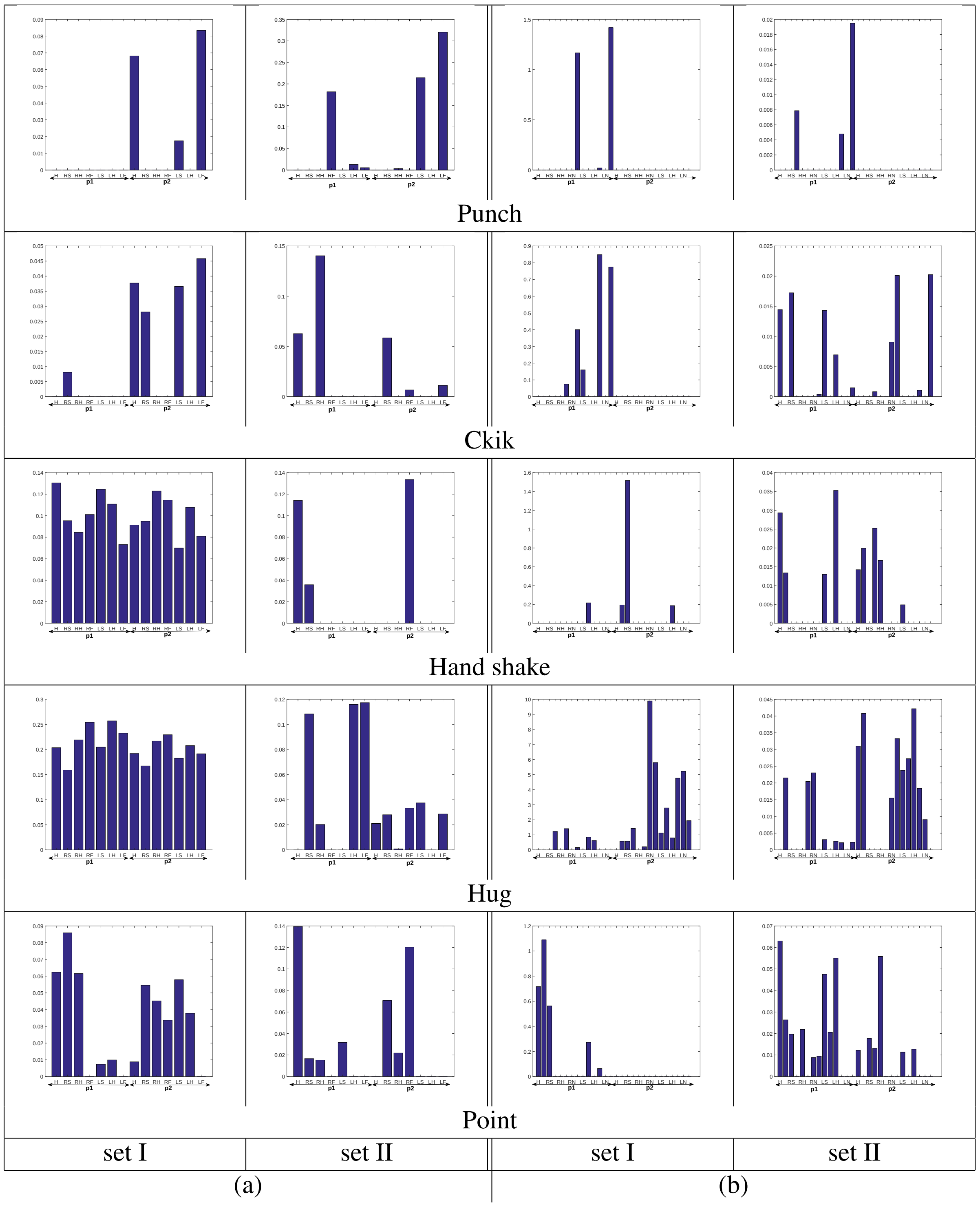


Figure 4.3.6 – Obtained weights for set I and set II using : (a) 14 trajectories and (b) 28 trajectories respectively per interaction.

4.4 Conclusion

This chapter has been devoted to the recognition of actions and interactions between two persons in a video sequences. In the first part, we have applied FR-MKLR model for action recognition by exploiting a new representation named *action shape context* (ASC) which is based on spatiotemporal shape context analysis of silhouettes. Experiments using standard datasets such as KTH , UIUC and the I3DPost have shown that combining ASC and fr-MKLR gives good results that outperform other methods such as MKLR, KSVM and LASSO and other state-of-art action recognition methods. In the second part of this chapter, we have investigated another part of human activity which involves two-persons iterations. To recognize human interactions, we study the shape and the motion of human joints trajectories to create a new descriptor divided into groups of features corresponding to body parts motion. We then applied the GFR-MKLR model to classify interactions between two persons. Results on the UT-interaction dataset have showed the effectiveness of our method with 100% accuracy achieved when increasing the number of trajectories per interaction.

Chapter 5

Conclusion and discussion

In this thesis, we addressed the problem of human activity recognition in a video sequences. Specifically, we considered the case of single action and two persons interaction recognition. We focused on two main problems related to the representation and classification which are important steps in the recognition process. Firstly, due to the high-dimensionality problem, we proposed a sparse model (fr-MKLR) incorporating directly the relevance of individual features into the classification process, and we have successfully apply it to recognize single actions even in datasets with a limited number of training data. Secondly, we generalized our approach to promote sparsity at the level of groups of features. This model is used to recognize human interactions in video sequences. We give details of each of these contributions as follows:

In the first contribution, we have proposed the FR-MKLR model incorporating feature relevance in multinomial kernel logistic regression. It consists of using an anisotropic kernel embedding weights controlling feature contribution in classification. These weights are estimated along the other parameters using training data. Obtained sparse models are less prone to overfitting and enable better classification generalization in case scarce training data and the presence of redundant features. Experiments on simulated data as well as standard UCI datasets have shown that the proposed approach outperforms several standard methods such as naive Bayes, MKLR, SMLR, KSVM and LASSO. We then applied fr-MKLR for single action recognition using action shape context (ASC) to represent actions. The ASC uses the gravity center of object silhouettes to calculate a shape representation which is invariant to affine transformations and scale, and allow to capture global and local motion information for discriminating actions. Experiments have shown that combining ASC and FR-MKLR outperforms state-of-art classification methods in terms of classification accuracy. Note that further ASC refinement using restricted Boltzman machines (RBM) have enabled to boost the classification accuracy in the tested datasets.

In the second contribution, We have generalized FR-MKLR by incorporating group of features sparsity in multi-class kernel logistic regression (GFR-MKLR). The need for group sparsity arises in several practical situations where a subset of factors can explain a predicted output and each factor consists of a group of variables/features. This is reflected in human interactions where motion patterns body parts can be organised into groups corresponding to body parts gestures(e.g., hands, legs, head, etc.). By describing interactions body parts trajectories, we applied the GFR-MKLR model for better discrimination of different interactions according to the involved gestures. Experiments conducted on the UT-Interaction dataset have demonstrated the performance of our method with regard to stat-of-art methods in terms of classification accuracy.

The efficiency of the proposed models is particularly demonstrated for non-linear separable classes in case of scarce training data and the presence of redundant features. However, as for any kernel-based method, when the number of training data is very large, computational time becomes a limitation since the formulation of the models incurs kernel matrix inversions. To be able to apply it in this case, data reduction using sampling or clustering may be necessary preprocessing step before classification. Also, for linearly separable data (e.g., text classification), FR-MKLR, GFR-MKLR and its competitors MKLR and KSVM, can be less efficient than their linear counterparts. Finally, we must put on emphasis the performance obtained by combining RBM with FR-MKLR in action classification. Indeed, the majority of deep learning methods apply a softmax function (e.g., MKLR) on the top of the network for generating class probabilities. Given the obtained performance by FR/GFR-MKLR, a promising pursuit in the future will be to use FR/GFR-MKLR as an alternative for softmax functions in classification problems using deep learning architectures. Finally, wether for single actions or interaction recognition, further tests of our approach on more complex databases presenting more challenges, such as videos captured in public places with crowded scenes, to show the performance of our models and their limitations as well.

Chapter 6

APPENDIX

6.1 Derivation of the NLL in Eq. (3.3.8)

Using the posterior probabilities in Eq. (3.3.7), we have:

$$\begin{aligned}\mathcal{L}(\mathcal{A}) &= -\sum_{i=1}^{n}\left(\sum_{j=1}^{m-1} y_i^{(j)} \ln\left[\frac{\exp(f_j(\mathbf{x}_i))}{1+\sum_{h=1}^{m-1}\exp(f_h(\mathbf{x}_i))}\right]\right. \\ &\quad \left. + \left(1-\sum_{j=1}^{m-1} y_i^{(j)}\right)\ln\left[\frac{1}{1+\sum_{h=1}^{m-1}\exp(f_h(\mathbf{x}_i))}\right]\right) + \frac{\lambda}{2}\sum_{j=1}^{m-1}\mathbf{a}^{(j)T}\mathbf{K}\mathbf{a}^{(j)} \quad (6.1.1)\end{aligned}$$

$$= \sum_{i=1}^{n}\left(\sum_{j=1}^{m-1} -y_i^{(j)} f_j(\mathbf{x}_i) + \frac{\lambda}{2}\mathbf{a}^{(j)T}\mathbf{K}\mathbf{a}^{(j)} + \ln\left[1+\sum_{h=1}^{m-1}\exp(f_h(\mathbf{x}_i))\right]\right), \quad (6.1.2)$$

By defining $\mathbf{y}^{(j)} = [y_1^{(j)}, y_2^{(j)}, ..., y_n^{(j)}]^T$ and using the compact form:

$$\ln\left[1+\sum_{h=1}^{m-1}\exp(\mathbf{K}\mathbf{a}^{(h)})\right] = \begin{pmatrix} \ln\left[1+\sum_{h=1}^{m-1}\exp(\mathbf{K}(1,.)\mathbf{a}^{(h)})\right] \\ \ln\left[1+\sum_{h=1}^{m-1}\exp(\mathbf{K}(2,.)\mathbf{a}^{(h)})\right] \\ \vdots \\ \ln\left[1+\sum_{h=1}^{m-1}\exp(\mathbf{K}(n,.)\mathbf{a}^{(h)})\right] \end{pmatrix}, \quad (6.1.3)$$

where $\mathbf{K}(i,.)$, $i \in \{1,...,n\}$, denotes the ith row of $\mathbf{K}$, we obtain Eq. (3.3.8).

6.2 Derivation of Eqs. (3.3.13) and (3.3.14)

Given the following derivation:

$$\frac{\partial \ln\left[1+\exp(\tilde{\mathbf{K}}(i,.)\mathbf{a})\right]}{\partial \mathbf{a}} = \frac{\exp(\tilde{\mathbf{K}}(i,.)\mathbf{a})}{1+\exp(\tilde{\mathbf{K}}(i,.)\mathbf{a})}\tilde{\mathbf{K}}(i,.)^T = p_i\tilde{\mathbf{K}}(.,i), \tag{6.2.1}$$

we have: $\partial\mathcal{L}/\partial\mathbf{a} = (-\tilde{\mathbf{K}}\mathbf{y} + \tilde{\mathbf{K}}\mathbf{p} + \lambda\tilde{\mathbf{K}}\mathbf{a}) = \tilde{\mathbf{K}}\mathbf{c}$. Using Eq. (3.3.10) and matrix derivation properties, we have $\forall k \in \{1,...,d\}$:

$$\mathbf{Q}_k = \frac{\partial\tilde{\mathbf{K}}}{\partial\psi_k} = \begin{pmatrix} \frac{\partial\tilde{\mathcal{K}}(\mathbf{x}_1,\mathbf{x}_2)}{\partial\psi_k} & \cdots & \frac{\partial\tilde{\mathcal{K}}(\mathbf{x}_1,\mathbf{x}_n)}{\partial\psi_k} \\ \vdots & \ddots & \vdots \\ \frac{\partial\tilde{\mathcal{K}}(\mathbf{x}_n,\mathbf{x}_1)}{\partial\psi_k} & \cdots & \frac{\partial\tilde{\mathcal{K}}(\mathbf{x}_n,\mathbf{x}_n)}{\partial\psi_k} \end{pmatrix} = \tilde{\mathbf{K}} \circ \mathbf{B}_k. \tag{6.2.2}$$

Therefore, we have: $\partial\mathcal{L}/\partial\psi_k = \mathbf{c}^T\mathbf{Q}_k\mathbf{a} + \mu\beta\exp(-\beta\psi_k)$, where $\mathbf{B}_k$ is the matrix defined in Section 3.6. By gathering the elements $\partial\mathcal{L}/\partial\psi_k$ in one vector, we obtain Eq. (3.3.14).

6.3 Derivation of Eq. (3.6.6)

First, we have $\partial p_i/\partial\mathbf{a} = p_i(1-p_i)\tilde{\mathbf{K}}(i,.)$. Then, the Hessian of the NLL with respect to the elements of $\mathbf{a}$ is given by:

$$\mathbf{H} = (\mathbf{KWK} + \lambda\tilde{\mathbf{K}}), \tag{6.3.1}$$

where $\mathbf{W} = \text{diag}[p_1(1-p_1), p_2(1-p_2), ..., p_n(1-p_n)]$. Also, note that $\forall k, \ell \in \{1, ..., d\}$ and $\forall i \in \{1, ..., n\}$:

$$\frac{\partial^2 \tilde{\mathbf{K}}}{\partial \psi_k \partial \psi_\ell} = \begin{pmatrix} \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_1,\mathbf{x}_2)}{\partial \psi_k \partial \psi_\ell} & \cdots & \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_1,\mathbf{x}_n)}{\partial \psi_k \partial \psi_\ell} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_n,\mathbf{x}_1)}{\partial \psi_k \partial \psi_\ell} & \cdots & \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_n,\mathbf{x}_n)}{\partial \psi_k \partial \psi_\ell} \end{pmatrix}, \tag{6.3.2}$$

$$\frac{\partial^2 \tilde{\mathbf{K}}}{\partial a_i \partial \psi_k} = \begin{pmatrix} \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_1,\mathbf{x}_2)}{\partial a_i \partial \psi_k} & \cdots & \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_1,\mathbf{x}_n)}{\partial a_i \partial \psi_k} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_n,\mathbf{x}_1)}{\partial a_i \partial \psi_k} & \cdots & \frac{\partial^2 \tilde{\mathcal{K}}(\mathbf{x}_n,\mathbf{x}_n)}{\partial a_i \partial \psi_k} \end{pmatrix}. \tag{6.3.3}$$

Therefore, we can build the matrices $\mathbf{T}$ and $\mathbf{M}$ containing the mixed derivatives as follows:

$$\mathbf{T}_{k\ell} = \frac{\partial^2 \mathcal{L}}{\partial \psi_k \partial \psi_\ell} = \begin{cases} [\mathbf{Q}_k \mathbf{W} \mathbf{a}]^T \mathbf{Q}_k \mathbf{a} + \mathbf{c}^T \mathbf{S}_k \mathbf{a} - \mu \beta^2 \text{exp}(-\beta \psi_k) & if \quad k = \ell \\ [\mathbf{Q}_\ell \mathbf{W} \mathbf{a}]^T \mathbf{Q}_k \mathbf{a} + \mathbf{c}^T (\mathbf{Q}_k \circ \mathbf{B}_\ell) \mathbf{a} & if \quad k \neq \ell \end{cases} \tag{6.3.4}$$

$$\mathbf{M}_{ik} = \frac{\partial^2 \mathcal{L}}{\partial a_i \partial \psi_k} = \mathbf{Q}_k(i,.)(-\mathbf{y} + \mathbf{p} + \lambda \mathbf{a}) + \tilde{\mathbf{K}}(i,.)\mathbf{Q}_k \mathbf{W} \mathbf{a} \tag{6.3.5}$$

where $\mathbf{S}_k = \tilde{\mathbf{K}} \circ (\mathbf{D}_k + \mathbf{B}_k \circ \mathbf{B}_k)$ and $\mathbf{D}_k$ is an $n \times n$ matrix where $\mathbf{D}_k(r,s) = -(x_{r,k} - x_{s,k})^2$. By putting together the elements of $\mathbf{H}$, $\mathbf{M}$ and $\mathbf{T}$, we obtain the Hessian of Eq. (3.6.6).

Bibliography

[1] I. Laptev and T. Lindeberg. Velocity adaptation of space time interest points. *IEEE International Conference on Pattern Recognition*, pages 52–56, 2004.

[2] P. Dollar, V. Rabaud, G. Cotrrel, and S.Belongie. Behavior recognition via sparse spatio-temporal features. *IEEE International Workshop on Visual Survaillance and Performance Evaluation of Tracking and Survaillance*, pages 65–72, 2005.

[3] S.F Wong and R. Cipolla. Extractiong spatiotemporal interest points using global information. *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[4] M. Bregonzio, S. Gong, and T. Xiang. Recognising action as clouds of space-time interest points. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2009.

[5] H. Wang, A. Klaser, C. Schmid, and C.L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103(1):60–79, 2013.

[6] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov model. *IEEE Conference on Conputer Vision and Pattern Recognition*, pages 379–385, 1992.

[7] A. F. Bobick and J. W. Davis. The recognition of human mouvement using temporal templates. *IEEE Transaction on Pattern Analysis and Machine Inteligence*, 23(3):257–267, 2001.

[8] A. Efros, A.C. Berg, G. Mori, and J. Malik. Rcognizing action at a distance. *IEEE International Conference on Computer Vision*, pages 726–733, 2003.

[9] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104 (2-3):249–257, 2006.

[10] M. Blank, L. Gorelik, M. Shechtman, and R. Basri. Actions as space-time shape. *IEEE International Conference on Computer Vision*, pages 1395–1402, 2005.

[11] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013.

[12] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequentiel deep learning for human action reognition. *International Workshop on Human Behavior Understanding*, pages 29–39, 2011.

[13] N.G. Cho, S.H. Park, J.S. Park, U. park, and S.W. Lee. Compositional interaction descriptor for human interaction recognition. *Neurocomputing*, 267:169–181, 2017.

[14] C. J. van Gemeren, R. T. Tan, R. W. Poppe, , and R. C. Veltkamp. Dyadic interaction detection from pose and flow. *International Workshop on Human Behavior Understanding*, 8749:101–115, 2014.

[15] J. Liang, C. Xu, Z. Feng, and X. Ma. Affective interaction recognition using spatio-temporal features and context. *Computer Vision and Image Understanding*, 144:155–165, 2016.

[16] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[17] W. Lin, M.-T. Sun, R. Poovendran, and Z. Zhang. Human activity recognition for video surveillanc. *IEEE International Symposium on Circuits and Systems*, pages 2737–2740, 2008.

[18] S.-R. Ke, H.L.U. Thuc, Y.-J. Lee, J.-N. Hwang, J.-H. Yoo, and K.-H. Choi. A review on video-based humain activity recognition. *Computers2013*, 2(2):88–131, 2013.

[19] X. Zabulis, H. Baltzakis, and A. Argyross. Vision-based hand gesture recognition for human-computer interaction. *In The Universal Access Handbook, LEA*, 2009.

[20] Y. Chen, R. Parent, R. Machiraju, and J. Davis. Human activity recognition for synthesis. *IEEE CVPR Workshop on Learning, Representation and Context for Human Sensing in Video*, 2006.

[21] A. F. Bobick and A. D. Wilson. A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337, 1997.

[22] S. Kumari and S.k. Mitra. Human action recognition using dft. *IEEE National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics*, pages 239–242, 2011.

[23] M. Vrigkas, C. Nikou, and I.A. Kakadiaris. A review of human activity recognition methods. *Frontiers in Robotics and AI*, 2:1–28, 2015.

[24] J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43(3), 2011.

[25] K. Yun, J. Honorio, D. Chattopadhyay, T.L. Berg, and D. Samaras. Two-person interaction detection using body-pose features and multiple instance learning. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 728–35, 2012.

[26] M. S. Ryoo and J. K. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1593–1600, Oct 2009.

[27] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28:976–990, 2010.

[28] P.K. Turaga, R. Chellappa, V.S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1473–1488, 2008.

[29] H. Shah and N.C. Chauhan. Recognition of human action in video. *International Journal on Recent Innovation Trends in Computing and Communication*, 1(5):489–493, 2013.

[30] N. C. Krishnan, C. Juillard, D. Colbry, and S. Panchanathan. Recognition of hand movements using wearable accelerometers. *Journal of Ambient Intelligence and Smart Environments*, 1(2):143–155, 2009.

[31] F. Alireza and G. Mori. Action recognition by learningmid-level motion features. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[32] C. Schuldt, I. Laptev, and B. Caputog. Recognizing human actions: A local svm aproach. *IEEE International Conference on Pattern Recognition*, pages 32–36, 2004.

[33] P. Scovanner, S. Ali, and M. Sha. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360, Sep 2007.

[34] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[35] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[36] G. Mori and J. Malik. Estimating human body configuration using shape context. *European Conference on Computer Vision*, pages 666–680, 2002.

[37] T. Guha and R.K. Ward. Learning sparse representations for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Inteligence*, 38(2):1576–1588, 2012.

[38] S.R. Fanello, I. Gori, G. Meta, and F. Odone. Keep it simple and sparse: Real-time action recognition. *Journal of Machine Learning Research*, 14:2617–2640, 2013.

[39] A.Y. Yang, A. Ganesh, and S.S. Sastry. Face recognition via sparse representation. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2008.

[40] X. Sun, M. Chen, and A. Hauptmam. Action recognition via local and holistic features. *IEEE Conputer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 58–65, 2009.

[41] Z. Zhao and A. Elgammal. Human activity recognition from frame's spatiotemporal representation. *19th International Conference on Pattern Recognition*, pages 1–4, 2008.

[42] A. Kläser, M. Marszalek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. *19th British Machine Vision Conference*, pages 1–10, 2008.

[43] Y. G. Jiang, Q. Dai, W. Liu, X. Xue, and C. w. Ngo. Human action recognition in unconstrained videos by explicit motion modeling. *IEEE Transactions Om Image Processing*, 24(11):3781–379, 2015.

[44] R. Rosales. Recognition of human action using moment-based features. Technical Report BUCS/1998-020, Computer Science Department, Boston University, Boston, 1998.

[45] L. Wang and D. Suter. Visual learning and recognition of sequential data manifolds with applications to human movement analysis. *Computer Vision and Image Understanding*, 110(2):153–172, 2008.

[46] J. Blackburn and E. Ribeiro. Human motion recognition using isomap and dynamic time wraping. *Human Motiom: Understanding, Modeling, Capture and Animation*, pages 285–298, 2007.

[47] T. J. Chin, L. Wang, and D. Suter. Extrapolating learned manifolds for human activity recognition. *IEEE International Confference on Image Processing*, pages 381–384, 2007.

[48] K. Huang and S. Aviyente. Sparse representation for signal classification. *Neural Information Processing Systems*, pages 609–616, 2006.

[49] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attribute. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3337–3344, 2011.

[50] An-An. Liu, N. Xu, Y. T. Su, H. Lin, T. Hao, and Z. X. Yang. Single/multi-view human action recognition via regularized multi-task learning. *Neurocomputing*, 115(2):544–553, 2015.

[51] V. Roth. The generalized lasso. *IEEE Transactions on Neural Networks*, 15(1):16–28, 2004.

[52] R. Tibshirani. Regression srinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[53] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68(1):49–67, 2006.

[54] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman and Hall/CRC Monographs on Statistics and Applied Probability, 2015.

[55] Y. Yuan, L. Qi, and X. Lu. Action recognition by joint learning. *Image and Vision Computing*, 55:77–85, 2016.

[56] N. Raman and S.J. Mayabank. Activity recognition using a supervised non-parametric hierarchical hmm. *Neurocomputing*, 199:163–177, 2016.

[57] L. Bazzani, M. Zanotto, and M. Cristaniand V. Murino. Joint individual-group modeling for tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(4):746–759, 2015.

[58] A. Boulmerka and M.S. Allili. Background modeling in videos revisited using finite mixtures of generalized gaussians and spatial information. *IEEE International Conference on Image Processing*, pages 3660–3664, 2015.

[59] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2878–2890, 2013.

[60] D. Dua and K. T. Efi. UCI machine learning repository. http://archive.ics.uci.edu/ml.

[61] N. Ghalelis, H. Kim, A. Hilton, N. Nikolaidis, and I. Pitas. The i3dpost multi-view and 3d human action/interaction database. *IEEE Conference for Visual Media Production*, pages 159–168, 2009.

[62] S.A. Vahora and N.C. Chauhan. A comprhensive study of group activity recognition methods in video. *Indian Journal of Science and Technology*, 10(23):1–11, 2017.

[63] D.G. Lowe. Object recognition from local scal-invariant features. *IEEE International Conference on Computer Vision ICCV*, pages 1150–1157, 1999.

[64] I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[65] T. Lindeberg. *Scale-Space Theory in Computer Vision.* Kluwer Academic Publisher/Boston/London/Dordrecht, Royal Institut of Technology, Stockholm, Sweden, 1993.

[66] J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3):299–318, 2008.

[67] A. Oikonomopoulos, I. Patras, and M. Pantic. Spatiotemporal salient points for visual recognition of human actions. *IEEE Transactions On Systems Man And Cybernetics*, 36(3):710–719, 2006.

[68] G. Willems, T. Tuytelaars, and L.J. Van. An efficient dense and scale invariant spatio-temporel interest point detector. *European Conference on Computer Vision*, pages 650–663, 2008.

[69] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[70] L. Zelnik-Manor and M. Irani. Statistical analysis of dynamic actions. *IEEE Transactions on Pattern Analysis and Machine Inteligence*, 28(9):1530–1535, 2006.

[71] L. Campbell and A. Bobick. Recognition of human body motion using phase space constraints. *IEEE International Conference on Computer Vision*, pages 624–630, 1995.

[72] X. Wu J. Sun, S. Yan, L. f. Cheong, T. S. Chua, and J. Li. Hierarchical spatio-temporal context modeling for action recognition. *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2004–2011, 2009.

[73] R. Messing, C. Pal, and H. Kautz. Activity recognition using the velocity histories of tracked keypoints. *IEEE International Conference on Computer Vision*, pages 104–111, 2009.

[74] A. Haiam and E. H Elsayed. Human action recognition using trajectory-based representation. *Egyptian Informatics Journal*, 16:187–198, 2015.

[75] J. W. Davis. Hierarchical motion history images foe recognizing human motion. *IEEE Workshop on Detection and Recognition of Events in Video*, pages 39–46, 2001.

[76] L. Shao, L. Ji, Y. Liu, and J. Zhang. Human action segmentation and recognition via motion and shape analysis. *Pattern Recognition*, 33(4):438–445, 2012.

[77] S. Ali and M. Shah. Human action recognition in videos using kinematic features and multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(2):288–303, 2010.

[78] S. Danafar and N. Gheissari. Action recognition for surveillance application using optical flow and svm. *Asian Conference on Computer Vision*, pages 457–466, 2007.

[79] N. Ikizler, R. Gokberk, and P. Duygulu. Human action recognition with line and flow histograms. *19th International Conference on Pattern Recognition*, pages 1–4, 2008.

[80] N. Kholgade and A. Savakis. Human activity recognition in video using two methods for matching shape context of silhouettes. In *Proceedings of Inteligent Computing: Theory and Application VI*, pages 1–8, 2008.

[81] M. Grundmann, F. Meier, and I. Essa. 3d shape context and distance transform for action recognition. *IEEE Conference on Pattern Recognition*, pages 1–4, 2008.

[82] N. Kholgade and A. Savakis. Human activity recognition using the 4d spatiotemporal shape context descriptor. *International Symposium on Advances in Visuel Computing*, pages 357–366, 2009.

[83] A. Yilmaz and M. Shah. A differential geometric approach to representing the human actions. *Computer Vision and Image Understanding*, 109(3):335–351, 2008.

[84] J. Wang, Z-Q. Zhao, X. Hu, Y-M. Cheung, M. Wang, and X. Wu. Online group feature selection. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pages 1757–1763, 2013.

[85] L. Xia, Chia-Chih. Chen, and J. K. Aggarwal. View invariant human action recognition using histograms of 3d joints. *IEEE Conferenc on Computer Vision and Pattern Recognition Workshops*, pages 1–8, 2012.

[86] G. Evangelidis, G. Singh, and R. Horaud. Skeletal quads: Human action recognition using joint quadruples. *IEEE International Conference on Pattern recognition*, pages 4513–4518, 2014.

[87] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. *IEEE Conference on Computer Vision and Pattern recognition*, pages 1290–1297, 2012.

[88] B. X. Nie, C. Xiong, and Song-Chun. Zhu. Joint action recognition and pose estimation from video. *IEEE Conference on Computer Vision and Pattern recognition*, pages 1293–1301, 2015.

[89] D. Wu, N. Sharma, and M. Blumenstein. Recent advances in video-based human action recognition using deep learning: A review. *International Joint Conference on Neural Networks*, pages 2865–2870, 2017.

[90] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[91] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, http://www.deeplearningbook.org, 2016.

[92] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep con-

volutional neural networks. *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[93] R. Ranganath A.Y. NG H. Lee, R. Grosse. Convolutional deep belief networks for scalable unsupervised learning of hierarachical representations. In *ICML '09 Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616, June 2009.

[94] M. Asadi-Aghbolaghi, M. Bellantonio A. Clapés, H.J. Escalante, V. Ponce-López, X. Bar'o, I. Guyon, S. Kasaei, and S. Escalera. A survey on deep learning approaches for action and gesture recognition in image sequences. *IEEE 12th International Conference on Automatic Face and Gesture recognition*, pages 539–578, 2017.

[95] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. *IEEE Conference on Computer Vision*, pages 4489–4497, 2015.

[96] Z. Liu, C. Zhang, and Y. Tian. 3d-based deep convolutional neural networks for action recognition. *Image and Vision Computing*, 55(2):93–100, 2016.

[97] G. Varol, I. Laptev, and C. Schmid. Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1510–1517, 2018.

[98] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4305–4314, 2015.

[99] M. Ravanbakhsh, H. Mousavi, M. Rastegari, V. Murino, and L.S. Davis. Action recognition with image based cnn features. *arXiv preprint arXiv:1512.03980*, pages 1–10, 2015.

[100] B. Banerjee and V. Murino. Efficient pooling of image based cnn features for action recognition in videos. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2637–2641, 2017.

[101] C. Kai, D. Guiguang, and H. Jungong. Attribute-based supervised deep learning model for action recognition. *Frontiers of Computer Science*, 11(2):219–229, 2017.

[102] K. Charalampous and A. Gasteratos. Online deep learning method for action recognition. *Pattern Analysis and Applications*, 19(2):337–354, 2016.

[103] G. Guo and A. Lai. A survey on still image based human action recognition. *Pattern Recognition*, 47:3343–3361, 2014.

[104] V.N. Vapnik. *Statistical Learning theory*. John Wiley & Sons, United States of America, 1998.

[105] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[106] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and Ronen Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, 2007.

[107] G. P. Zhang. Neural networks for classification: a survey. *IEEE Transactions on Systems, man and Cyberenetics, Part C*, 30(4):451–462, 2000.

[108] H. Foroughi, A. Naseri, A. Saberi, and H.S. Yazdi. An eigenspase based approach for human fall detection using itegrated time motion image and neural network. *IEEE 9th International Conference on Signal Processing*, pages 1499–1503, 2008.

[109] M.K. Fiaz and B. Ijaz. Vision based human activity traching using artifitial neural network. *IEEE International Conference on Intelligent and Advanced Systems*, pages 1–5, 2010.

[110] A. Iosifidis and A. Tefas. View-invariant action recognition based on artificial neural networks. *IEEE transactions ON Neural Networks Ans Learning Systems*, 23(3):412–424, 2012.

[111] R. E. Schapire. A brief introduction to boosting. *International Joint Conference on Artificial Intelligence*, 2:1401–1406, 1999.

[112] L. R. Rabiner. A tutorial on hidden markov models and selected application in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

[113] K. P. Murphy. *Dynamic Bayesian Networks: Representation, Iference and Learning*. PhD thesis, University of Clifornia, Berkeley, 2002.

[114] T. Starner and A. Pentland. Real time american sign language recognition from video using hidden markov models. *International Symposium on Computer Vision*, pages 109–116, 1995.

[115] H. L. U. Thuc, Shian-Re. Ke, P. V. Tuan, and T. N. Chan. Quasi-periodic action recognition from monocular videos via 3d human models and cyclic hmms. In *Proceedings of IEEE International Conference on Advanced Technologies for Communications*, pages 110–113,, January 2012.

[116] S. Park and J.K. Aggarwal. Ahierarchical bayesian network for event recognition of human actions and interactions. *Multimedia Systems*, 10(2):164–179, 2004.

[117] V. K. Singh and R. Nevatia. Human action recognition using a dynamic bayesian action network with 2d part models. *Indian Conference on Computer Vision, Graphics and Image Processing*, pages 17–24, 2010.

[118] N. Noceti and F. Odone. Human in groups: The importance of contextual information for understanding collective activities. *Pattern Recognition*, 47(11):3535–3551, 2014.

[119] M. Meng, H. Drira, M. Daoudi, and J. Booneart. Human object interaction recognition using rate-invariant shape analysis of inter joint distance trajectories. *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 999–1004, 2016.

[120] K.N.E. Slimani, Y. Benezeth, and F. Souami. Human interaction recognition based on the co-occurrence of visual words. *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 455–460, 2014.

[121] Y.S. Sofidgar, A. Vahdat, S. Se, and G. Mori. Discriminative key-component models for interaction detection and recognition. *computer Vision and Image Understanding*, 135:16–60, 2015.

[122] F. Sener and N. I. Cinbis. Two-person interaction recognition via spatial multiple instance embedding. *Journal of Visual Communication and Image Representation*, 32:63–73, 2015.

[123] A. Datta, M. Shah, and N. da Vitoria Lobo. Person-on-person violence detection in video data. *IEEE International Conference on Pattern Recognition*, pages 433–438, 2002.

[124] F. Yuan, Gui-Song. Xia, H. Sahbi, and V. Prinet. Mid-level features and spatiotemporal context for activity recognition. *Pattern Recognition*, 45(12):4182–4191, 2012.

[125] B. Zhang, P. Rota, N. Conci, and Francesco G.B De Natale. Human interaction recognition in the wild: Analyzing trajectory clustering from multiple-instance-learning perspective. *IEEE International Conference on Multimedia and Expo*, pages 1–6, 2015.

[126] S. Motiian, F. Siyahjani, R. Almohsen, and G. Doretto. Online human interaction detection and recognition with multiple cameras. *IEEE transactions on Circuits and Systems for Video Technology*, 27(3):649–663, 2017.

[127] S. Park and J.K . Aggarwal. Recognition of two-person interactions using a hierarchical

bayesian network. In *Proceedings of international workshop on Video surveillance*, pages 65–76, November 2003.

[128] Shi-Zhong Zhan and I-Cheng Chang. Pictorial structures model based human interaction recognition. In *Proceedings of International Conference on machine Learning and Cybernetics*, pages 862–866, November 2014.

[129] L. Meng, L. Qing, P. Yang, and J. Miao amd X. Chen. Activity recognition based on sementic spatial relation. *IEEE International Conference on Pattern Recognition*, pages 609–612, 2012.

[130] S. Jeba Berlin and M. John. Human interaction recognition through deep learning network. *IEEE International Carnahan Conference on Security technology*, pages 1–4, 2016.

[131] Y. Zhao, T. Sun, X. Jiang, and S. Wang. Long-term residual recurrent network for human interaction recognition in videos. *19th International congress on Image and signal processing, Biomedical Engineering and informatics*, pages 78–83, 2016.

[132] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[133] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[134] M. Bregonzio, S. Gong, and T. Xiang. Action recognition with cascade feature selection and classification. *Conference on: 3^{rd} International Crime Detection and Prevention* (ICDP 2009), pages 1–6, 2009.

[135] P. Moreno, P. Ribeiro, and J. Santos-Voctor. Feature selection for tracker-less human activity recognition. *International Conference on Image Analysis and Recognition*, pages 152–160, 2013.

[136] J. Zheng and Z. Jiang. Submodular attribute selection for action recognition in video. *Neural Information Processing Systems*, pages 1341–1349, 2014.

[137] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd Ediction, Springer, Stanford, California, 2009.

[138] I. Guyon annd S. Gunn, M. Nikravesh, and L. Zadeh. *Feature Extraction: Foundations and Applications*. Studies in Fuzziness and Soft Computing, Springer, New York, 2006.

[139] K. Hwang, K. Lee, C. Lee, and S. Park. Multi-class classification using a signomial function. *J. of the Operational Research Society*, 66(3):434–449, 2015.

[140] K. Lee, N. Kim, and M-K. Jeong. The sparse signomial classification and regression model. *Annals of Operations Research*, 216(1):257–286, 2014.

[141] J. Tang, S. Alelyani, and H. Liu. Feature selection for classification: A review. *In: Data Classification: Algorithms and Applications. CRC Press*, pages 38–64, 2014.

[142] T.N. Lal, O. Chapelle, J. Weston, and A. Elisseeff. Embedded methods. *In: I. Guyon and M. Nikravesh and S. Gunn and L.A. Zadeh (eds) Feature Extraction. Studies in Fuzziness and Soft Computing, Springer*, 207:137–165, 2006.

[143] L. Hermes and J.M. Buhmann. Feature selection for support vector machines. *IEEE International Conference on Pattern Recognition*, II:712–715, 2000.

[144] S. Perkins, K. Lacker, and J. Theiler. Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356, 2003.

[145] A. Rakotomamonjy. Variable selection using svm-based criteria. *Journal of Machine Learning Research*, 3:1357–1370, 2003.

[146] M.S. Allili and D. Ziou. Likelihood-based feature relevance for figure-ground segmentation in images and videos. *Neurocomputing*, 167:658–670, 2015.

[147] Y. Grandvalet and S. Canu. Adaptive scaling for feature selection using svms. *Neural Information Processing Systems*, pages 569–576, 2002.

[148] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, and T. Poggio. Feature selection for svms. *Neural Information Processing Systems*, pages 668–674, 2000.

[149] B. Krishnapuram, L. Carin, M.T. Figueiredo, and A.J. Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):957–968, 2005.

[150] M. Shuangge and H. Jian. Regularized roc method for disease classification and biomarker selection with microarray data. *Bioinformatics*, 21(24):4356–4362, 2005.

[151] J. Huang, M. Shuange, and C. H. Zhang. The iterated lasso for high dimensional logistic regression. Technical Report 392, Department of Statistics and Actuarial Science, University of Iowa, November 2008.

[152] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. A. Müller, E. Säckinger, P. Simard, and V. Vapnik. Comparison of classifier methods: a case study in handwriting digit recognition. *International Conference on Pattern Recognition*, pages 77–87, 1994.

[153] J. H. Friedman. Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University, 1996.

[154] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. *In Advances in Neural Information Processing Systems*, 12:547–553, 2000.

[155] J. Zhu and T. Hastie. Kernel logistic regression and import vector machine. *Journal of Computer and Graphical Statistics*, 14(1):185–205, 2005.

[156] P. Raghavan C.D. Manning and M. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, England, 2008.

[157] B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, July 1992.

[158] B. Scholkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* The MIT Press, Massachusetts, United States of America, 2001.

[159] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000.

[160] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, pages 265–292, 2001.

[161] Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines. *Computing Science and Statistics*, pages 1–15, 2001.

[162] Chih-Wei. Hsu and Chih-Jen. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):1–26, 2002.

[163] K.P. Murphy. *Machine Leaming: A Probabilsitic Perspective*. The MIT Press, Massachusetts Institute of Technology, United States of America, 2012.

[164] A.Y. Ng and M.I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in Neural Information Processing Systems*, pages 841–848, 2001.

[165] A. Genklin, D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

[166] D. W Hosmer and S. Lemeshow. *Applied Logistic regression.* Wiley Series in probability and statistics, United States of America, 2013.

[167] J. Klimaszewski. A comparison of regularization techniques in the classification of handwriting digits. *Journal of Theoretical and Appleid Computer Science*, 9(4):957–968, 2015.

[168] G. Kimeldorf and G. Wahba. Some results on tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82–95, 1971.

[169] J. Weston, A. Elisseff, and B. Schölkopf. Use of the zero-norm with linear models and kernel models. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

[170] P. Bradley and O. Mangasarian. Features selection via concave minimization and support vector machine. *International Conference on Machine Learning*, pages 82–90, 1998.

[171] O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. *Advanced Lectures on Machine Learning. Lecture Notes in Computer Science*, 3176:169–207, 2004.

[172] G. Sharma, A. Agarwala, and B. Bhattacharya. A fast parallel gauss-jordan algorithm for matrix inversion using cuda. *Computers & Structures*, 128:31–37, 2013.

[173] N. Rao, R. Nowak, C. Cox, and T. Rogers. Classification with the sparse group lasso. *IEEE Transactions on Signal Processing*, 64(2):448–463, 2016.

[174] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson, J.R. Marks, and J.R. Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. In *Proceedings of the National Academy of Sciences*, pages 11462–11467, 2001.

[175] M. Vincent and N-R. Hansen. Sparse group lasso and high dimensional multinomial classification. *Computational Statistics & Data Analysis*, 71:771–786, 2014.

[176] H. Wang and C. Leng. A note on adaptive group lasso. *Computational Statistics and Data Analysis*, 52(12):5277–5286, 2008.

[177] M.J. Bravo, , and H. Farid. Object recognition in dense clutter. *Perception and Psychophysics*, 68(6):911–918, 2006.

[178] A. Elgammal and C.-S. Lee. Tracking people on a torus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):520–538, 2010.

[179] L. Wang and D. Suter. Recognition human activities from silhouettes: Motion subspace and factorial discriminative graphical model. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[180] D. Wu and L. Shao. Silhouette analysis-based action recognition via exploiting human poses. *IEEE Transactions on Cyrcuits and Systems for Video Technology*, 23(2):236–243, 2013.

[181] J. Le and M. Park. An adaptive background subtraction method based on kernel density estimation. *Sensors*, 12(9):12279–12300, 2013.

[182] O. Ouyed and M.S. Allili. Feature relevance for kernel logistic regression and application to action classification. *IEEE International Conference on Pattern Recognition*, pages 1325–1329, 2014.

[183] R. Salakhutdinov and G.E. Hinton. Deep boltzmann machines. *International Conference on Artificial Intelligence and Statistics*, pages 448–455, 2009.

[184] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313:504–507, 2006.

[185] W. Yang, Y. Wang, and G. Mori. Human action recognition from a single clip per action. *IEEE 12th International Conference on Computer Vision, ICCV Workshops*, pages 482–489, 2009.

[186] Y. Yuan, X. Zheng, and X. Lu. A discriminative representation for human action recognition. *Pattern Recognition*, 59:88–97, 2016.

[187] Q. Li, H. Cheng, Y. Zhou, and G. Huo. Human action recognition using improved salient dense trajectories. *Computational Intelligence and neuroscience*, pages 1–8, 2016.

[188] Hong bin Tu, Li min Xia, and Zheng wu Wang. The complex action recognition via the correlated topic model. *The Scientific World Journal*, pages 1–11, 2014.

[189] Y.Y. Lin, J.H. Hua, N.C. Tang, M.H. Chen, and H.Y.M. Liao. Depth and skeleton associated action recognition without online accessibl rgb-d cameras. *IEEE Conference on Computer Vision and Pattern recognition*, pages 61–70, 2014.

[190] M.B. Holte, T.B. Moeslund, N. Nikolaidis, and I. Pitas. 3d human action recognition for multi-view camera systems. *IEEE Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 342–349, 2011.

[191] S. Spurlock, H. Wu, and R. Souvenir. Multi-view recognition using weighted view selections. *Asian Conf. on Computer Vision*, pages 538–552, 2014.

[192] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligences*, 32(9):1627–164, 2010.

[193] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *international Journal of Computer Vision*, 50(2):203–226, 2002.