SAPIENZA
UNIVERSITÀ DI ROMA

An Original Framework for Understanding Human Actions and Body Language by using Deep Neural Networks

PhD School of the Department of Computer Science at Sapienza University of Rome

Dottorato di Ricerca in Informatica – XXXII Ciclo

Candidate

Massaroni Cristiano
ID number 1246355

Thesis Advisor	Co-Advisor
Prof. Luigi Cinque	Prof. Danilo Avola

29 Settembre 2019

Thesis defended on 28/02/2020
in front of a Board of Examiners composed by:

Prof. Andrea Torsello (chairman)
Prof. Francesco Lopresti
Prof. Montella Raffaele

An Original Framework for Understanding Human Actions and Body Language by using Deep Neural Networks
Ph.D. thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Version: March 13, 2020

Author's email: massaroni@di.uniroma1.it

To live is to risk it all. Otherwise, you're just an inert chunk of randomly assembled molecules drifting wherever the universe blows you.
Rick Sanchez from "Rick and Morty" sitcom

Abstract

The evolution of both fields of Computer Vision (CV) and Artificial Neural Networks (ANNs) has allowed the development of efficient automatic systems for the analysis of people's behaviour. By studying hand movements it is possible to recognize gestures, often used by people to communicate information in a non-verbal way. These gestures can also be used to control or interact with devices without physically touching them. In particular, sign language and semaphoric hand gestures are the two foremost areas of interest due to their importance in Human-Human Communication (HHC) and Human-Computer Interaction (HCI), respectively. While the processing of body movements play a key role in the action recognition and affective computing fields. The former is essential to understand how people act in an environment, while the latter tries to interpret people's emotions based on their poses and movements; both are essential tasks in many computer vision applications, including event recognition, and video surveillance.

In this Ph.D. thesis, an original framework for understanding Actions and body language is presented. The framework is composed of three main modules: in the first one, a Long Short Term Memory Recurrent Neural Networks (LSTM-RNNs) based method for the Recognition of Sign Language and Semaphoric Hand Gestures is proposed; the second module presents a solution based on 2D skeleton and two-branch stacked LSTM-RNNs for action recognition in video sequences; finally, in the last module, a solution for basic non-acted emotion recognition by using 3D skeleton and Deep Neural Networks (DNNs) is provided. The performances of RNN-LSTMs are explored in depth, due to their ability to model the long term contextual information of temporal sequences, making them suitable for analysing body movements. All the modules were tested by using challenging datasets, well known in the state of the art, showing remarkable results compared to the current literature methods.

Acknowledgements

First of all, I would like to express my special appreciation and thanks to my advisor Professor Luigi Cinque. I would like to thank you for encouraging my research since the Master Degree thesis, and for allowing me to grow up as a research scientist.
I could not have imagined having a better advisor and mentor for my Ph.D study.

A special thanks to Professor Gian Luca Foresti. I would like to thank you for your insightful comments and encouragement, your guidance helped me in all the time to widen my research from various perspectives.

Another special thanks to Professor Danilo Avola. Through your support and teachings, you taught me what it means to be a researcher.

Thanks to my friends of the VisionLab, in these years you have shared with me victories and defeats, and above all a lot of laughs. During my Ph.D. you have been one of the most beautiful things that happened to me.

The biggest thanks to my parents, they are my guiding light. Behind all my goals achieved, there is always your love to encourage me. Inspired by you every day, I try to become a better person.

Finally, thanks to all my friends and my wonderful girlfriend Ilaria, you are my lifeline. However things may go wrong, I know you are close to me and I so everything will be fine.

Contents

1 Introduction 1

1.1 State of the art . . . 5

1.1.1 Hand gesture recognition . . . 5

1.1.2 Action recognition . . . 6

1.1.3 Affect recognition . . . 8

1.2 Contributions and outline . . . 10

2 Deep Neural Networks 13

2.1 Introduction . . . 13

2.2 Neural network basic concepts . . . 13

2.2.1 Learning Paradigm . . . 14

2.2.2 Perceptron . . . 14

2.2.3 Multi-layer perceptron . . . 16

2.3 Neural Network Architectures . . . 17

2.3.1 CNN: Convolutional Neural Network . . . 18

2.3.2 RNN: Recurrent Neural Network . . . 20

2.3.3 LSTM: Long-Short Term Memory Network . . . 20

2.4 Conclusions . . . 23

3 Proposed Framework 25

3.1 Overall architecture overview . . . 26

3.2 Hand gesture recognition module . . . 26

3.2.1 Architecture overview . . . 26

3.2.2 Feature Extraction . . . 26

3.2.3 Sampling Process . . . 28

3.2.4 N-Stacked LSTMs . . . 30

3.2.5 Network Training . . . 32

3.3 Action recognition module . . . 32

3.3.1 Architecture overview . . . 32

3.3.2 2D Skeleton and Feature Extraction . . . 33

3.3.3 Two-Branch Stacked LSTM-RNNs . . . 33

3.3.4 Partial Body Occlusion Management . . . 34

3.3.5 Multiple Subject Identification . . . 36

3.3.6 3D-DenseNet . . . 37

3.4 Non-acted body affect recognition module . . . 38

3.4.1 Architecture overview . . . 39

3.4.2 Feature description 39
3.4.3 Feature extraction 43
3.4.4 Proposed network architecture 44
3.5 Conclusions 45

4 Test and evaluation 47
4.1 Hand gesture recognition experiments 47
4.1.1 ASL Dataset 47
4.1.2 Selection of the Optimal Number of Stacked LSTMs 49
4.1.3 Effectiveness of the Selected Features 50
4.1.4 Hand Gesture Recognition on the ASL Dataset 50
4.1.5 Comparisons 51
4.2 Action recognition experiments 54
4.2.1 Datasets 54
4.2.2 Data Augmentation by C-RNN-GANs 56
4.2.3 LSTM Versus Phased LSTM 57
4.2.4 Partial Occlusion Tests 58
4.2.5 Feature Analysis 60
4.2.6 Quantitative Comparisons 61
4.2.7 Two-Branch Stacked LSTM versus 3D-DenseNet 64
4.3 Non-acted body affect recognition experiments 64
4.3.1 Dataset 64
4.3.2 Implementation details 65
4.3.3 Model and feature analysis 65
4.3.4 Data augmentation 68
4.3.5 Comparisons 70
4.4 Conclusions 71

5 Conclusions 73
5.1 Contribution 73
5.2 Future developments 74

Chapter 1

Introduction

You are walking around the city, wearing headphones and listening to your favourite music. Although you cannot hear people's speeches and street sounds, your eyes receive a lot of information that allow you to analyse what is happening in the surrounding environment. In particular, focusing only on the body and hands of people within your visual field, you could understand what they want to communicate, what they do, and, in same cases, what they feel. Through the use of hands, people can communicate with you through the non-verbal language of gestures (i.e., Hand Gesture Recognition), thus managing your inability to listen because of headphones. While, examining the whole body movements, you are able to understand the human action, understanding better the events that are taking place in the scene (i.e., Action Recognition). For example, we can detect which person has to take the bus that is leaving, recognizing who is running at breakneck speed, compared to others who walk slowly. Recognizing actions is an easy task for us, as they are explicitly codified in precise body movements, but often the latter can be implicitly related to affects (i.e., affective computing), much more difficult to classify. Snorting or tapping the foot or fingers can be considered actions, but at the same time they can indicate nervousness, if we consider, for example, a person who is waiting for a train with considerable delay at the station. So, thanks to visual information, we can also be able to perceive people's emotions, greatly increasing the degree of awareness of what is happening around us.
Based on these considerations, in this PhD thesis, a framework based on Computer Vision (CV) methodologies and Artificial Neural Networks (ANN) is presented to reproduce the human capacity to understand actions and body language by using RGB cameras and depth sensors. The framework is mainly composed of three modules: the first one is used to recognize **hand gestures**, the second one is involved in **action recognition** task and, finally, the last one is focused on **body affective computing**.

Hand gesture recognition is still a topic of great interest for the computer vision community, thanks to the wide range of information that can be expressed using the many gestures that the fingers can compose. Different categorizations of hand gestures can be defined depending on the type of information that the hands intend to transmit. Based on the researches of Kendon [75] and Quek *et al.* [137], a possible taxonomy of hand gesture categories can be proposed as follows:

- **Deictic** are the hand gestures that involve a pointing activity to establish the identity or spatial location of an object within the context of an application domain;
- **Manipulative** are usually performed by freehand movements to mimic manipulations of physical objects, such as in virtual or augmented reality interfaces;
- **Semaphoric** are specific hand gestures that define a set of commands and/or symbols to interact with machines. They are often used alternatively to the speech modality, when the latter is unusable or ineffective;
- **Gesticulation** is one of the most natural forms of gesturing. It is commonly used in combination with conversational speech interfaces. These hand gestures are often unpredictable and difficult to analyse;
- **Language** are the hand gestures used for sign language. They are performed by combining a set of gestures to form grammatical structures for conversational style interfaces. In case of finger spelling, these gestures can be considered like semaphoric ones.

Hand gesture recognition provides a means to decode the information expressed by the reported categories, which are always more used to interact with innovative applications, such as interactive games [92, 139], serious games [9, 130], sign language recognisers [6, 77, 101, 105, 161], emotional expression identifiers [12, 175], remote controllers in robotics [19, 48], advanced computer interfaces [123, 129, 140, 142], and others. In general, as shown in the taxonomy presented in Fig.1.1, the approaches used in hand gesture recognition can be divided into two main classes: 3D model-based [24] and appearance-based [97]. The former uses key elements of the body parts to acquire relevant 3D information, while the latter uses images or video sequences to acquire key features. In 3D model-based, the hand can be represented using mainly two ways: volumetric models [207] and skeletal models [96]. The first ones reproduce with high accuracy the shape of the hand as mesh of vertices or non-uniform rational B-spline, while the second ones, instead of using all the parameters of volumetric type, focus on the significant part of the hand, taking into account a set of equivalent joint angle parameters together with segment lengths. In the past, several RGB cameras were necessary to obtain a 3D model of the body parts, including hands. Recent works, supported by advanced devices, e.g., Microsoft Kinect [205] or LMC [55], as well as novel modelling algorithms based on depth map concept [158], have enabled the use of 3D models within everyday application domains. In the proposed hand gesture recognition module, a 3D skeletal model-based method to classify sign language and semaphoric hand gestures is implemented.

For what concerns **human action recognition**, in recent years, it is becoming increasingly important for the computer vision community [133, 33]. This is because it can be considered a main prerequisite for many smart and automatic applications, ranging from event interpretation to behaviour understanding. In [146], for example, a system for the automatic detection of crimes like chain and purse snatching is

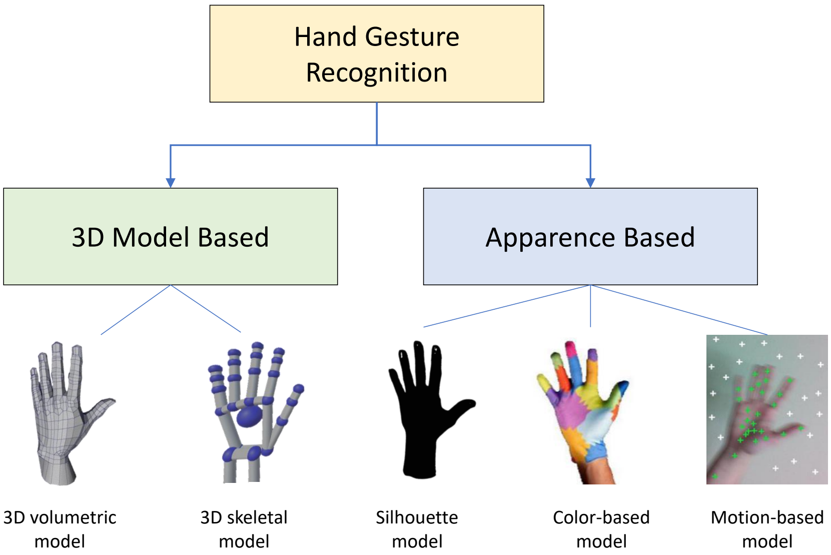


Figure 1.1. Taxonomy of gesture representation models.

reported. In [143], instead, an automated video surveillance system to detect anomalous human behaviours is presented. In general, action recognition approaches can be distinguished between not-skeleton-based and skeleton-based. The first category is composed of all those methods that analyse video streams acquired by RGB cameras. The works in this class, often supported by a preprocessing stage in which foreground subjects are detected [8, 5], use techniques based on frame-by-frame image analysis to extract key features (e.g., shapes, regions, keypoints) and compute spatio-temporal relationships among them [133, 124]. Differently, skeleton-based techniques analyse video streams acquired by RGB-D cameras and process features derived by skeletons [134]. The latter allow to calculate a wide range of dynamic attributes (e.g., leg acceleration, elbow rotation), which can be used to discriminate specific actions and enrich contextual information useful to classify them in complex scenarios. Nowadays, most systems that use skeletons are based on consumer RGB-D cameras. These devices can provide suitable 3D skeletons, but also present several drawbacks (e.g., low spatial resolutions, saturation problems, optical interferences) due to the inability of depth sensors (e.g., structured-light, time-of-flight, stereo-vision) to work properly in uncontrolled outdoor environments [69, 199]. Conversely, RGB cameras are now robust enough to be used in any kind of environment. Moreover, recent studies are showing how accurate and reliable 2D skeletons can be generated even by using a single RGB camera [21], thus overcoming many of the limitations previously reported. An additional crucial aspect of the action recognition regards the modelling of long dynamic activities, especially for the classification of complex activities that share sub-actions. Even if different solutions have been developed in the current literature [133, 33, 124], none of them can still be considered a reference point. Recently, Recurrent Neural Networks (RNNs) and, specifically, variants with Long Short-Term Memory (LSTM) cells [60], have obtained remarkable results in many applications that manage long sequential data [52, 136, 169, 51, 6, 68]. Despite this, their use in action recognition still requires to be further investigated. This

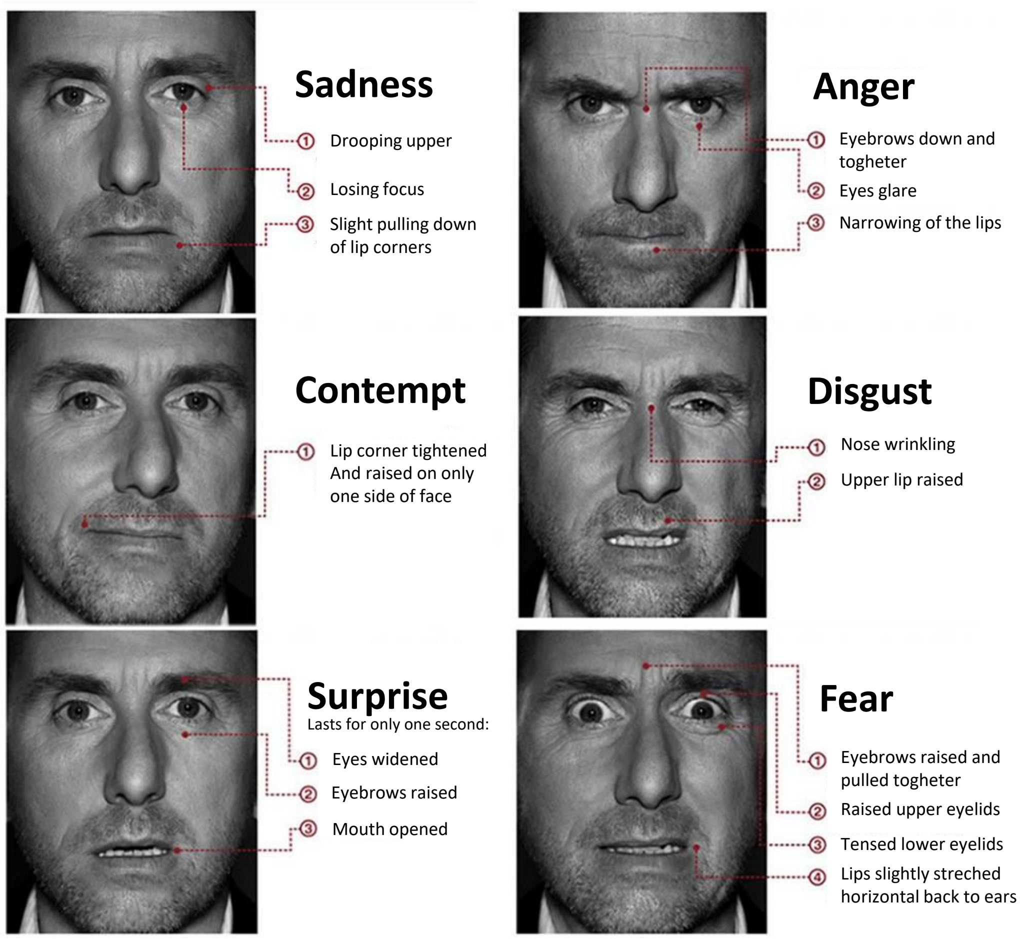

Sadness
① Drooping upper
② Losing focus
③ Slight pulling down of lip corners
Anger
① Eyebrows down and togheter
② Eyes glare
③ Narrowing of the lips
Contempt
① Lip corner tightened And raised on only one side of face
Disgust
① Nose wrinkling
② Upper lip raised
Surprise
Lasts for only one second:
① Eyes widened
② Eyebrows raised
③ Mouth opened
Fear
① Eyebrows raised and pulled togheter
② Raised upper eyelids
③ Tensed lower eyelids
④ Lips slightly streched horizontal back to ears

Figure 1.2. Example of methodical approach proposed by Facial Action Coding System.

thesis proposes an action recognition module based on 2D skeleton and two-branch stacked LSTM-RNNs.

Finally, the last topic covered by this thesis regards the **body affect recognition**. Emotions are an important aspect of our daily lives [32] since they affect our cognitive processes as well as how we respond to social interactions. The affective behaviours exhibited by a person allow to convey several levels of information to an interlocutor and can be expressed via various communication channels, including language, facial expressions, or body postures. Humans are in general apt to understand and interpret the information given by these behavioural cues which are, in fact, key in human-human interaction [31]. Affective computing [20] is the research field focused on the design of autonomous systems that try to reproduce this interpretative human ability. Over the years, this field has received great interest from the scientific community, resulting in relevant applications, such as security [53], marketing [10], or health-care [172]. Most of the available computer vision research on affective computing deals with the emotions perceived by facial expressions [201, 206, 29]. This is due to the wide availability of formal models, e.g., the Facial Action Coding System (FACS) [39], which provides a methodical approach to treat the affect recognition task, as shown in Fig.1.2. Contrarily, the lack of formal models has strongly limited the research on emotions related to postures and body movements. Moreover, it was customary to consider the body as a simple intensity indicator of the emotions previously detected by the face [110]. Thanks to later studies [108, 178], body analysis has been recently considered an important component in the affect recognition task and, encouraged by this new perspective, several works have started to analyse the body [45, 72]. Despite the good results shown by these solutions, their experiments were based on acted expressions performed by actors, which tried to reproduce as genuine as possible affects. As a matter of fact, a real challenging scenario for an automatic system is provided by non-acted natural expressions [81, 44], which are more complex and less separable compared to the acted ones. In the last module, a solution for basic non-acted emotion recognition based on 3D skeleton and Deep Neural Networks (DNNs) is provided.

1.1 State of the art

In this Section, the state-of-the-arts of the hand gesture recognition (Sec.1.1.1), action recognition (Sec.1.1.2) and affect recognition (Sec.1.1.3) fields are presented.

1.1.1 Hand gesture recognition

In the current literature of 3D model-based, hand and body gesture recognition follows a conventional scheme: the features are acquired from one or more sensors (such as, Kinect [203, 182, 42], LMC [181, 101]) and machine learning techniques (e.g., Support Vector Machines (SVMs) [122, 186], Hidden Markov Models (HMMs) [190, 96], Convolutional Neural Networks (CNNs) [118, 51]) are used to perform a classification phase. A reference work is reported in [122], where an SVM is used with Histogram of Oriented Gradients (HOGs) as feature vectors. Wang *et al.* [186] and Suryanarayan *et al.* [170] used an SVM with volumetric shape descriptors. Using the same classifier, Marin *et al.* [105] applied a combination of features extracted by Kinect and LMC sensors. Other interesting solutions are based on HMMs, such as that proposed in Zun *et al.* [208], where a robust hand tracking to recognize hand signed digit gestures is reported.

Different well-known techniques are extended and customized to reach increasingly better results. An example is shown in [77], where a semi-Markov conditional model to perform finger-spelling gesture recognition on video sequences is presented. The Hidden Conditional Random Field (HCRF) method, proposed in Wang *et al.* [190], is instead used to recognize different human gestures. Lu *et al.* [101] use an extension of the HCRF to recognize dynamic hand gestures driven by depth data. Regarding the hand pose estimation, the solution proposed in Li *et al.* [96] shows excellent results by applying a Randomized Decision Tree (RDT).

Another common solution is based on the use of Dynamic Time Warping (DTW). Although DTW does not belong to the class of machine learning techniques, it is often used in time series classification. In Vikram *et al.* [181], a DTW to support a handwriting recognition process based on the trajectory of the fingers extracted by a LMC is presented. In [152], the DTW with a novel error metric to match patterns, combined with a statistical classifier, is used to perform a tool to aid the study of basic music conducting gestures. In Sohn *et al.* [161], a pattern matching method by the combination of a DTW and a simple K-Nearest Neighbor (K-NN) classifier is used.

Recently, the great performance of the deep neural networks has motivated the use of the CNNs in different application domains, including the gesture recognition. For example, in [118], a multimodal convolutional neural network for classification of dynamic poses of varying duration is presented, taking in input the audio stream, both the video and depth stream of each hand, and the articulated pose information extracted from depth maps of the upper part of the body. In [195], a Hidden Markov Model (HMM) is proposed for simultaneous gesture segmentation and recognition using multimodal input, where a Gaussian-Bernouilli Deep Belief Network (DBN) is used to handle skeletal dynamics, and a 3D Convolutional Neural Network (3DCNN) is implemented to manage and fuse batches of depth and RGB images. While, in [34], a VGG11 network [171] is used to classify 3D dynamic hand gesture, proposing a new dataset, composed by sequences of hand skeletal data in addition to the depth, to train this network. Moreover, some recent works, such as [115], have shown how the LSTMs are potentially more effective than CNNs in recognizing gestures.

1.1.2 Action recognition

In the last decade, action recognition has been extensively studied [133, 33, 124]. These studies have led to the development of a large variety of different approaches, which can be commonly classified as *descriptive*, *syntactic*, or *statistical*.

The first class includes those approaches that represent actions by composing logical, spatial, or temporal relationships. A key work is reported in [17], where the authors describe videos by using spatio-temporal graphs in which nodes correspond to multi-scale video segments, while arcs define their hierarchical, spatial, and temporal connections. In a more recent work [95], the authors define occurring relationships among acts composing an activity by extracting a sequential scheme. On the latter, a set of concurrent contextual actions and information is coded by a sequence of symbols. Approaches of the second class define a set of rules and predicates for each application domain. Actions are represented by means of entities, which are described by types, attributes, functions, and relationships. A typical example is reported in [35], where a set of rules is defined to estimate several complex actions. In particular, the authors initially use a modelling approach to provide a preliminary classification of the observed action. Afterwards, a common-sense reasoning algorithm is adopted to analyse and correct the initial estimation. In [174], the authors use a common-sense domain knowledge algorithm based on first-order logic production rules. These are used in combination with a relaxed deduction algorithm to construct a Markov Logic Network (MLN), which is used to perform probabilistic inference for input queries on events of interest. In [127], to deduce the objectives of a set of subjects in a video, plausible actions are predicted by an algorithm based on a Stochastic Context-Sensitive Grammar (SCSG). By the grammar, hierarchical compositions of events and temporal relations of sub-events are defined. The alphabets of the SCSG describe actions which are defined by the poses of subjects and their interactions with objects in the scene.

The approaches considered above, although effective, can have different drawbacks

[133, 124, 132, 112]. First, the distinction of fine actions cannot be easily handled. Second, the modelling of actions, especially by graphs and grammars, can be quite inconvenient. Finally, in the classification of very long actions various interpretation errors can occur. To overcome the just reported limitations, and thanks to recent advances in computational capability, last few years have seen an increasing diffusion of methods based on the third class. The latter enable the definition of actions by a set of states and support probabilistic models to easily describe the key dependences. Relevant works, in this class, are based on Dynamic Bayesian Networks (DBNs). In [89], the authors use a DBN, an object-oriented action hierarchy, and an approximate Viterbi-like algorithm, to recognize complex long-term activities. In [191], instead, the authors describe a Probabilistic Graphical Model (PGM) to incorporate a scene, an event object interaction, and an event temporal context into DBNs for the recognition of different events, including actions. Recently, other kinds of solutions are increasingly proposed. In [135], for example, a method to learn 4-D spatio-temporal features from body joint coordinate data by using $L0$ norm constrained dictionary learning and sparse coding is proposed. In [176], the authors propose a Multi-label Hierarchical Dirichlet Process (ML-HDP) to recognize human actions modelling complex activities and motion units, simultaneously. The hierarchical Bayesian non-parametric model, combining the Improved Dense Trajectories (iDT) with Motion Boundary Histogram (MBH) descriptor, formulates the co-occurrence relationship of actions and motion units, therefore highly accurate recognition can be achieved. To achieve great performance for 3D human action recognition, in [64], an approach based on manifold analysis and deep network structure with rotation mapping layers is proposed. As an alternative, methods based on Probabilistic Petri Nets (PPNs) and Convolutional Neural Networks (CNNs) are also used. In [88], a Particle Filter Petri Net (PFPN) is used to combine uncertain event observations, thus determining the likelihood that a particular activity can occur in a video. In [98], the authors propose an original video representation, named VLAD for Deep Dynamics (VLAD^3), based on deep CNN features, for recognizing complex activities that share sub-actions. Relying on the idea that human poses are helpful for human action recognition, in [103], the authors introduce a multitask CNN to estimate human poses and predict human actions, simultaneously. In [183], to address action recognition in multi-view videos, a dividing and aggregating multi-branch CNN is proposed. The lower CNN layers, shared by all views, are used to learn view-independent features, whereas one CNN per view is used to learn view-specific features. A message passing mechanism, based on Conditional Random Fields (CRFs), is used to integrate the different view-specific representations in order to generate more refined features. Finally, a fusion module combines the prediction results from all the view-specific classifiers. In [93], instead, to overcome the view dependence, distances between pair-wise skeleton joints in both the three orthogonal 2D planes and the 3D space are calculated. The authors encode those pair-wise distances into color texture images, referred as Joint Distance Maps (JDM), then they use CNNs to learn the discriminative features for human action and interaction recognition. In [179], contrary to other methods that leverage from a few video frames, a long-term 3D convolution framework is proposed. The authors introduce long-term temporal convolutions (LTC) 3D-CNNs, integrating longer 3D max pooling, to model a large number of frames, thus capturing the action representation at its full temporal scale. They demonstrate an advantage over the use of 16 video frames. However, in [188], to handle long action videos (i.e., long-range temporal information) Temporal Segment Networks (TSN) have been introduced. The action video sequence is divided into segments from each of which snippets are randomly sampled using a sparse temporal sampling strategy. A consensus of CNNs on different snippets is used to combine the preliminary prediction of each snippet. In [200], instead, the authors introduce an Attention-based Temporal Weighted CNN (ATW) to select best clips of a video for action recognition. Differently, in [128], the authors encode 3D skeletal data into a sequence of RGB images. Then, the obtained motion images are given as input to ResNets to recognize the human actions. With the aim to avoid the time-consuming optical flows calculation and speed up the human action recognition for real-time applications, in [202], a two-stream CNN based on motion vectors (MVs) is proposed. However, directly training CNN from MVs results in degrading performance due to noise and lack of fine details in MVs. To solve the problem, based on the intuition that optical flows and motion vectors inherit similar structures, the authors transferred knowledge from the optical flows domain to the motion vectors one, obtaining a very fast approach which meet the real-time requirement. Among statistical approaches, those based on RNNs and, especially, on LSTM models are recently gaining great attention [65, 99, 94, 7]. The reason is that RNNs are suitable for the management of long video sequences. Moreover, they provide a high level of abstraction, thus allowing a detailed description of events, including actions. A pioneering work, in this direction, is reported in [65], where the authors propose a two-stage deep temporal model for group activity recognition based on two LSTM models. The first designed to represent action dynamics of individual people, the second designed to aggregate person-level information for whole activity understanding. In [94], the authors propose a method capable of locating the attended spatial areas and selective video segments for action recognition. They used CNNs to model spatial information and LSTM networks to identify the most temporally indicative video segments. Recently, several solutions have proposed to manage human skeleton as if they are graphs [197, 157]; obtaining excellent results in different action recognition challenges based on 3D skeletal data. For example, in [157], the skeleton data is considered ad a directed acyclic graph (DAG), based on the kinematic dependency between the joints and bones in the natural human body.

1.1.3 Affect recognition

The most accurate affect recognition systems are based on electroencephalography [163, 150]. Although these systems achieve excellent results, they are limited by the use of dedicated sensors that require a controlled environment. The computer vision applications, on the other hand, are more suitable to work in real and uncontrolled scenarios, thanks to the use of more versatile sensors, such as RGB, RGB-D, or thermal cameras. Most of the vision based methods involve emotion recognition by the analysis of facial expressions [29, 66, 106]. This is due to the presence of a great deal of labelled data in the state-of-the-art, organized in datasets, such as in [114]. Although the face is one of the most discriminative ways to identify people's emotions, it is not always possible to capture facial expressions in large and crowded environments. This aspect motivated researchers to try other solutions, including body analysis. Thanks to several studies [108, 178, 87], data concerning the body, correctly labelled with emotions, are beginning to appear. This is also a consequence of the evolution of sensors and feature extraction techniques, which allow to obtain increasingly higher performance in the estimation of body data. Thus, in the last years, different methods for body affect recognition [80, 119, 73] were developed. In the current literature, approaches are typically divided into two main categories: body posture-based methods (i.e., focusing on key poses), and body movement-based methods (i.e., exploiting the analysis of movements).

Body posture-based methods

This type of approach, given a sequence of frames, analyses a key body pose contained in a single frame. A valid example is reported in [30], where the correlation between six emotions and body postures is analysed by using anatomical features extracted from computer-generated mannequins. In [72], several basic machine learning techniques, such as logistic regression, naive Bayes, and decision tree classifier, were used to analyse the data acquired by the VICON system. A key work was proposed in [82], where a Mixture Discriminant Analysis (MDA) and an unsupervised Expectation Maximization (EM) model were used to build separate cultural models for affective posture recognition. Postures can communicate both discrete emotion categories and affective dimensions, as shown in [79], by using body models derived by analysing several postural configuration features. In [71], a method based on Gaussian process classification and Bayesian inference, to detect the level of frustration of students in an Intelligent Tutoring System, is proposed. This solution uses facial and postural features, where the former is obtained by RGB cameras, while the latter is derived from a pressure-sensitive chair. The method proposed in [49], extracts body poses from a depth video corpus of computer-mediated human-human tutoring. The authors discovered that the identified postural patterns are associated with frustration, focused attention, decreased involvement, and disengagement affective states. Finally, one of the main works that focuses on the recognition of non-acted body emotions is presented in [81], where a benchmark dataset, concerning non-acted affects extrapolated from people playing with the Nintendo Wii console, is presented. Using key body postures and an MLP network, the authors manage to get results which are close to the base rate obtained by human observers.

Body movement-based methods

This type of approach processes human body movements by analysing all the frames of a sequence. In [131], the authors use a linear model based on cues combination to examine facial expressions and body movements, so that an affect impression can be produced. In [45], a small set of visual features extracted from two video cameras, and an unsupervised learning method based on clustering techniques, are used as a framework to analyse affective behaviour. While these presented works show good results on acted data, they still do not address the more difficult non-acted scenario. The authors of [44], using the dataset proposed in [81], present an interesting collection of meta-features, based on body movements, to train a Support Vector Machine (SVM). Focusing on the field of Human–computer interaction (HCI), in [149], a measure of the aesthetic experience of people playing Nintendo sport videogames, is proposed. The authors perform this task using a method to automatically recognize emotional expressions conveyed by body movements. Finally, in [54], the proposed method focuses on recognizing natural laughters from fake ones. Achieved by analysing natural laughter categorisations patterns, animated on a minimal avatar generated from both natural and acted motion-capture data.

1.2 Contributions and outline

This section reports in brief the contribution of this thesis work with respect to the state-of-the-art. Concerning the hand gesture recognition, our framework presents the following contributions:

- the selection of a simple set of features, based on the joint angles, that are highly discriminative for the recognition of any type of hand gesture, especially for sign language and semaphoric hand gestures;
- the creation, by the LMC, of a large dataset to support the comparison of sign language recognizers based on the hand skeleton model. Notice that, the LMC guarantees a high precision in the estimation of the joint positions [192];
- the capability of analysing and recognizing a large number of hand gestures in two main areas of interest like sign language and semaphoric hand gestures. Notice that, the study of static and dynamic hand gestures of the ASL provides a prerequisite for achieving wider recognition systems for the sign language;
- for the first time in hand gesture recognition field, the use of stacked LSTMs is explored in depth. Furthermore, even more important, an accuracy of over 96% on the created sign language based dataset and a comparison on the SHREC dataset that outperforms in accuracy competing approaches of the current literature, are reported.

While, The key contributions of the proposed action recognition module can be summarized as follows:

- An alternative approach based on 2D skeletons extracted from RGB videos to detect and track multiple people performing long dynamic actions even in uncontrolled and crowded outdoor environments;
- An original pipeline, based on two-branch stacked LSTM-RNNs and lightweight features derived by 2D skeleton joints, to parallelize the framework performance and manage partial body occlusions;
- Management of the lack of sufficient data (on three datasets), in training phase, by the creation of real-looking video sequences generated by means of a C-RNN-GAN;
- Comparative experiments on two benchmarks showing that the proposed method outperforms the current state-of-the-art. Other tests, on a total of eight datasets, showing its robustness in managing incorrect 2D skeleton

extraction, perspective changes, partial body occlusions, and comparability with 3D skeleton based approaches.

To conclude, the main contributions of the last module, in relation to the current literature for natural non-acted affect recognition, can be summarized in three key points:

- the proposal of an original combination of local and global temporal features to describe body movements both at a low level, examining a given time instant (local) as well as at a high level, considering the whole analysed time window (global);
- the analysis of body movements in relation to time as a consequence of temporal local features, therefore allowing a dynamic posture examination instead of a static one;
- the introduction of deep learning, to handle body affect recognition, by leveraging a custom architecture based on merging LSTM and MLP networks to correctly manage the proposed features.

It can be noted that all the modules don not care about the device used to acquire the input, but they only offer processing methods. The rest of this thesis is structured as follows. In the Chapter 2, basic concepts and the most common algorithms used in Deep Learning are presented, giving particular notice to advanced architectures such as the RNN. The Chapter 3 presents the proposed framework, describing in detail the pipeline of each module. In Chapter 4, test and evaluation, performed on all the framework modules, are discussed. Finally, the Chapter 5 concludes this thesis, exposing what are the thoughts about this work and how the system can be improved in the future.

Chapter 2

Deep Neural Networks

This chapter is structured as follow. In Section 2.1, the main definition and the aim of machine learning and ANNs are introduced. In Section 2.2, base concepts underlying ANNs are presented. In Section 2.3, the most famous ANN architectures, typically used in deep learning applications, are described in detail. In Section 2.4, the different type of features analysed in the proposed framework are explained. Finally, in Section 2.4, a summary of the treated topics is reported.

2.1 Introduction

Machine Learning (ML) [13] is the scientific branch focused on the study of algorithms and statistical models that computer systems use to learn and act like humans do. Their learning improves over time in autonomous way, by feeding them data and information derived from observations and real-world interactions. Over the years, ML has been employed in numerous application domains, such as: spam filters [117], search engines [107], text categorization [154], and artificial vision [145, 15].

A remarkable class of ML methods is the family of ANNs, strongly inspired by a simplification of the biological neural networks that constitute animal brains. An ANN is based on a collection of connected units, also known as artificial neurons, where each connection, like the synapses in a biological brain, transmit a signal to other neurons. The artificial neuron process the input signal and then propagates the output to other neurons connected to it.

In this PhD thesis, the methodologies presented belong to the branch of DNNs [46], a typology of ANN algorithms that uses multiple layers of neurons to progressively compute higher level features from the input data. This branch is often also referred to as Deep Learning (DL). In the last years, DNN models have improved the state-of-the-art results of many domains, ranging from speech recognition [59] to object detection [141].

2.2 Neural network basic concepts

In this section, the common learning paradigms used in ML and basic concepts of the ANNs are presented.

2.2.1 Learning Paradigm

The neural networks require a training phase base on sample data, in order to update their weight and derive the correct predictions for each input. Three main learning paradigms are used to describe the different typology of training that a network (or any ML method) can employ, as reported to follow:

- **Supervised Learning**: this learning paradigm is usually used when a large and well defined dataset is available for a given task. The algorithms based on supervise learning take as input an input-label pairs for each data samples, where the label describes the correct expected output. In this way, the network tries to create a good world model representation relative to the given task. Typically the field experts generate a dataset to be used at training time. To follow, a validation step is used to determine the model abstraction ability. In case the results are not satisfactory, the network architecture parameters (e.g., the learning rate) can be modified in order to improve the performances.

- **Unsupervised Learning**: in this learning paradigm the input is neither classified nor labelled. Consequently, the algorithms have to manage the data without guidance. Objective for this learning paradigm, is to find and understand common characteristics among the provided samples in order to aggregate them in the right way and provide the correct output. The required steps for this paradigm are typically composed of: a dataset generation phase used to generate the sample data, a model inference via information clustering of the sample data, and, finally, a validation phase to check the quality of the selected clustering. Once the training phase is completed, the network can be used to classify new unseen data. This paradigm is a good choice when there is no way to manually label a meaningful quantity of samples to apply the Supervised Learning paradigm.

- **Semi-Supervised Learning**: this last learning paradigm try to merge same concept of both Supervised and Unsupervised learning paradigms. The Semi-Supervised Learning paradigm performs the training using typically a small amount of labelled data and a large amount of unlabelled data, the latter used as additional support information. This paradigm is useful when there are not enough labelled data to produce an accurate model and the resources are insufficient to get more, so the semi-supervised techniques can be applied to increase the size of your training data.

2.2.2 Perceptron

The perceptron model is a simple architecture, based on human brain neurons [144], and it can be used to recognize and classify patterns. A real neuron, as shown in Fig.2.1, is mainly composed by four main components, namely the dendrites, soma, axon, and synapses, which allow to receive, elaborate and transmit, electrical signals. The latter are received via the dendrites and soma, and, then, they are sent out down the axon. The synapse is the structure that permits to connect an axon and other neuron dendrites. Neurons are electrically excitable and, if there is a large

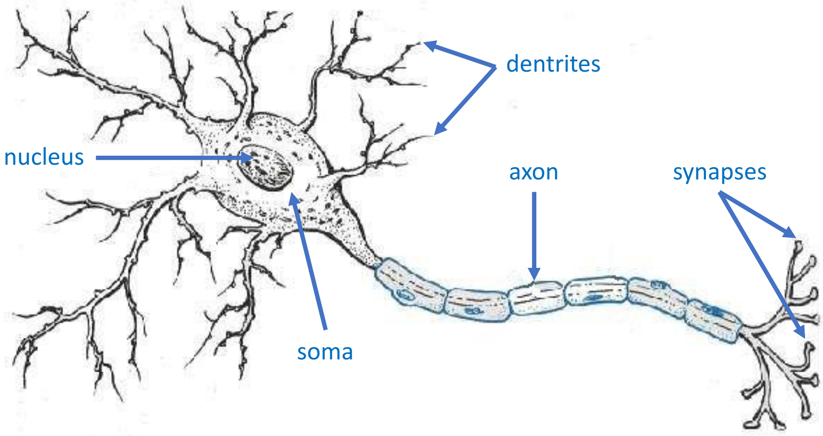


Figure 2.1. Human neuron diagram.

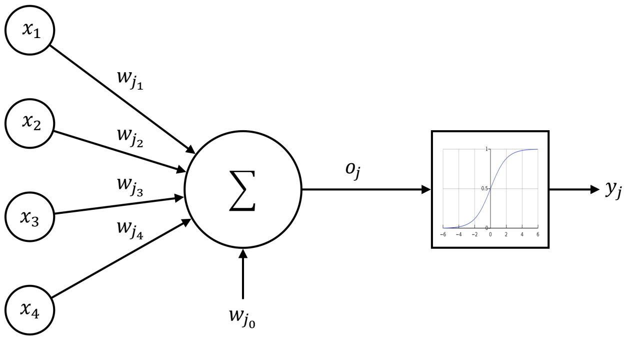


Figure 2.2. Perceptron architecture.

enough amount of voltage changes over a short time interval, it can generates an all-or-nothing electrochemical pulse. This potential travels rapidly along the axon, and activates synaptic connections as it reaches them. The voltage that reaches the soma can be reduced or increased, based on the type of synaptic signal, which can be, respectively, inhibitory or excitatory.

The perceptron model is a feed-forward neuron, which means that the data flow is unidirectional from input to output. A generic j-th perceptron j can receive multiple inputs which are multiplied by weights and then summed together (i.e., weighted sum), in accordance with the following equation:

$$o_j = \omega_{j_0} + \sum_{i=1}^{n} \omega_{j_i} x_i \tag{2.1}$$

where, x_i represents the i-th input, w_{j_i} is the weight associated to the i-th input and w_{j_0} is the bias for this j-th unit. The bias as the shifting parameter that makes it possible to move or “translate” the activation function left or right on the graph. As shown in Fig.2.2, the perceptron output o_j is then used in conjunction with an activation function, generating the final output y_j. Sigmoid, Tanh or binary functions are some of the most common activation functions used by the perceptron.

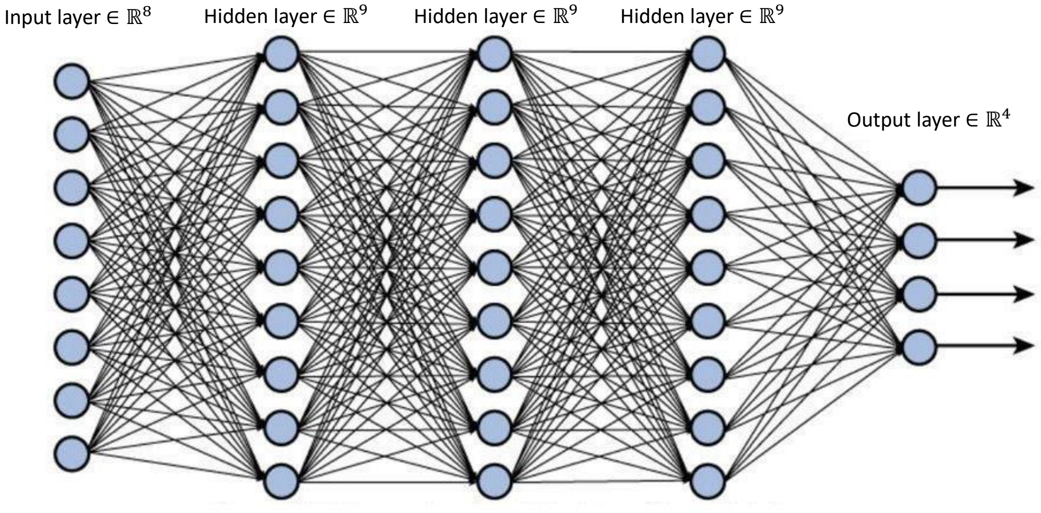

Input layer $\in \mathbb{R}^8$
Hidden layer $\in \mathbb{R}^9$
Hidden layer $\in \mathbb{R}^9$
Hidden layer $\in \mathbb{R}^9$
Output layer $\in \mathbb{R}^4$

Figure 2.3. Example of Multi-layer perceptron composed by 8 input nodes, 9 nodes in both hidden layers, and 4 nodes in the output layer.

2.2.3 Multi-layer perceptron

A perceptron, although it replies well the behaviour of a neuron, cannot handle non linearly separable inputs. The Multi-Layer Perceptron (MLP) [13] is a network of perceptrons, called nodes, arranged on multiple layers, designed to handle this issue. As shown in Fig.2.3, based on their position inside the network, the layers are divided into three typologies, called input layer, hidden layer and output layer, respectively. Like the perceptron, the MLP is a “feed-forward” network, where all units of a layer are connected with those in the next layer via weights, moving the informations in a unidirectional path from input layer to output. The MLP weights, usually, are updated during the training phase (i.e., following the supervised learning paradigm) by using the Back-Propagation (BP) [57] algorithm. The BP uses an optimization function to chose the best weights for each layer nodes, trying to reduce the loss, which is a number indicating how the network output is different compared to the correct result. The loss function can be divided into the following groups:

- The **Regression loss functions** are used in regression predictive modelling problem, which involves predicting a real-valued quantity.
- The **Binary Classification loss functions** are used in predictive modelling problems, where samples are assigned one of two labels (i.e., binary classification).
- The **Multi-Class Classification loss functions** are used in predictive modelling problems, where samples are assigned one of more than two classes (i.e., multi-class classification).

In this PHD thesis, having to deal mainly with multi-class classification problems, we are focused on Multi-Class Classification Loss Functions. Specifically, we use the Multi-Class Cross-Entropy Loss, defined by the following equation:

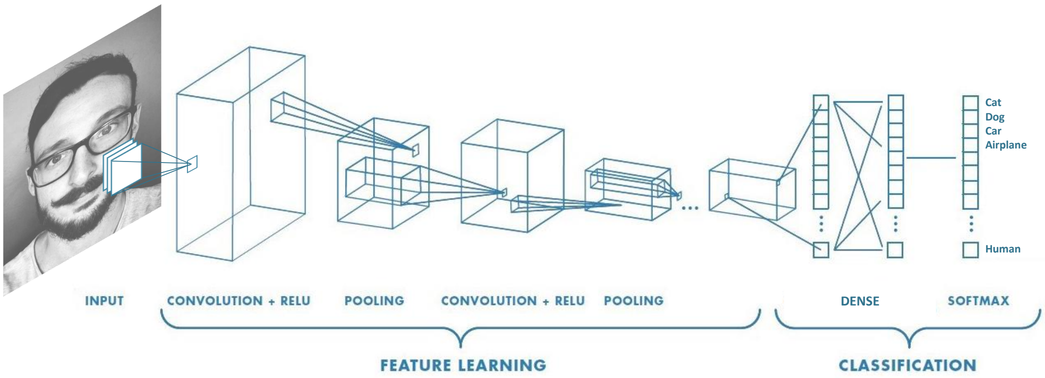


Figure 2.4. Example of CNN architecture.

$$H_{\hat{y}}(y) = -\sum_{i=1}^{N} \hat{y}_i \log(y_i) \tag{2.2}$$

where N indicates the number of classes, y_i is the predicted probability value for class i and $\hat{y}_i$ is the true probability for that class. It should be noted that, in the case of classification tasks, each node of the output layer is associated with a specific i-th class. The weight updating of a j-th node in the MLP network can be described by a general rule, called *delta rule*. The first step is to compute the weight variation between k and j, where k represents each node connected in input to j, using the following equation:

$$\Delta w_{jk} = -\eta \delta_j x_{jk} \tag{2.3}$$

where η indicates the learning rate; x_{jk} represents the k-th input of j, where $x_{jk} \equiv y_K$ (i.e., the output of the k-th neuron); and δ_j is the error. The value δ_j depends on the layer to which j belongs, and it is defined as follow:

$$\delta_j = \begin{cases} y_j(1-y_j)(\hat{y}_j - y_j), & \text{if j is an outout node} \\ y_j(1-y_j)\sum_k \delta_k w_{kj} y_i, & \text{if j is a hidden node.} \end{cases} \tag{2.4}$$

where $\sum_k \delta_k w_{kj}$ is the weighted sum of errors obtained by all units connected to the j-th neuron, and $\hat{y}_j$ is the real value of the output node j. Finally, the BT algorithm updates weights using the following equation:

$$w_{kj} = w_{kj} + \Delta w_{jk} \tag{2.5}$$

2.3 Neural Network Architectures

In this section, some of the most famous DL architectures are presented and explained in detail.

2.3.1 CNN: Convolutional Neural Network

The Convolutional Neural Network (CNN) [91] is a variation of the MLP, inspired by the animal visual system neurons, which react to specific tiny areas of the visual field, called receptive fields. Analysing portions of the given sample image, replicating the animal neuron behaviour, the CNN is able to find discriminative patterns inside the provided data. In this way, it can perform classification tasks without requiring human support to extrapolate the features. As in the MLP architecture, the CNN can be composed of hundreds or even thousands of hidden layers, where each layer learns different features from the image. Starting from the input, several filters are applied, obtaining a convoluted output used as input of the next layer. At the beginning, filters extract basic features (e.g., luminance, edges, etc.), but later, by varying the training iterations, they start to become more complex, defining each considered class in a unique way. The main layers, used for learning these complex features, are mainly three, namely the convolution, ReLU (Rectified Linear Unit), and pooling layers, respectively. Finally, a classification phase uses the feature learning output to make a predictions, by using a dense layer in conjunction with a softmax layer. This architecture can handle different types of inputs, ranging from texts [78] to sounds [83, 76], not focusing solely on image analysis. An overview of a CNN architecture is shown in Fig.2.4, while the various mentioned layers are briefly explained to follow:

- **Convolution:** this layer applies to the input image a series of convolutional filters, in order to learn specific features. Each filter (or kernel) slides across the image and it is used to perform a convolution operation, generating a new filtered image called feature map. The convolution operation is an element wise matrix multiplication, where the first term is the image, and the other one is the kernel. The general expression of a convolution operation is the follow:

$$g(x, y) = \omega * f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s, t) f(x - s, y - t), \tag{2.6}$$

  where $g(x, y)$ represents the filtered image, $f(x, y)$ is the input image and ω is the kernel, where $-a \leq s \leq a$ and $-b \leq t \leq b$ represent the local neighbours of the pixel. The number of pixels by which the kernel slides across the image is known by the term stride. Based on the filters used in the convolutions, different operations can be performed on the image, such as edge detection, sharpening, and blurring. The output of this layer is represented by all feature maps obtained by convolutional filters. An example of convolution operation is shown in Fig.2.5

- **ReLU:** this layer computes the activation values from each feature map entries, and it enables a faster and improved learning phase. In fact, with respect to the more common non-linear activation functions, like sigmoid functions (or logistic) and hyperbolic tangent, the ReLu function is able to accelerate the convergence of stochastic gradient descent [84]. Moreover, ReLU allows a network to easily obtain sparse representations.

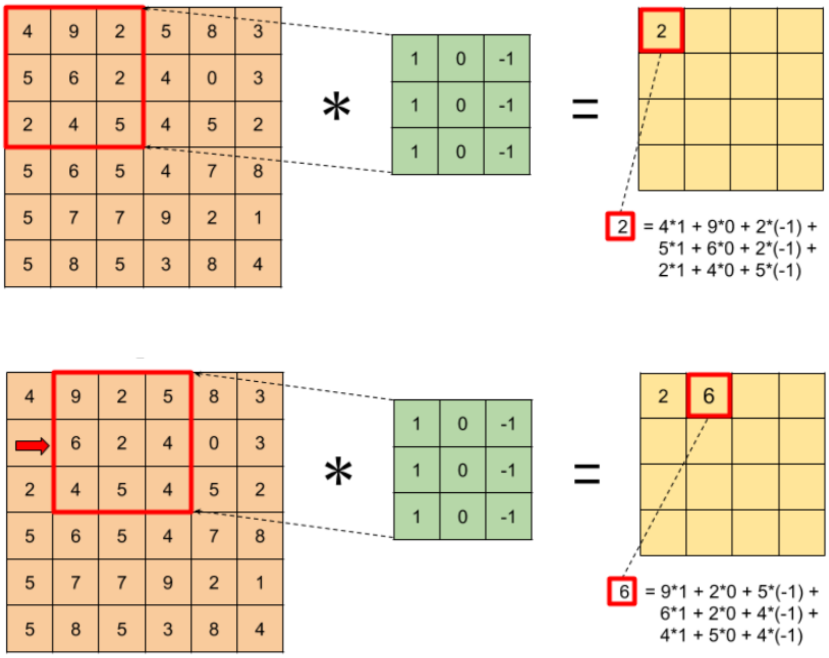


Figure 2.5. Example of convolution operation with stride=1, where the orange matrix represents the input image, the green one is the kernel and the yellow matrix is the output feature map.

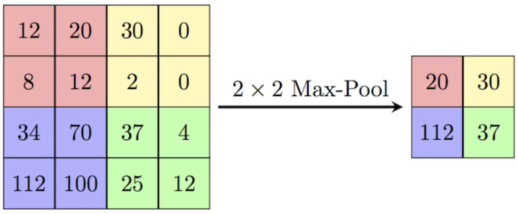


Figure 2.6. Example of 2 × 2 max pooling downsampling on a 4 × 4 matrix.

- **Pooling:** this layer is used to reduce the number of network parameters, by applying a non-linear downsampling on the feature maps. An example of pooling filter is given by the max-pooling, where the maximum value is computed for each patch of the feature map (Fig. 2.6).

- **Dense:** it is a fully connected layer, where each neuron receives input from all the neurons in the previous layer (i.e. densely connected). This layer gives as output a vector with dimension K, where K is the number of considered classes, which contains the analysis of high-level features generated by the previous layers.

- **Softmax:** this layer gives the final classification output. The softmax function [46] takes as input a vector of dimension K and normalizes it into a probability distribution. The class with the highest probability represents the prediction for the input image.

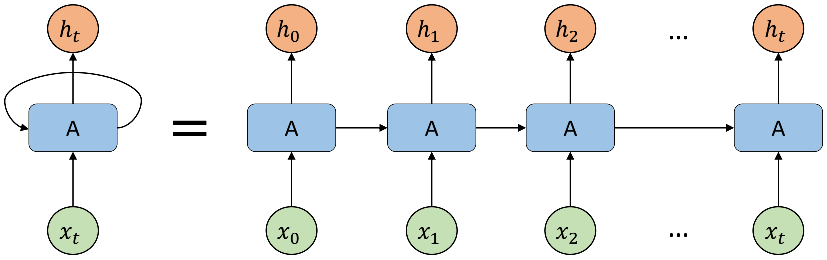


Figure 2.7. Example of folded (on th left) and unfolded (on the right) RNN unit diagrams.

2.3.2 RNN: Recurrent Neural Network

The most common neural network architectures can not correlate information through time. For example, in the classification of events in various points of a film, it is unclear how a MLP could use its reasoning about previous events in the video sequence to inform later ones. The Recurrent Neural Network (RNN) [51] was proposed to resolve this issue, using an internal loops that allow information to persist as a sort of memory.

The RNN architecture presents a network unit A that analyses, at a generic time step t, its x_t input, and generates the h_t output. In addition, the unit A passes information from one time-step to the next one, generating a loop, as shown in Fig.2.7. As can be seen further in Fig.2.7, the RNN can be represented like a chain of units, where each unit conveys information to its successor.

Formally, taken $x_1, \ldots, x_t \in R^n$ as inputs, the network computes $h_1, \ldots, h_t \in R^m$ outputs, according to the following equation:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{2.7}$$

where h_t represents the state at time t for an hidden neuron; σ indicates the sigmoid activation function (e.g. tanh function); W_{xh} is a weight matrix used in the connection between the input and hidden layers; W_{hh} is a weight matrix among recursive connections in a layer; W_{hy} denotes a weight matrix among the hidden and output layers, while the b_h vector is the bias.

2.3.3 LSTM: Long-Short Term Memory Network

Several factors, such as the error blowing up problem [151] and the vanishing gradient [50], do not allow the use of common activation functions (e.g., tanh or sigmoid) to suitably train a network composed by RNNs using long data sequences. The Long-Short Term Memory (LSTM) architecture, unlike RNNs, was designed to leverage long-term dependencies and is able to retain information (called cell state) over long periods of time, by using dedicated gates inside the single units. The difference between the LSTM and RNN architectures can be seen in Fig.2.8. Then, the major innovation of LSTM consists in conveying the cell state directly to the next unit via linear operations, as shown in Fig.2.9a, choosing which information to keep or remove through the dedicated gates.

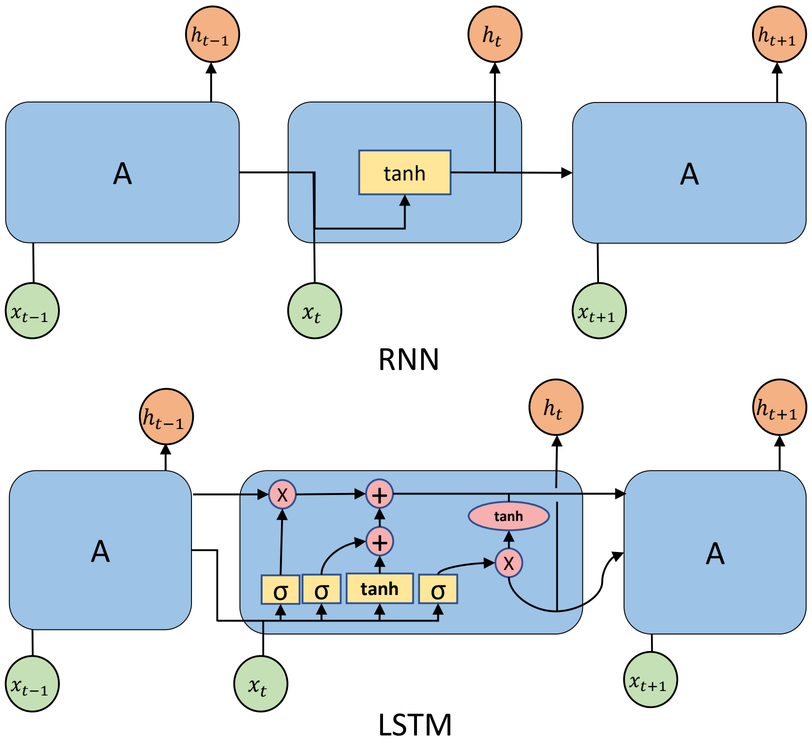


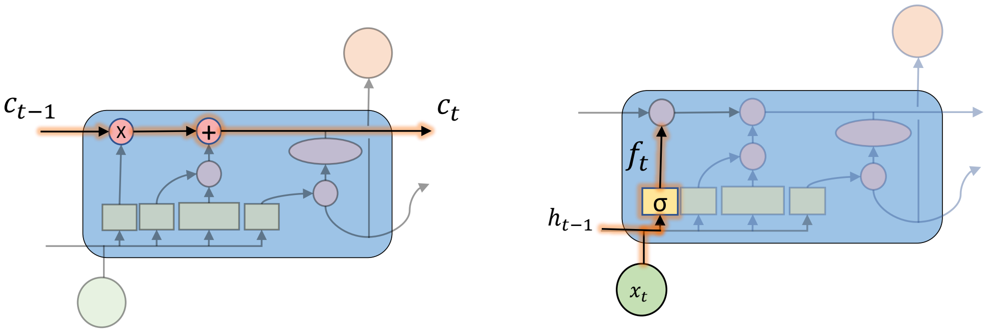
Figure 2.8. Comparison between RNN and LSTM units.

The *Forget Gate* decides whether information, from the input x_t and the previous hidden state h_{t-1}, has to be kept in the cell state. A sigmoid function σ is applied to x_t and h_t, providing either 0 or 1 as an output. Whenever a 1 is returned, the information is kept in the state cell, otherwise it is forgotten. An overview of the LSTM forget gate elements is shown in Fig.2.9b. Formally, activation vector equation of the forget gate can be defined as follows:

$$f_t = \sigma(W_{xf}x_t + W_{h_f}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2.8}$$

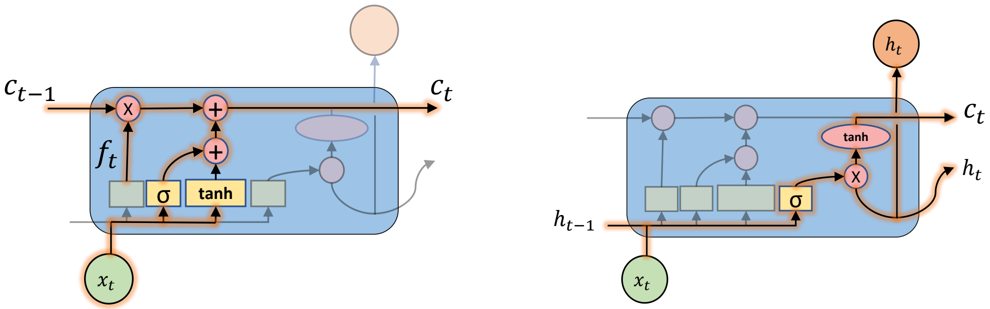
where h_f and W_{h_f} are the forget gate hidden state vector and its linked weight matrix, respectively. The term W_{xf} encodes the weights of forget gate, the matrix W_{cf} indicates the diagonal weights for peep-hole connections between the cell state vector c_{t-1} and the forget gate, while b_f denotes the forget bias vector.

The *Input Gate* decides what information can be stored inside the cell state. Two different path are implemented for this purpose, where the first one applies a sigmoid function to decide what has to be updated, while the second one uses a *tanh* function to create a vector containing the new candidate values for the new state. The outputs of these two paths are then multiplied and combined with the previous state c_{t-1} and the vector f_t obtained by the forget gate. An overview of the input gate paths is shown in 2.10a. Formally, activation vector equations of the input gate and the cell can be defined as follows:



(a) (b)

Figure 2.9. The LSTM cell-state transferral pathway (a), and the forget gate (b).



(a) (b)

Figure 2.10. The LSTM input (a) and ouput (b) gates.

$$i_t = \sigma(W_{xi}x_t + W_{h_i}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{2.9}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{h_c}h_{t-1} + b_c \tag{2.10}$$

where i, and c denote input gate and cell activation vectors. The terms h_ω and W_{h_ω} (with $\omega \in \{i, c\}$) indicate the hidden state vector and their linked weight matrix, respectively. The terms W_{xi} and W_{xc} encode the weights of input gate and cell. The matrix W_{ci} indicates the diagonal weights for peep-hole connection between the cell state vector c_{t-1} and the input gate, while b_i and b_c denotes the input gate and cell bias vectors, respectively. The $\odot$ operation denotes the element-wise product.

The *Output Gate:* this gate is used to decide the unit output, and is based on a filtered version of the updated cell state. A sigmoid function is used to decide what components of the cell state have to be output, while a *tanh* function is applied to the updated cell state to normalize the values between $[-1, 1]$. The two outputs are then multiplied and only the chosen parts will result as an output for a given unit. The gate pathway is shown in Fig. 2.10b. Formally, activation vector equation of the output gate can be defined as follows:

$$o_t = \sigma(W_{xo}x_t + W_{h_o}h_{t-1} + W_{co}c_{t-1} + b_o) \tag{2.11}$$

where h_o and W_{h_o} are the output gate hidden state vector and its linked weight matrix, respectively. The term W_{xo} encodes the weights of output gate, the matrix W_{co} indicates the diagonal weights for peep-hole connection between the cell state vector c_t and the output gate, while b_o denotes the output bias vector.

Finally, the LSTM unit output at time t, represented by the hidden state h_t, can be expressed as follows:

$$h_t = o_t \odot \tanh(c_t) \tag{2.12}$$

2.4 Conclusions

In this chapter, several ANN concepts were presented, starting from a brief introduction of ML and DL, up to the most common DNN architectures. Since the data processed in this thesis are organized as sequences, the LSTM-based architectures are the most suitable to be used in the implementation of the proposed modules. Despite this aspect, the MLPs and CNNs can be used to manage other supplementary data, calculated such as the information acquired from the surrounding environment, or global features computed over the entire sequence.

Chapter 3

Proposed Framework

This chapter is structured as follows. In Section 3.1, the architecture of the proposed framework and its modules at a high level are introduced. In Section 3.2, the features and the proposed method used for the recognition of hand gestures are described in detail. In Section 3.3, the architecture implemented for human action analysis is explained. In Section 3.4, the solution adopted to classify non-acted emotions by using body movements is presented. Finally, in Section 3.5, considerations on the framework modules are summarized.

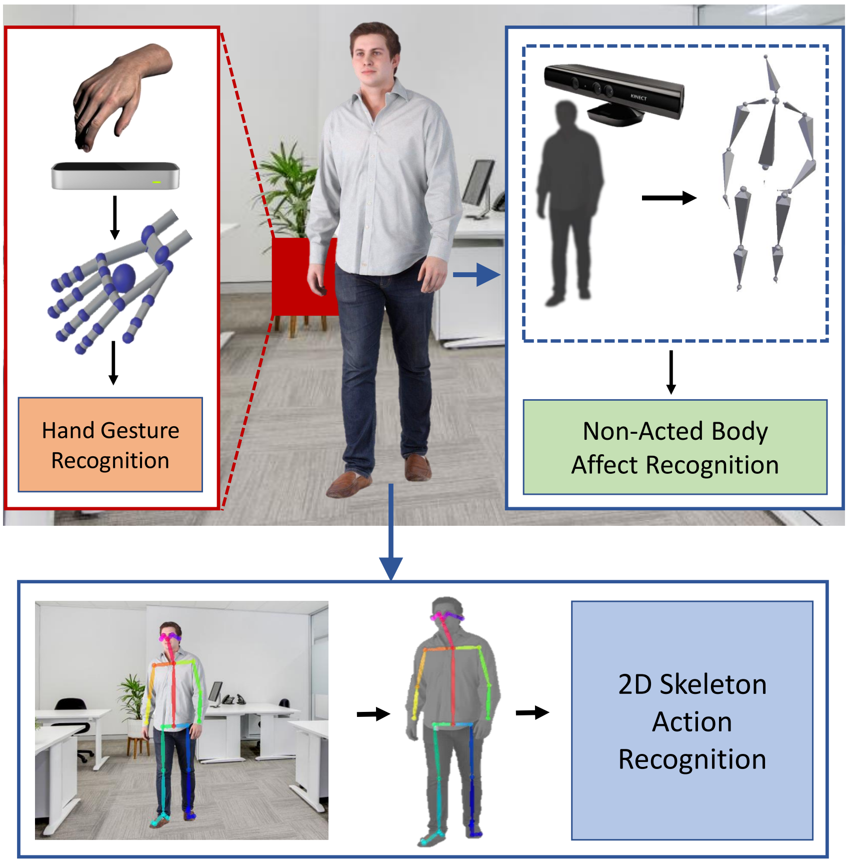

Hand Gesture Recognition
Non-Acted Body Affect Recognition
2D Skeleton Action Recognition

Figure 3.1. The overall architecture of the proposed framework.

3.1 Overall architecture overview

As shown in Fig.3.1, the proposed framework is divided mainly into three modules: hand gesture recognition, 2D Skeleton Action Recognition and Non-Acted Body Affect Recognition; the first two are regards the processing of body information expressed voluntarily, while the latter is used for the analysis of involuntary body information. The hand gesture and the non-acted body recognition modules work on 3D skeleton joints obtained by depth sensors, instead, the 2D Skeleton Action Recognition module is the only one that works on RGB video sequences, from which 2D skeletal data are extrapolated. In this framework, all the modules are focused solely on input data processing, abstracting from the acquisition sensors and methods.

3.2 Hand gesture recognition module

In this Section, the proposed hand gesture recognition method is described. The designed pipeline (Section 3.2.1) is composed of two main stages. In the first one, the features are extracted from a 3D hand skeletal data sequence (Section 3.2.2), and then sampled (Section 3.2.3). Finally, in the second one, the hand gesture features are classified by means of a trained network model, based on stacked LSTM units (Sections 3.2.4 and 3.2.5).

3.2.1 Architecture overview

In Fig. 3.2, the whole logical architecture of the proposed method is shown. Let us consider, each hand gesture acquired by a user is represented by a set $\mathbf{X} = \{\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_{T-1}\}$ of feature vectors, where T indicates the maximum number of time instants, inside a time interval Θ, in which the features are extracted by a LMC. Notice that, a LMC is chosen as reference device for the acquisitions because it is optimized for the hands and the obtained skeleton model provides very accurate dynamic information about finger bones [192]. The N-stacked LSTMs are applied to model these sequences of data, where a time series of feature vectors (one vector for each time instant) is converted into a series of hidden vectors $\mathbf{H} = \{\mathbf{h}_{N-1,0}, \mathbf{h}_{N-1,1}, ..., \mathbf{h}_{N-1,T-1}\}$. To follow, the last of these hidden vectors $\mathbf{h}_{T-1}$ is mapped by a dense layer to an output vector $\mathbf{y}$, which indicates the class probability of the gesture carried out at time t, with $0 \leq t \leq T-1$. Finally, the classification of the gestures is performed by a softmax layer [18] using $K = |C|$ classes, where C is the set of the considered gesture classes.

3.2.2 Feature Extraction

Each gesture can be considered as the composition of different poses, where each pose is characterized by particular angles. Such a concept has already been applied in several works, using the angles formed by the body joints to recognize human actions [120, 121, 23]. So, each feature vector $x_t \in X$, with $0 \leq t \leq T-1$, is mainly composed by (Fig. 3.3):

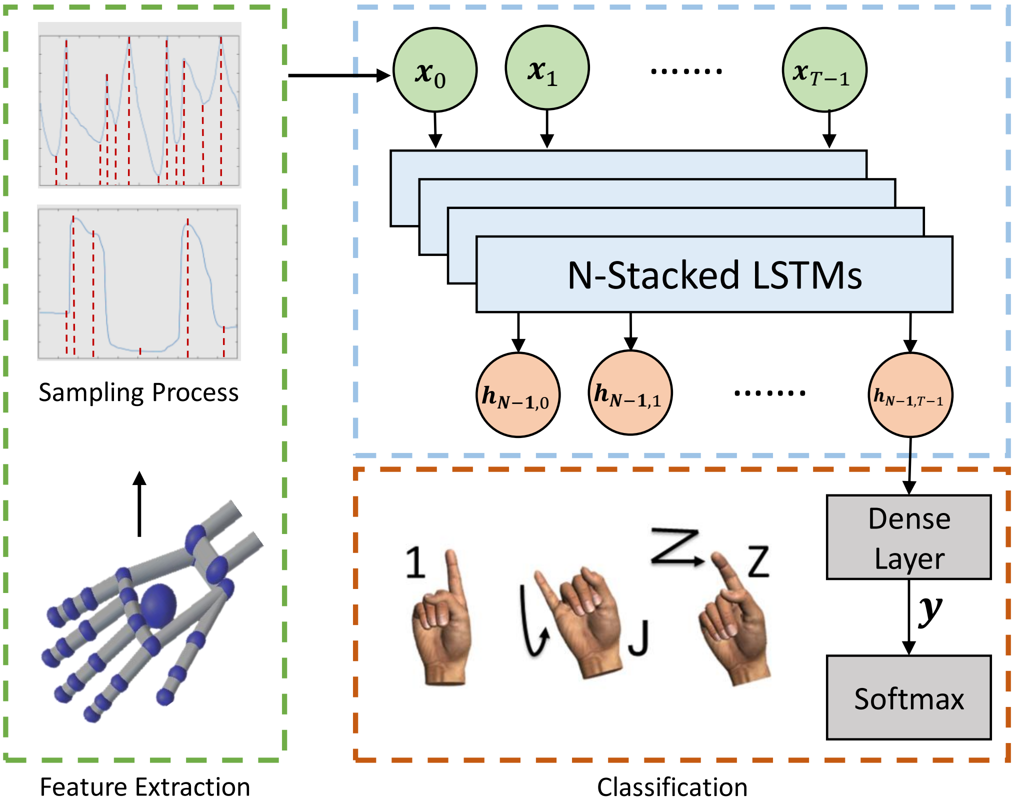


Figure 3.2. Logical architecture of the proposed method. The training phase is performed by N-stacked LSTMs. Given a sequence of input vectors, the N-stacked LSTMs return an output vector for each time instant t, with $0 \leq t \leq T-1$, that contains the probabilities for each class. K and T are the different classes of the hand gestures and the maximum number of time instants in which a gesture is acquired, respectively.

- the internal angles ω_1, ω_2, ω_3, and ω_4 of the joints between distal phalanges and intermediate phalanges. The internal angle ω_0, considered for the thumb, is computed between distal phalanx and proximal phalanx;

- the internal angles β_1, β_2, β_3, and β_4 of the joints between intermediate phalanges and proximal phalanges. The internal angle β_0, considered for the thumb, is computed between proximal phalanx and metacarpal.

Each finger can be seen as a set of segments, where $\overline{CD}$ is the distal phalanx, $\overline{BC}$ is the intermediate phalanx (with the exception of the thumb, where $\overline{BC}$ is the proximal phalanx), and $\overline{AB}$ is the proximal phalanx (with the exception of the thumb, where $\overline{AB}$ is the metacarpal). The angles are calculated as follows:

$$\omega_j = \arccos\left(\frac{\overline{BC} \cdot \overline{CD}}{|\overline{BC}||\overline{CD}|}\right) \tag{3.1}$$

$$\beta_j = \arccos\left(\frac{\overline{AB} \cdot \overline{BC}}{|\overline{AB}||\overline{BC}|}\right) \tag{3.2}$$

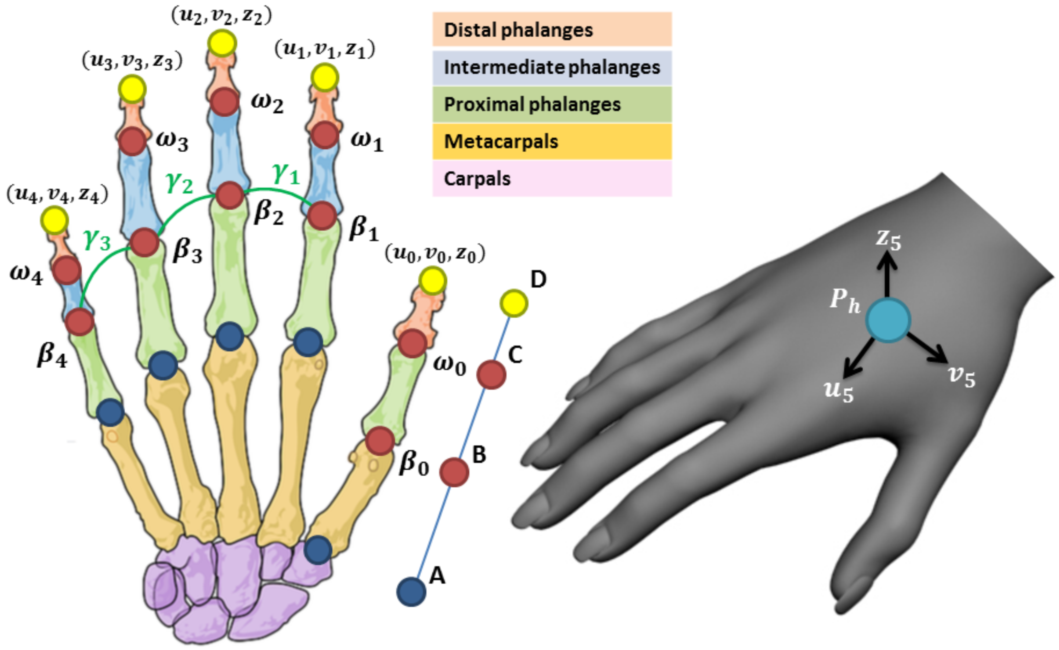


Figure 3.3. The features extracted from the hand: joint angles and fingertip positions. The yellow points indicate the fingertip positions on which the 3D displacements are computed. The red points indicate the joints on which the angles are computed.

where, $j = 0, .., 4$. Since the information provided by the angles is not sufficient to manage all types of existing hand gestures, especially dynamic gestures that perform movements in 3D space, additional information is used by considering the following features:

- 3D displacements u_5, v_5, z_5 of the position of the central point P_h of the palm of the hand. These features are considered to manage hand translation on the 3D space;
- 3D displacements u_l, v_l, z_l of the fingertip positions, with $l = 0, .., 4$. These features are considered to manage hand rotation in 3D space;
- the intra-finger angles γ_1, γ_2, and γ_3, i.e., the angles between two consecutive fingers, where the fingers considered are: the pointer finger, the middle finger, the ring finger, and the pink finger. These features are used to handle special cases of static gestures that differ from each other only in intra-finger angles, as shown in Fig. 3.4.

All the reported features are independent by the reference. Thus, the input vector assigned to the first layer of the N-Stacked LSTMs, at time t, is:

$$\mathbf{x}_t = \{\omega_0, ..., \omega_4, \beta_0, ..., \beta_4, u_0, v_0, z_0, ..., u_5, v_5, z_5, \gamma_1, \gamma_2, \gamma_3\} \tag{3.3}$$

3.2.3 Sampling Process

Since each person can perform the same gesture with different speeds, and since the proposed method requires that all the videos that must be analysed are composed

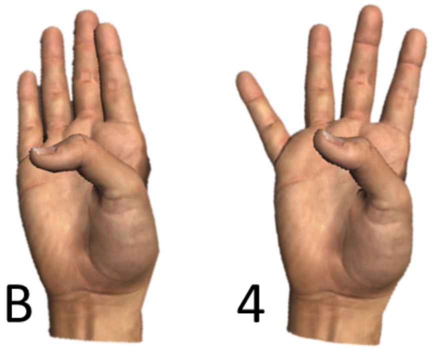


Figure 3.4. Example of gestures differentiated by the intra-finger angles γ_1, γ_2, and γ_3.

by the same number T of samples, a sampling process to select the most significant feature values within the entire time interval Θ of the hand gesture sequences was implemented. This means that data are acquired only in the most significant T time instants, where a time instant $t \in \Theta$ is defined as significant when the joint angles and the hand central point position P_h vary substantially between t and $t+1$ (as explained below).

Let $f_{\omega_i}(t), f_{\beta_i}(t)$, and $f_{\gamma_j}(t)$, with $0 \leq i \leq 4$ and $1 \leq j \leq 3$, be the functions that represent the value of ω_i, β_i, and γ_j angles at time t. In addition, let $f_{\phi_{(t)}}$ be the function that represents the value of ϕ (i.e., the displacement of the centre of the hand P_h with respect to the previous position at time $t-1$) at time t. Then, for each function $f_g(t)$, with $g \in G$ and $G = \{\omega_i, \beta_i, \gamma_j, \phi\}$, the Savitzky-Golay filter [148] is applied. The Savitzky-Golay filter is a digital filter able to smooth a set of digital data in order to increase the signal-to-noise ratio without greatly distorting the signal. Now, the significant variations on the considered features are identified through the relative maximum and minimum of each $f_g(t)$. All the time instants t, associated with at least one relative minimum or relative maximum of a feature g, are used to create a new set Θ^*, which represents a set of possible important time instants to be sampled. In Fig. 3.5, an example of this sampling phase is shown, where the behaviour of the function $f_{\omega_1}(t)$ (i.e., the angle of the distal phalanx of the index finger) for an instance of the gesture "milk" is considered. By applying the Savitzky-Golay filter, the signal shown in Fig. 3.5, that is affected by a certain amount of noise due to the acquisition device or tremors of the hand, can be suitably cleaned. Then, the maximum and minimum relative points are identified and sampled. In the example, only the procedure for the feature ω_1 is shown, but this step is performed for each feature $g \in G$. Now, depending on the cardinality of the set of the sampled time instants, the following cases must be considered:

- if $|\Theta^*| < T$, then the remaining $(|\Theta^*| - T)$ time instants to be sampled are randomly selected in Θ;
- if $|\Theta^*| > T$, then, only some significant time instants are sampled for each g feature. Let Θ_g be the set of the samples in Θ^* obtained from the relative maximum and minimum of the feature g ($\Theta_g \subseteq \Theta^*$), we need to know the number of time instants T_g that can be sampled for each g such that $\sum_{g \in G} T_g =$

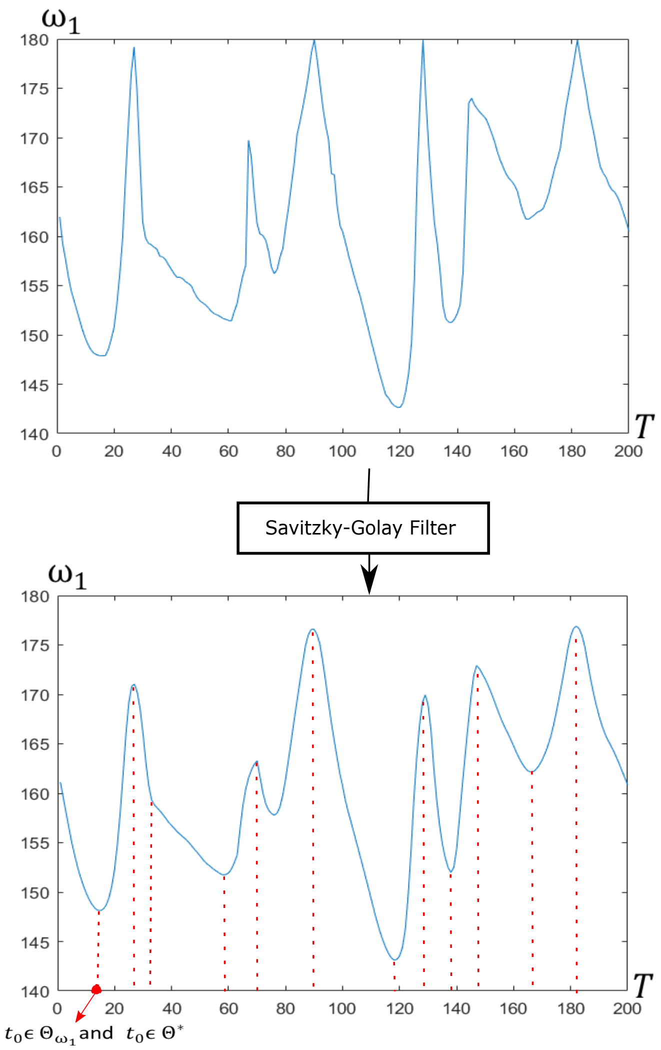


Figure 3.5. Sampling example for the feature ω_1 on the "milk" gesture.

T. Each T_g is obtained thought the following proportion $|\Theta_g| : |\Theta^*| = T_g : T$. Then, from each Θ_g set, we randomly take T_g samples.

After the sampling step, each acquisition instance is composed by a sequence $\{x_0, .., x_{T-1}\}$ of feature vectors. The proposed sampling procedure is dynamically based on the value of the features.

3.2.4 N-Stacked LSTMs

Although an LSTM allows to manage the problem of the vanishing gradient, the input time series often have a temporal hierarchy, with information that is spread out over multiple time scales which can not be adequately recognized by single LSTM layer. For this reason, stacked LSTMs were adapted in this method. In fact, by constructing recurring networks formed over multiple layers, a higher abstraction on the input data is reached [58]. Increased input abstraction does not always bring benefits, because the effectiveness of these networks depends on both task and analysed input.

In several works, such as [51, 147, 41], it was observed empirically that stacked LSTMs work better than shallower ones on speech recognition. The audio signals, analysed for example in speech-to-text task, can be elaborated on more abstractions ranging from the entire pronounced phrase to the syllables of each word. Moreover, each abstraction can be captured in different time scales within the considered period. Like in the case of audio sequences analysed in the speech recognition problem, hand gestures can be examined over multiple time scales. In fact, each gesture can be considered as composed by many small movements and sub-gestures of the hand and, as observed, this type of data processing is particularly suitable for this kind of network.

Based on these considerations, the LSTM stack-based solution was experimented and then compared to the performance of a single-level network. For each time instant $t = 0, \dots, T-1$, the activation functions for an LSTM unit at the l-th stack layer are the following:

$$\mathbf{i}_{l,t} = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{l,t-1} + \mathbf{W}_{ci}\mathbf{c}_{l,t-1} + \mathbf{b}_i) \tag{3.4}$$

$$\mathbf{f}_{l,t} = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{l,t-1} + \mathbf{W}_{cf}\mathbf{c}_{l,t-1} + \mathbf{b}_f) \tag{3.5}$$

$$\mathbf{c}_{l,t} = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_{l,t} \odot \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{l,t-1} + \mathbf{b}_c) \tag{3.6}$$

$$\mathbf{o}_{l,t} = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{l,t-1} + \mathbf{W}_{co}\mathbf{c}_{l,t-1} + \mathbf{b}_o) \tag{3.7}$$

$$\mathbf{h}_{l,t} = \mathbf{o}_{l,t} \odot \tanh(\mathbf{c}_{l,t}) \tag{3.8}$$

where $\mathbf{i}, \mathbf{f}, \mathbf{c}$, and $\mathbf{o}$ denote input gate, forget gate, output gate, and cell activation vectors, respectively. Moreover, $\mathbf{h}_\omega$ vectors model the hidden states (with $\omega \in \{\mathbf{i}, \mathbf{f}, \mathbf{o}, \mathbf{c}\}$) and $\mathbf{W}_{\mathbf{h}_\omega}$ terms indicate their linked weight matrices. Matrices $\mathbf{W}_{xi}$, $\mathbf{W}_{xf}$, $\mathbf{W}_{xo}$, and $\mathbf{W}_{xc}$ encode the weights of input gate, forget gate, output gate, and cell; while $\mathbf{W}_{ci}$, $\mathbf{W}_{cf}$, and $\mathbf{W}_{co}$ are the diagonal weights for peep-hole connections. Finally, $\mathbf{b}_i$, $\mathbf{b}_f$, $\mathbf{b}_c$, and $\mathbf{b}_o$ denote input, forget, cell, and output bias vectors, respectively. Function σ is the logistic sigmoid and $\odot$ denotes the element-wise product. The output of whole stacked LSTMs, at time t, is represented by the $\mathbf{h}_{N-1,T-1}$ vector.

Afterwards, a dense layer using a ReLU activation function is applied, thus connecting each entry value in $\mathbf{h}_{N-1,T-1}$ to an entry of the output vector $\mathbf{y}$ via a weight. This layer is used to map the vector $\mathbf{h}_{N-1,T-1}$ to a number of output nodes equal to the size of the set of hand gestures to be recognized.

The output $\mathbf{y}$ defines a probability distribution over the K possible gesture classes, where $\mathbf{y}^k$ (i.e., the k^{th} element of $\mathbf{y}$) is the estimated probability of a specific class C_k at time t for the acquired gesture $\mathbf{X}$. Finally, all results $\mathbf{y}$ are collected and normalized into the softmax layer, through the following equations:

$$\hat{\mathbf{y}} = \sum_{t=0}^{T-1} \mathbf{y} \tag{3.9}$$

$$\tilde{\mathbf{y}}^k = p(C_k|X) = \frac{e^{\hat{\mathbf{y}}^k}}{\sum_{q=0}^{K-1} e^{\hat{\mathbf{y}}^q}} \tag{3.10}$$

for each k, with $1 \leq k \leq K$. The classification of the gesture $\mathbf{X}$ will be given by the highest probability contained in $\tilde{\mathbf{y}}$.

3.2.5 Network Training

Given a dataset D composed of M train gesture sequences, the goal is to minimize the following maximun-likelihood loss function:

$$\mathcal{L}(D) = -\sum_{m=0}^{M-1} \ln \sum_{k=0}^{K-1} \delta(k, \tau) p(C_k | D_m) \tag{3.11}$$

where, D_m, $0 \leq m \leq M$, is an input sequence of the training dataset D, τ is the ground-truth label of D_m, and $\delta(\bullet, \bullet)$ is the Kronecker delta or delta function. This formulation is referred to the cross-entropy error proposed in [13]. The Back-Propagation Through Time (BPTT) algorithm [50] is used to obtain the objective function derived with respect to all the weights and to compute the minimization based on the stochastic gradient descent.

3.3 Action recognition module

In this section, the proposed human action recognition method based on 2D skeletons is described. Initially, in Section 3.3.2, the whole architecture is summarized. In Section 3.3.2, the preliminary extraction method of 2D skeletons from RGB video sequences is described. To follow, Section 3.3.3, in the hierarchical network model based upon stacked LSTM units is presented. Next, the solution proposed for the partial body occlusion and multiple subject tracking issues are defined defined in Section 3.3.4 and Section 3.3.5, respectively. Finally, in Section 3.3.6, the support network based on 3D-CNNs, used to manage missing skeletal data, is introduced.

3.3.1 Architecture overview

The overall pipeline of the human action recognition module is shown in Fig. 3.6. The skeleton extraction stage (including the training step) works in the 2D space by using, as input, only RGB data. Extending the work in [21], our method further segment the extracted 2D skeleton into two regions: an upper part U that includes torso, arms, and head; and a lower part L that includes hips and legs (see Fig. 3.7a). This subdivision has two main purposes. First, in learning and classification steps it supports the reduction of the feature space with respect to an entire network, therefore improving the overall performance of the method. Indeed, most complex human actions are *structured*, since they are composed of motions which take place within the upper and lower body parts; explicitly injecting this knowledge in the learning process has a stabilizing effect on the energy landscape at training time, and poor local minima are more easily avoided. Notice that a similar approach was also taken in [37], although in the latter case the subdivision followed a 5-level hierarchy and it assumed the skeleton to be embedded in $\mathbb{R}^3$. Moreover, in all our tests, we did not observe significant advantages when using more than two parts. Second, according to real scenarios in which objects (e.g., cars, advertising signs) can hide parts of moving subjects, the subdivision supports an effective strategy for the body occlusions.

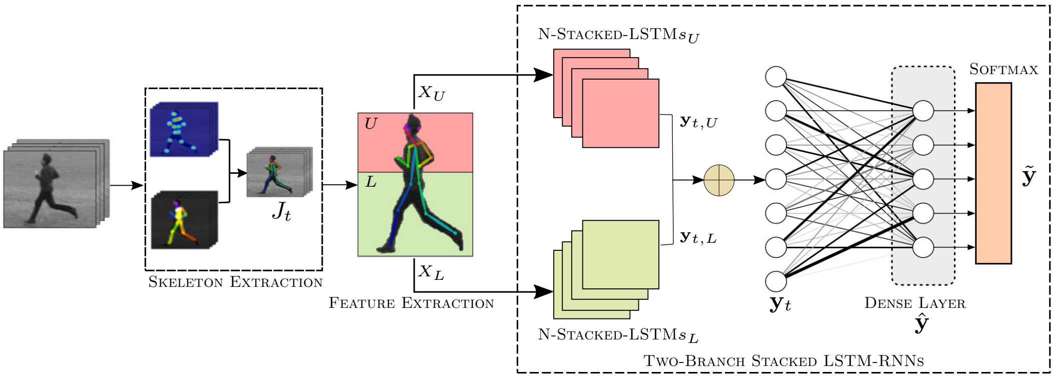


Figure 3.6. Architecture of the proposed method.

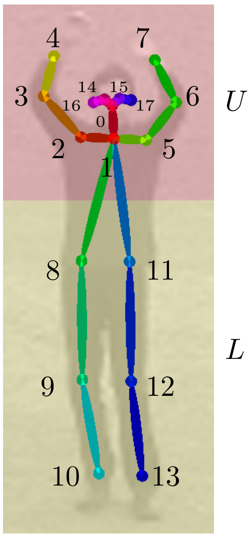
3.3.2 2D Skeleton and Feature Extraction

The skeleton extraction stage aims at estimating a 2D skeleton for each human subject that appears in an input video. This is obtained via a two-branch CNN to jointly predict confidence maps for body part detection and PAFs for parts association, an architecture recently proposed in [21] by which the skeleton structure of each subject in the scene is represented as a set J_t of 18 joint coordinates in $\mathbb{R}^2$. Let $J_t = \{j_i\}_{i=0}^{17}$ be the set of joints for a given 2D skeleton at time $t = 0, \ldots, T$; and, for a frame acquired at time t, let $\mathbf{x}_{t,U} \in \mathbb{R}^9$ and $\mathbf{x}_{t,L} \in \mathbb{R}^{12}$ be the feature vectors for the parts U and L, respectively, constructed as reported in Fig. 3.7b. We denote by $X_U = (\mathbf{x}_{0,U}, \ldots, \mathbf{x}_{T,U})$ and $X_L = (\mathbf{x}_{0,L}, \ldots, \mathbf{x}_{T,L})$ the ordered collections of upper and lower body part feature vectors, respectively, along the analysed sequence. The adoption of angles as features allows our model to capture the change in *posture* over time, while acceleration allows to capture the *speed* of the pose transition.

3.3.3 Two-Branch Stacked LSTM-RNNs

The proposed network is composed of two branches: an upper branch and a lower branch operating on the features X_U and X_L, respectively. Each branch is composed of N-stacked LSTM-RNNs to obtain a higher-level of abstraction on the input data [58]. In the last years, stacked LSTM models have already been used with remarkable results for speech recognition tasks [51, 147], where feature abstraction ranges from phrase- to syllable-level, and where each level of the hierarchy is captured at different scales within the considered time interval. This intuition can be invariably applied to video data depicting human actions, which can be regarded as motion (similarly to audio) sequences consisting of several small movements (the syllables in our analogy) making up the complete "visual sentence". Stacked LSTMs are therefore a suitable choice for this type of classification problem.

According to the formulation previously described in Section 3.2.4, each memory cell of the first layer $LSTM_{l=0,\theta}$ of each stacked LSTMs, where $\theta \in \{U, L\}$ denotes the branch, takes as input both X_θ and $\mathbf{h}_{l=0,t-1,\theta}$. While, each memory cell of a



(a)

Index	Feature
0	$\angle(j_1, j_2, j_3)$
1	$\angle(j_1, j_5, j_6)$
2	$\angle(j_2, j_3, j_4)$
3	$\angle(j_5, j_6, j_7)$
4	$\angle(j_8, j_1, j_{11})$
5,6	$\mathbf{a}_x(j_4)$, $\mathbf{a}_y(j_4)$
7,8	$\mathbf{a}_x(j_7)$, $\mathbf{a}_y(j_7)$
9	$\angle(j_1, j_8, j_9)$
10	$\angle(j_1, j_{11}, j_{12})$
11	$\angle(j_8, j_9, j_{10})$
12	$\angle(j_{11}, j_{12}, j_{13})$
13,14	$\mathbf{a}_x(j_8)$, $\mathbf{a}_y(j_8)$
15,16	$\mathbf{a}_x(j_{11})$, $\mathbf{a}_y(j_{11})$
17,18	$\mathbf{a}_x(j_{10})$, $\mathbf{a}_y(j_{10})$
19,20	$\mathbf{a}_x(j_{13})$, $\mathbf{a}_y(j_{13})$

(b)

Figure 3.7. Skeleton joints and linked features: (a) the 18 joints obtained by the 2D skeleton extraction phase and (b) the joint features used as input in the proposed method. The symbols $\angle(j_a, j_b, j_c)$ and $\mathbf{a}_{\{x,y\}}$ denote the angle formed by the joint j_b and the acceleration in the $\{x, y\}$ direction, respectively.

deeper layer, i.e. $l > 0$, takes as input both $\mathbf{h}_{l,t-1,\theta}$ and $\mathbf{h}_{l-1,t,\theta}$, the latter is the hidden state, at time t, of the previous $LSTM_{l-1,\theta}$ layer. The output of each is represented by the hidden states $\mathbf{h}_{N-1,t,\theta}$ of the last layer, which is indicated with the term $\mathbf{y}_{t,\theta}$.

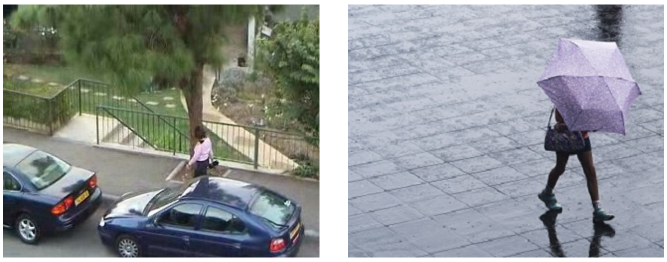
We have further added a fusion layer taking as input the outputs of the two branches $\mathbf{y}_{t,U}$ and $\mathbf{y}_{t,L}$, and yielding the fused prediction $\mathbf{y}_t = \mathbf{y}_{t,U} \oplus \mathbf{y}_{t,L}$, where $\oplus$ indicates the concatenation operator. Finally, each vector $\mathbf{y}_t$ is provided as input to a dense layer returning a single output vector $\hat{\mathbf{y}}$, to which in turn we apply a softmax activation to yield:

$$\tilde{\mathbf{y}}(k) = p(C_k | (X_U, X_L)) = \frac{e^{\hat{\mathbf{y}}(k)}}{\sum_{q=0}^{K-1} e^{\hat{\mathbf{y}}(q)}} . \tag{3.12}$$

Vector $\tilde{\mathbf{y}}$ contains the (normalized) probability for each class C_k, with $k = 1, \ldots, K$. The network is trained by using the cross-entropy loss and the RMSprop algorithm [38].

3.3.4 Partial Body Occlusion Management

In many real scenarios, we can only have a partial view of the body. In fact, subjects can be occluded by static or moving objects (e.g., cars) or they can be partially located outside the field of view of the camera. In most cases, these occlusions do not concern a single limb, but involve larger body parts, as shown in Fig. 3.8, where lower (Fig. 3.8a) and upper body parts (Fig. 3.8b) are completely covered. Moreover, there are some types of human actions that only require of partial body parts to be



(a) (b)

Figure 3.8. Examples of partial body occlusions in real scenarios: (a) lower part and (b) upper part.

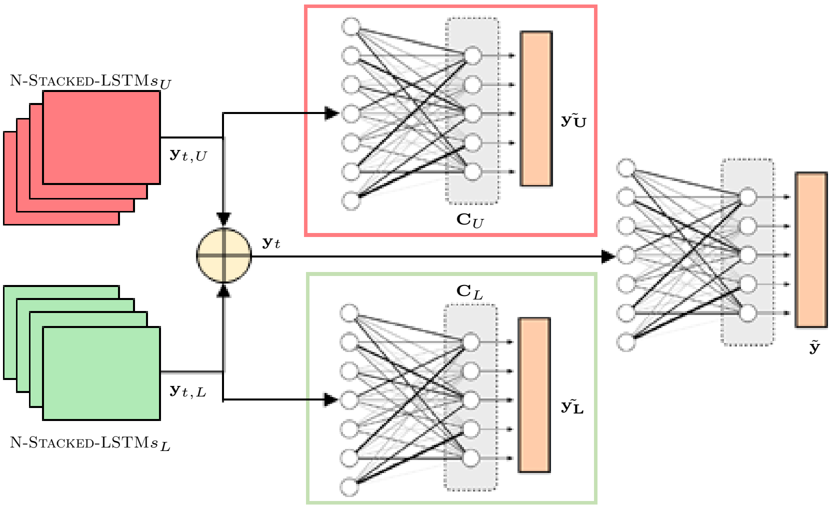
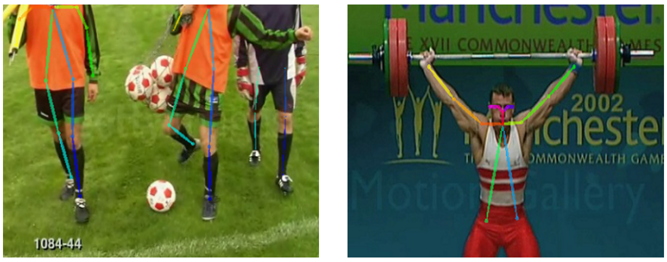


Figure 3.9. The C_U and C_L networks used to manage the partial body occlusions.

recognized, e.g., walking can be distinguished from running by using only leg features. Relying on the above analysis, we designed our architecture using two feed-forward networks, called C_U and C_L, consisting of a dense layer, with a ReLu activation function, and a softmax layer, to separately analyse each output $\mathbf{y}_{t,\theta}$ of a single branch (Fig. 3.9). Each C_θ is trained to recognize only a subset of actions related to θ using the cross-entropy loss and the RMSprop optimization algorithm. The output $\tilde{\mathbf{y}}_\theta$ denotes the normalized probability for each class that can be recognized by using only the branch θ. Then, depending on the occluded body part, the network flow is moved to the branch that manages the part of the body not occluded. If occlusions do not occur, then the network flow passes through both branches, following the full pipeline expressed in Sec.3.3.2. Several solutions in different application fields, ranging from action recognition [183] to person Re-identification [168], have been proposed to handle the issue of partial occlusion dividing the subject's body in multiple parts, thus splitting up the processing. Although these methods have shown accurate and robust results, they are only focused on RGB images. Differently, since our proposed method is based on 2D skeletal data, our ad-hoc strategy for occlusion management has been specifically designed for this type of input. In



(a) (b)

Figure 3.10. Examples of missing joint coordinates in: (a) lower and (b) upper body parts.

the proposed solution, a body part θ is considered occluded if more than 30% of the joints are missed in more than half of the analysed video frames, examples as reported in Fig. 3.10. We have not considered further subdivisions of the network in sub-branches because a single body limb does not provide enough information for classification purposes (even in the case of a limited set of actions).

3.3.5 Multiple Subject Identification

When more subjects are present in the scene, the skeleton extraction step returns multiple skeletons. This information is not propagated across video frames, thus making necessary the inclusion of a tracking algorithm, within the proposed pipeline, to reliably associate each skeleton to the linked subject. Examples of this step are shown in Fig. 3.11. The tracking algorithm proposed in this module extends the method reported in [204] allowing the tracking of multiple subjects. The extension is based on three main steps. First, for each frame of the video sequence a gray-scale image $b \in \mathbb{R}^{w \times h}$ (where h and w are height and width, respectively) is wrapped around each skeleton using a bounding box defined by the head/feet joint positions. Second, each image b is then convoluted with a set $z = \{z_{1,1}, \ldots, z_{w,h}\}$ of rectangular filters, at multiple scales, to capture possible scale variations of tracked subjects (e.g., due to the perspective). Each kernel $z_{i,j}$ assigns the value 1 to the pixel in position (x, y) when $0 \leq i \leq x$ and $0 \leq j \leq y$, and 0 otherwise. The resulting convolutions are reshaped as columns and concatenated into a feature vector $x_b = \{x_{b_1}, \ldots, x_{b_m}\}$, with $m = (wh)^2$. Finally, for compactness, the vector x_b is projected into a lower-dimensional vector $v_b = \{v_{b_1}, \ldots, v_{b_n}\} \in \mathbb{R}^n$ (with $n \ll m$) by means of a sparse random projector [204].

The low-dimensional representation v_b is, lastly, provided to a Naïve Bayes classifier [56] to estimate the correct label ϑ_b corresponding to a person $s \in \{1, \ldots, S\}$ in the scene:

$$\vartheta_b = \underset{s \in \{1,\ldots,S\}}{\arg\max}\, p(C_s|v_b) = \underset{s \in \{1,\ldots,S\}}{\arg\max}\, p(C_s)p(v_b|C_s) \tag{3.13}$$

where $p(C_s) = \frac{1}{S}$ and $p(v_b|C_s)$ is modeled as a Gaussian distribution:

$$p(v_b|C_s) = \frac{1}{\sqrt{2\pi\sigma_s^2}} e^{\frac{(v_b - \mu_s)^2}{2\sigma_s^2}} \tag{3.14}$$

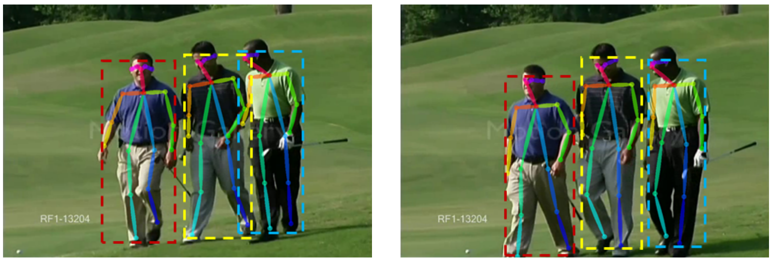


Figure 3.11. Example of multi-person skeleton tracking.

with σ_s^2 being the sample variance of class C_s and μ_s the corresponding sample mean. For $\vartheta_b = C_s$, the values for σ_s^2 and μ_s are updated according to:

$$\mu_s = \lambda\mu_s + (1-\lambda)\hat{\mu_s} \tag{3.15}$$

$$\sigma_s^2 = \sqrt{\lambda\sigma_s^2 + (1-\lambda)\hat{\sigma}_s^2 + \lambda(1-\lambda)(\mu_s + \hat{\mu}_s)}\,, \tag{3.16}$$

where λ is the learning rate (empirically set to 0.85 in our experiments), $\hat{\mu}_s$ is the mean of the v_b values, and $\hat{\sigma}_s$ is the corresponding variance. As shown in Fig 3.11, the output of this tracking step is a set of skeleton sequences, each of them with a different subject label.

3.3.6 3D-DenseNet

Extracting skeleton information from video sequences is not always possible, especially for those composed by face close-ups. To improve the robustness of our approach, with the aim to support the two-branch stacked LSTM in such case, a 3D CNN-type network is introduced into the pipeline. This support network, other than helping to manage missing skeletal data, allows us to perform feature extraction from the scene with which the subject is interacting. In detail, the chosen support network architecture is the 3D-DenseNet, an alternative version of the standard DenseNet [63], where all the 2D convolution and pooling layers are replaced by their 3D versions. Although this type of network could be less accurate in analysing the body movements, it does not depend on the skeletal data correctness, therefore appearing suitable in several contexts. The strength of the DenseNet derives from the use of dense blocks. In these blocks, the layers are densely connected together, receiving in input all previous layers output feature maps. A single layer of a dense block typically consists of: a batch normalization, a ReLU activation function, and a convolution (i.e., 3D convolution). A 3D convolution can be easily obtained by adding the temporal dimension into the input volume, the kernel, and the output volume of a 2D convolution. The input is represented with three dimensions: height, width, and depth. The latter is the number of images sampled from the video clip. The 3D pooling operation involves just adding the temporal dimension to the 2D pooling operation. In Fig.3.12, the overall architecture of a 3D-DenseNet with 3 dense blocks is shown. Let $\mathbf{y}_c$ be the output of the last 3D max pooling layer of our 3D-DenseNet, a new vector named $\dot{\mathbf{y}}$ is obtained from the concatenation between $\mathbf{y}_c$ and $\hat{\mathbf{y}}$, as shown in Fig.3.13. In case of partial occlusion (Sec.3.3.4), then $\dot{\mathbf{y}} = \tilde{\mathbf{y}}_\theta \oplus \mathbf{y}_c$ (e.g., if the lower part is occluded, then $\theta = U$). Notice that, in this combination

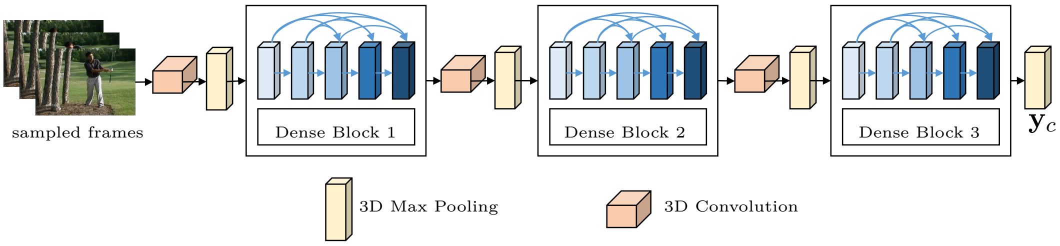


Figure 3.12. 3D-DenseNet architecture with 3 dense blocks.

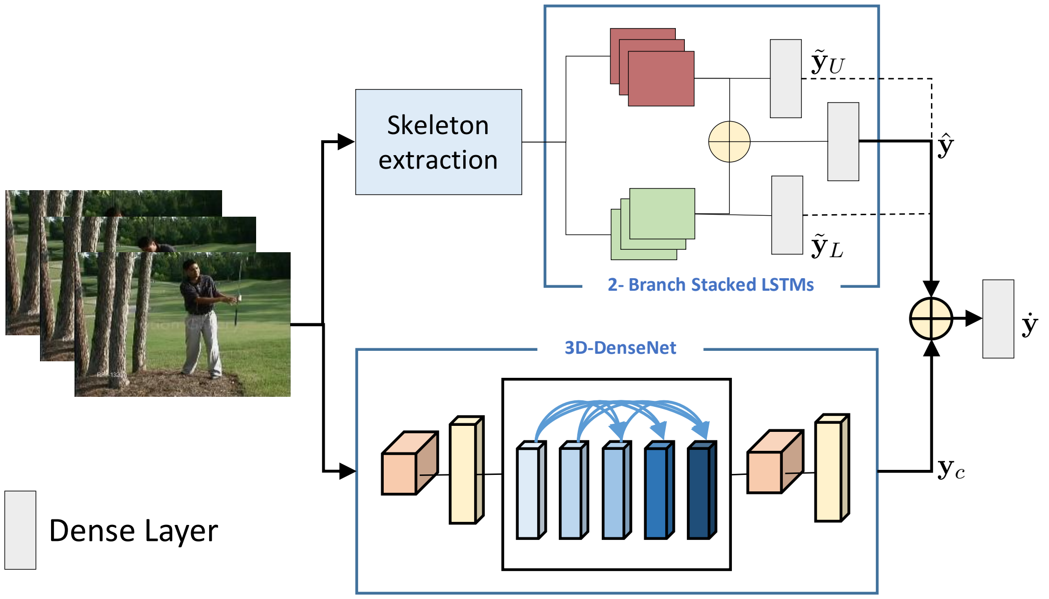


Figure 3.13. Integration of 3D-DenseNet within the proposed architecture.

with the 3D-DenseNet, the values contained in $\tilde{\mathbf{y}}_\theta$ do not need to be normalized through a softmax function. Since $\hat{\mathbf{y}}$ and $\tilde{\mathbf{y}}_\theta$ have different sizes, a simple zero padding operation will be applied to $\tilde{\mathbf{y}}_\theta$ before the concatenation. Next, the vector $\dot{\mathbf{y}}$ is given as input into a dense level and, finally, a softmax function is applied to normalize each class probability for the final classification.

3.4 Non-acted body affect recognition module

In this section, the pipeline for non-acted affect recognition by body movements is defined (Section 3.4.1). Initially, the implemented meta-features are a briefly introduced (Section 3.4.2). To follow, the procedure applied to create the temporal local and global feature sets is presented (Section 3.4.3). Finally, our two branch network used to classify the body movements, in order to recognize the affects, is described (Section 3.4.4).

3.4.1 Architecture overview

Ah shown in Fig. 3.14, the overall architecture is reported. The collection of temporal local feature vectors $\mathbf{V}_\alpha$ and temporal global feature vector $\mathbf{v}_\beta$ are extracted from a frame sequence $\mathbf{S} = \{\mathbf{s}_0, \ldots, \mathbf{s}_t, \ldots, \mathbf{s}_{T-1}\}$, representing joints rotations of a 3D skeleton for each time instant $t \in [0, T-1]$. Afterwards, $\mathbf{V}_\alpha$ and $\mathbf{v}_\beta$ are given as inputs to the two branches of the proposed network, based on stacked LSTMs and MLP, respectively. Finally, the output vectors of the two branches, $\mathbf{z}_\alpha$ and $\mathbf{z}_\beta$, are combined and used to obtain the normalized vector $\hat{\mathbf{y}}$ containing the classification.

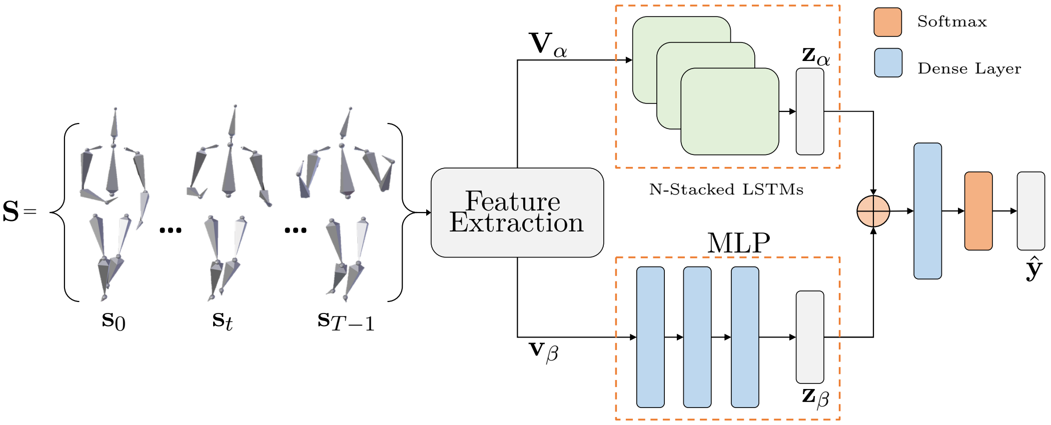


Figure 3.14. Architecture of the proposed non-acted body affect recognition method.

3.4.2 Feature description

In order to have an in-depth description of body movements, both classical measures such as velocity, acceleration, jerk, and snap, as well as more complex meta-features, are computed starting from the available joints.

All classical measures are derived directly from the 3D skeleton rotations, and are used to describe movements variations along all the available joint axes (e.g., vertical snap of a wrist, horizontal elbow acceleration, etc.). While simpler with respect to the second category (i.e., meta-features), these quantities allow the system to describe meaningful, as well as sudden, changes related to the body movement. For example, a person winning a game might quickly raise his arms, resulting in the corresponding joints (e.g, wrist, elbow, etc.) having high acceleration.

The *meta-feature* group, is based on [44]. The term meta-feature is retained from previous works [44, 45] in order to maintain a continuity with the available literature. Nonetheless, new features, computed over the proposed procedures, were introduced to further enrich the already existing studies. In the following paragraphs, a description of the available meta-features used in this manuscript, as well as what aspects they can represent for a specific emotion, are given.

Symmetry and Limb Alignment

This meta-feature can estimate asymmetries through three different algorithms: a *pose symmetry algorithm*, used to estimate joint misalignment; a *pose difference* *algorithm*, to return the mean misalignment of corresponding joints among the two sides of the body; and a *directed symmetry algorithm*, to reflect asymmetries directions. The first two algorithms can be good indicators of a person composure. For example, a calm person can retain a symmetric and aligned body posture more easily with respect to an angry one. The third algorithm can give hints on the body asymmetry direction, namely toward which direction the body is oriented. If a cheerful or frustrated person is considered, the general body limb asymmetry direction would typically be upwards and outwards (e.g., straight arms raised in opposite ways), or downwards and inwards (e.g., arms bent toward either the chest or knees), in accordance with the emotion being felt.

Concerning the actual computation of meta-features themselves, the first algorithm (i.e., pose symmetry) calculates the ratio between horizontal and vertical symmetries. These symmetries are computed over the X and Y axes of the chosen limbs, with respect to a central point. For example, chest X and Y axes are used to compute, respectively, the horizontal and vertical symmetry of the other joints. Two examples are shown in Fig. 3.15a and Fig. 3.15b. To calculate the second algorithm (i.e, directed symmetry), the pose symmetry is first computed and its direction (i.e., sign) is changed depending on which limb results farther away, according to the Euclidean distance, from the central point. This choice effectively allows the farthest limb to guide the asymmetry direction. Finally, for the third algorithm (i.e., pose difference), an average of the Euclidean distances between corresponding joints is returned. For example, if the pose difference between the arms is being considered, the average of the distances between left and right shoulders, elbows, and wrists, is returned.

Head Alignment and Offset

This meta-feature compares head and chest rotations, and measures the relative location among head and hips to understand both head and body tilt. These quantities can help discern whether the person is feeling a submissive emotion (e.g., an embarrassed person tends to slightly tilt her head and body downwards) or a dominant one (e.g., a contemptuous or prideful person generally stands straight with a raised head, resulting in low tilt).

In order to obtain this meta feature, three different values are computed: *head offset*, represented by the Euclidean distance between head and hips X, Y, and Z axes; *head alignment*, described by comparing the distance between head and hips X, Y, and Z rotation angles; and *head-chest ratio*, representing the average among the X, Y, and Z ratios computed using head and chest joints (i.e., allows to understand how head and chest are inclined with respect to one another).

Body Openness

In this thesis, we call *Body Openness* an extended version of the *Leg-Hip Openness* meta-feature presented in [44]. The name “Body Openness” was chosen as it better represents the extended meta-feature version, which is devised to also handle the upper body following the studies in [109, 4]. The original meta-feature, leg-hip openness, examines the distance between the location of each leg, representing lower

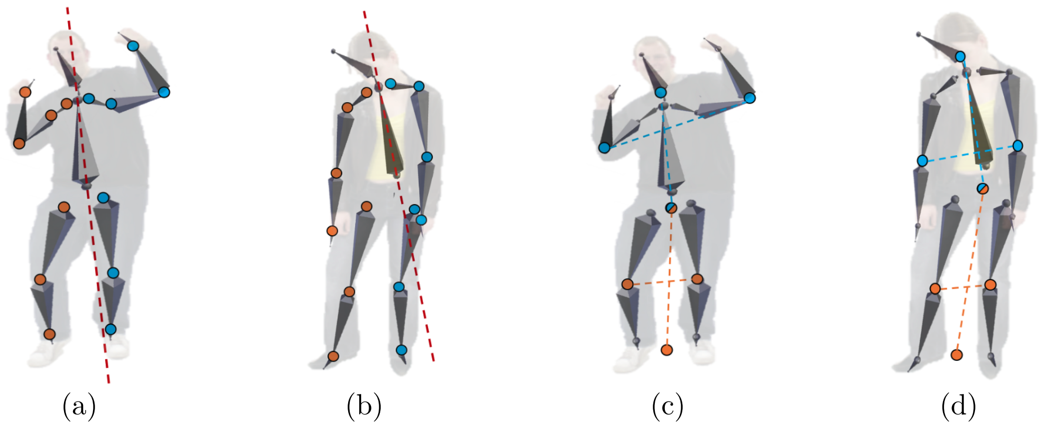


Figure 3.15. In the image (a) and (b) show examples of body symmetry, where left and right limbs joints are represented by blue and orange dots, respectively; (c) and (d) represent upper body openness with high and low ratios, where upper and lower joints are represented by blue and orange dots, respectively.

body openness, and the ankles distance from the hips, describing how much the torso is bent. In the improved version used in this work, the same rationale is applied to the upper body, in order to understand whether a person has a broaden out chest. To capture this posture aspect, the distances between arms locations are examined by also taking into account the neck-hips distance. Similarly to the Head Alignment and Offset meta-feature, the body openness can highlight power poses by analysing both the lower and upper halves of the body. For example, a prosecutor (i.e., a confident person) would have an open chest while questioning witnesses. On the other hand, a student during an oral examination would have a closed chest being, most likely, stressed out by the situation.

These quantities can be computed as follows: regarding the lower body half openness, the ratio between hips-ankles and left/right knees distances is calculated. Low ratios indicate bent legs and torso, corresponding to a crouched position (e.g., a person cowering in fear); mid-range ratios denote straight open legs and unbent torso, representing a power pose in the lower half of the body (e.g., a person showing confidence); high ratios correspond to straight closed legs and unbent torso, meaning the person is simply standing (e.g., a person demonstrating calm). Concerning the upper body half openness, the ratio between neck-hips and elbows distances is computed. Intuitively, low ratios indicate a power pose in the upper half of the body (e.g., a confident person with a broaden out chest as shown in Fig. 3.15c); mid-range ratios correspond to straight arms (i.e., elbows close to the chest) and straight torso (e.g., a calm or sad person in a neutral pose as shown in Fig. 3.15d); while high ratios denote open elbows and bent torso (e.g., a person cowering in fear protecting the head).

Average Rate of Change

This meta-feature calculates the average rate of change of features (e.g., the speed of a moving shoulder). This measure allows to understand how much movement a given joint is undergoing when an emotion is being felt, and can help discriminate static emotions from dynamic ones. For example, a sad person would hardly jump around, while a cheerful one is likely to either quickly raise the arms, or walk faster than normal. In order to obtain this quantity, the ratio between the sum of the changes of a given feature (e.g., wrist position) and the number of frames analysed (e.g., 20 frames) is computed. Once elaborated, this measure can give human behaviour cues in association with a specific affect, due to its ability to summarize body postures changes.

Relative Movement

This meta-feature represents the level of change of a feature at a given moment (e.g., an average of the arm movement). To obtain this meta-feature, the ratio between the average rate of change of a key posture frame neighbourhood, and the average rate of change over the recording for that given feature, is computed. Intuitively, the relative movement of a feature expresses how much that specific feature (e.g. right knee) generally changes. For example, an angry person usually has an high relative movement associated to both arms and legs.

Smooth-Jerk

This meta-feature computes the body position change frequency as the relative variance of a feature over time. (e.g., how many times a leg position changes). In order to compute this value, the ratio between the variance of a key posture frame neighbourhood, and the variance over the recording for that given feature, is computed. While similar to the relative movement, the smooth-jerk meta-feature allows to grasp the effective value of change for a given feature by leveraging its variance. In more details, if the relative movement can measure the total amount of change, the smooth-jerk is able to grasp whether changes happened in a fast or slow way.

Location of Activity

This meta-feature encodes the movement rotation of specific body part features with respect to the other ones (e.g., how much an arm is moving in relation to the whole body). To compute this value, the ratio between the relative movement of a feature and the sum of relative movements of all available features, is computed. This feature helps describe which body parts are being moved the most. For example, a furious person with arms flailing about, would have high location of activity values for the arms joints.

Extra Features

In this work, beyond implementing all meta-features found in [44], we introduced various new ones based on those presented in the previous paragraphs, in order to better describe the body affects. In more details, we computed the symmetry and limb alignment (Sec. 3.4.2) applied on the arms using both 3 (as in [44]) and 4 joints (e.g., wrist, elbow, shoulder, and collar). This decision was taken since shoulder and collar allow us to describe the arm movement along different axes. Using a fourth joint extends the representational power of the *symmetry and limb alignment* meta-feature group, while still closely representing arm movements since collar and should joints are close to one another. For example, an uneasy person with straight arms could have a closed chest, which can be captured by also using the collar as an arm joint.

Moreover, the leg-hip openness in [44] was modified into the *body openness* meta-feature. In this proposed version, beyond legs openness and torso bending degree, upper half body cues can also be captured by outlining chest openness (e.g., self-confident people have a broaden out chest) as explained in Sec. 3.4.2.

Finally, further features were also generated starting from the *location of activity* group (Sec. 3.4.2). These meta-features were used to obtain the activity location of head, arms and legs, contrary to [44] which computes this feature only for the head. In this module we decided to exploit activity information related to all limbs, since different emotions tend to have different reactions on the body posture as well as on its movement. For example, if a cheerful and a sad person are considered, they will have, respectively, high and low activity values on the limbs since the first person will tend to move more (being happier) than the second one.

3.4.3 Feature extraction

In this module, both classical as well as meta-features can be divided into two main categories: *temporal global features* and *temporal local features*. In this feature extraction step, both versions (i.e., global and local) of all classical features (i.e.,velocity, acceleration, jerk, and snap) as well as each of the presented meta-features, are implemented. By leveraging both representations, the model effectively computes a coarse summary as well as a fine-grained and time-evolving representation of body movements. The final result of this feature extraction module is represented by a collection of vectors $\mathbf{V}_\alpha$ and a single vector $\mathbf{v}_\beta$, i.e., temporal local features and temporal global features, respectively. Thorough descriptions of both categories are provided in the following sections.

Temporal Global Features

This group of features is computed over the whole input sequence $\mathbf{S}$. It represents a global overview of body movements that takes into account time flow, hence the temporal global feature (TGF) name we devised. Intuitively, given a sequence containing a body affect, this feature group represents both classical and meta-feature measures a body has during that specific affect. For example, how often a cheerful person moves his arms or legs, the average velocity of the body, and so on.

In order to generate these features, a summary of the motion is computed by averaging the values obtained from the defined features (i.e., both classical and meta), when analysing a given sequence. This procedure effectively generates a vector of features that is able to resume various aspects of the movement itself. In more details, given the set of available features F and a sequence S, the temporal global features are computed as follows:

$$TGF_f = \frac{1}{n}\sum_{t=0}^{n} f(s_t), \quad \forall f \in F \tag{3.17}$$

where n represents the length of the sequence S; $s_t \in S$ are single frames of the sequence; and $f(\cdot)$ is a function returning the value for any feature in F, on a given frame s. By leveraging these temporal global features, the proposed method is able to better grasp generic movements associated to given affects, contrary to [44] that focuses, instead, on single postures.

Temporal Local Features

This second group of feature is computed on each 3D acquisition $\mathbf{s}_t \in \mathbf{S}$. It describes the variation of body movements as time flows, hence the temporal local feature (TLF) name we devised. In this group, beyond classical and meta-features, also joints positions and rotations are analysed. Intuitively, via this feature group it is possible to grasp small variations along the defined dimensions (i.e., all available features) by comparing the motion step-by-step. For example, the shoulder movement of a sad, sulking person would be described, and better define the affect.

In more details, given the set of available meta-features F_{meta} and a sequence S, the temporal local features of meta-features (TLM) are computed as follows:

$$TLM_{f,t} = f(s_t) - f(s_{t+1}), \quad \forall f \in F_{meta} \tag{3.18}$$

where $\mathbf{s}_t$ and $\mathbf{s}_{t+1}$ represent two consecutive frames, with $\mathbf{s}_t, \mathbf{s}_{t+1} \in \mathbf{S}$ and $\forall t \in [0, T-2]$. While, given the set $F_{classic}$, containing positions, rotations, as well as classical features (i.e., velocity, acceleration, jerk, and snap), and a sequence S, the temporal local features of classical features (TLC) are computed as follows:

$$TLC_{f,t} = f(s_t), \quad \forall f \in F_{classic} \tag{3.19}$$

where $s_t \in S$ represents a single frame of the sequence; and $t \in [0, T-2]$ indicates a frame at a given time step. Finally, temporal local features are defined as:

$$TLF = TLM \cup TLC. \tag{3.20}$$

By leveraging these temporal local features, the proposed method is able to better grasp detailed movements associated to given affects, further extending the ability to distinguish different emotions.

The final result of this feature extraction step is represented by a collection of vectors $\mathbf{V}_\alpha$ and a single vector $\mathbf{v}_\beta$, i.e., temporal local features and temporal global features, respectively.

3.4.4 Proposed network architecture

The proposed network is composed of two branches: a local branch, based on stacked LSTMs, and a global branch, based on a MLP, operating on the features $\mathbf{V}_\alpha$ and $\mathbf{v}_\beta$, respectively. The local branch consists of N-stacked LSTMs to obtain a higher-level of abstraction on the input data [58]. The choice to use the LSTM derives from the remarkable results obtained in several issues regarding the analysis of body movements, such as gesture recognition [6] or action recognition [167].

According to the stacked LSTM architecture (Sec. 3.2.3), each unit of an $LSTM_l$, at time t, takes as input a vector $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{l,t-1}$. Where the given input $\mathbf{x}_t$ indicates a temporal local feature vector $v_t \in V_\alpha$ if $l = 0$ (i.e., the first level of the stack), otherwise it represents the hidden vector of the underlying layer $\mathbf{x}_t = h_{l-1,t}$ (i.e., for layers higher than the first one). The output vector $\mathbf{z}_\alpha$ of the local branch is represented by $\mathbf{h}_{N-1,T-1}$, namely the hidden state of the last layer $N - 1$ at the last time instant $T - 1$ for the analysed time window.

Regarding the global branch, it is composed by a MLP with 3 hidden layers, where each hidden node transforms its input, obtained from the weighted sum of the output values of the previous layer, with a rectified linear unit (ReLU) activation function. The first hidden layer has weighted connections with the temporal global feature vector $\mathbf{v}_\beta$, which represents the input layer. The number of hidden nodes in each layer (as well as the number of output nodes) is smaller than the number of input entries, in this way the MLP is used to extrapolate highly significant patterns from the input, mapping $\mathbf{v}_\beta$ into a low-dimensional description represented by the output layer vector $\mathbf{z}_\beta$.

In the last part of the network, the $\mathbf{z}_\alpha$ and $\mathbf{z}_\beta$ vectors are combined in a new vector called $\mathbf{z}$, using the concatenation operator. Afterwards, a dense layer using a ReLU activation function is applied, thus connecting each entry value in $\mathbf{z}$ to an entry of the output vector $\mathbf{y}$ via a weight. This layer is used to map the vector $\mathbf{z}$ to a number of output nodes equal to the size of the set of affects to be recognized. The size of this set is indicated with the value K. The final classification $\hat{\mathbf{y}}$ is obtained by applying to $\mathbf{y}$ a softmax regularization:

$$\hat{\mathbf{y}}(k) = \frac{e^{\mathbf{y}(k)}}{\sum_{q=0}^{K-1} e^{\mathbf{y}(q)}} \,. \tag{3.21}$$

Finally, the proposed network is trained using the cross-entropy loss and the RMSprop optimization algorithm [38].

3.5 Conclusions

In this chapter, the proposed framework was presented, explaining the implementation details. For each module, the features given as network input were indicated and motivated. Since the inputs are mainly data sequences, the use of stacked LSTMs was fundamental for the realization of each method, although in the action recognition and body affect recognition modules, it is possible to note how it was necessary the support of other DNN architectures. In fact, the MLP was used to analyze the global features obtained from the movements of the body in Section 3.3, while, in Sec. 3.4, the CNNs (specifically the 3DCNNs) allowed to analyse the surrounding environment and the object details with which people interacted within the scene.

Chapter 4

Test and evaluation

This chapter presents the experimental phases of all the proposed framework module and it is structured as follows. In Section 4.1, the results obtained in the classification of hand gestures are introduced. In Section 4.2, tests and analysis performed of the action recognition module are reported. In Section 4.3, the experiments carried out in the non-acted affect recognition module are described. Finally, in Section 4.4, a summary discussion on the experiments of the framework modules is summarized.

4.1 Hand gesture recognition experiments

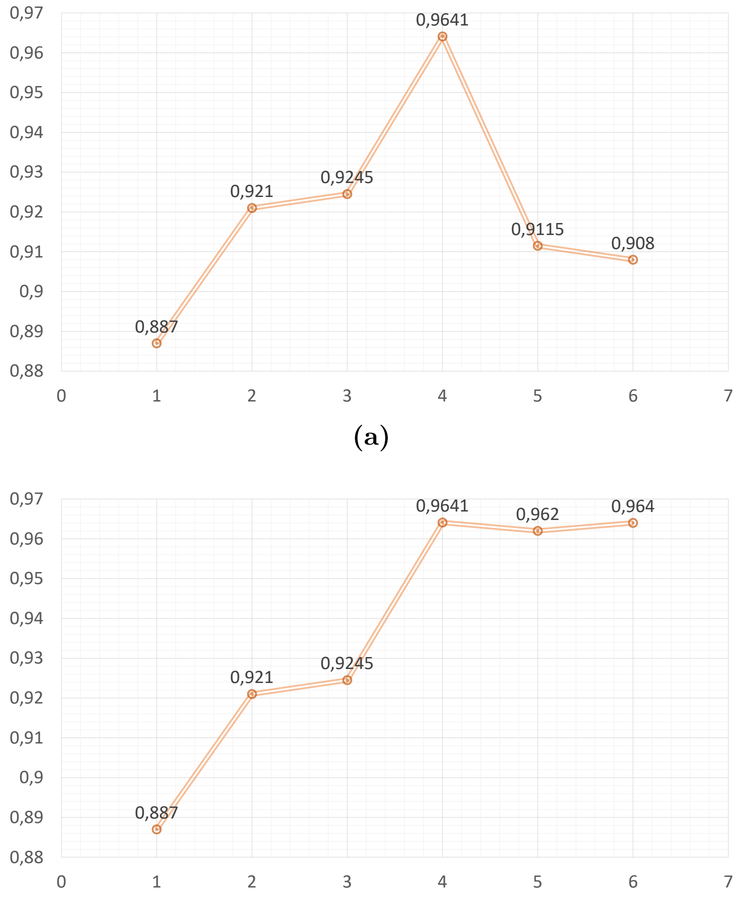
This section describes the experimental tests performed to evaluate the performance of the proposed Hand gesture recognition module. All the experiments were executed by using a LMC on an Intel i5 3.2GHz, 16GB RAM, with a GeForce GTX 1050ti graphics card. The stacked LSTMs and the BPTT algorithm, used to compute the minimization based on the stochastic gradient descent, were implemented by using the Keras[1] framework.

The main aims of the experimental session were both the validation of the proposed method, including the assessment of the joint angles as salient features for the hand gesture recognition, and the outperforming of competing works of the current state-of-the-art. The achievement of the first goal was obtained by creating a challenging dataset based on the sign language (Section 4.1.1) on which the optimal number of stacked LSTMs (Section 4.1.2) and the effectiveness of the selected joint features (Section 4.1.3) were analysed. In addition, on the same dataset, a set of well-known metrics was computed to evaluate the overall performance of the approach (Section 4.1.4). Instead, the second goal was obtain by comparing the proposed method with other considerable works on the basis of the SHREC dataset (Section 4.1.5).

4.1.1 ASL Dataset

Currently, there are no public datasets, with a large number of classes and with information on the hand joints, that allow to test approaches like that we propose.

[1] https://keras.io/



(b)

Figure 4.1. Accuracy results on the proposed dataset by varying the number of stacked LSTMs in the network architecture: (a) accuracy results using 800 epochs for each considered architecture and (b) accuracy results using 800 epochs for one LSTMs, 2-stacked LSTMs, 3-stacked LSTMs, and 4-stacked LSTM; for the 5-stacked and 6-stacked LSTMs are used 1600 and 1800 epochs, respectively. The x-axis indicates the number of the stacked LSTMs, while the y-axis indicates the accuracy values.

For this reason, we created a new dataset[2] composed of 30 hand gestures. In particular, the dataset consists of 12 dynamic gestures and 18 static gestures taken by the ASL. These gestures were chosen to stress the ability of the method in learning the variations of both joint angles and finger positions that occur when a hand performs a complex hand gesture. The static gestures are: 1, 2-V, 3, 4, 5, 6-W, 7, 8, 9, A, B, C, D, H, I, L, X, and Y. Instead, the dynamic gestures are: bathroom, blue, finish, green, hungry, milk, past, pig, store, and where.

The dataset is composed of 1200 hand gesture sequences, coming from 20 different people. Each gesture was collected by 15 males and 5 females, aged 20 to 28 years. Each person performed the 30 different hand gestures twice, once for each hand. The sequences from 13 people were used to create the training set, while the sequences of

[2] `https://bitbucket.org/visionlab-sapienza/2018-jrl-ieee-tmm_-application-dataset/src`

	ω_i	β_i	γ_j	u_w, v_w, z_w	ω_i, β_i	$\omega_i, \gamma_j, \beta_i$
Accuracy%	62.70%	68.1204 %	46.67%	56.92%	79.74%	85.13%

Table 4.1. Accuracy of the proposed solution obtained on the ASL dataset by varying the different features, where $0 \leq i \leq 4$, $0 \leq j \leq 3$, and $0 \leq w \leq 5$.

Accuracy	Precision	Recall	F1-Score
96.4102%	96.6434%	96.4102%	96.3717%

Table 4.2. Performance of the method on the ASL dataset using Accuracy, Precision, Recall, and F1-Score metrics.

the remaining 7 people were used to form the test set. So, the 7 people used in the tests were never taken into consideration during the training phase. As previously described in Section 3.2.3, each sequence was acquired according to a sampling process, with $T = 200$ and $\Theta = 5s$.

4.1.2 Selection of the Optimal Number of Stacked LSTMs

Several tests were conducted to choose the optimal number of stacked LSTMs needed to be used in the proposed architecture. The hidden units per LSTM were fixed to 200, i.e., the hidden units were fixed equal to the number of input time instances considered for each gesture (i.e., $T = 200$). Fig. 4.1a shows as an architecture composed by 4 levels provides the best accuracy results by using 800 epochs. In fact, although several levels of an LSTM allow to analyse complex time sequences by dividing them into multiple time scales, the $5 - LSTM$ and the $6 - LSTM$ require more epochs to be trained. Increasing the number of epochs needed to train the $5 - LSTM$ and $6 - LSTM$ architectures (i.e., 1600 epochs for the $5 - LSTM$ and 1800 for the $6 - LSTM$), the Fig. 4.1b shows how their results improves. We can notice how greater abstraction on input does not provide substantial benefits from a certain number of levels, and the accuracy gained by the network begins to converge to a fixed value.

In conclusion, 4 levels are appropriated for the proposed network and represent

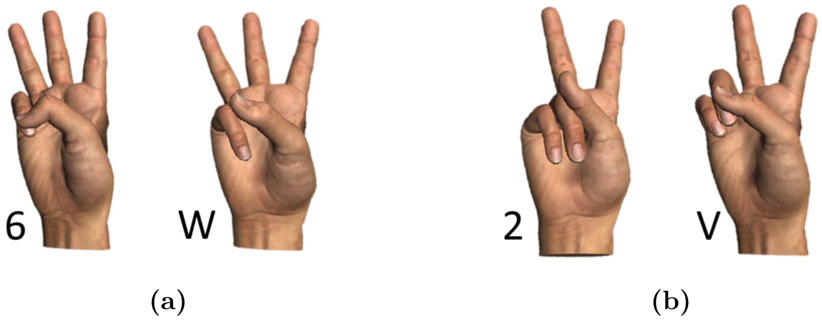


Figure 4.2. Pairs of gestures joined into a common class: (a) 6 and W hand gestures, 2 and V hand gestures.

a good compromise between training time and accuracy. The choice of the learning rate influences the speed of the convergence of the cost function. If the learning rate is too small, the convergence is obtained slowly, while if the learning rate is too large, the cost function may not decrease in each iteration and therefore it could not converge. In the proposed method, the learning rate was set to 0.0001 through large empirical tests.

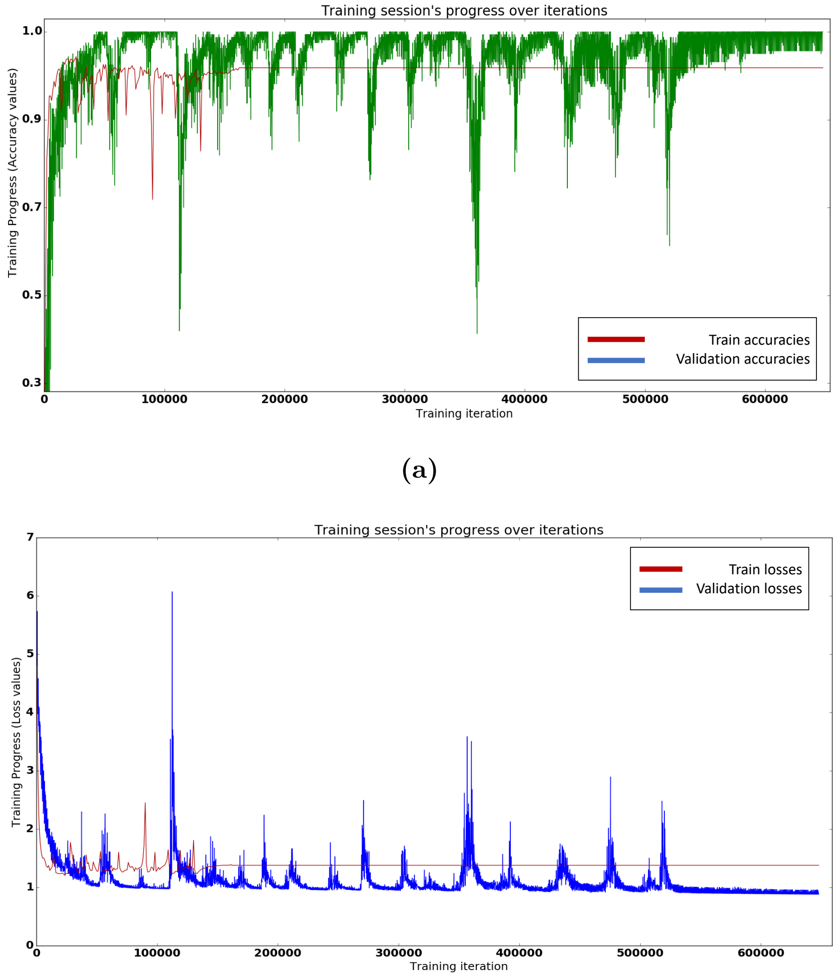
4.1.3 Effectiveness of the Selected Features

To verify the effectiveness of the features in classifying the set of gestures taken by the ASL dataset, various tests were carried out. The adopted network was composed by 4 stacked LSTMs (as explained in Section 3.2.4). Two different sets of hand gesture sequences of the ASL dataset (i.e., a set for the training and a separate set for the classification, respectively) were used to check the contribute of subsets of x_t as features.

The results in Table 4.1 shows that the combination of the features ω_i and β_i is sufficient to discriminate a high number of hand gestures. Notice that, this combination reaches better classification results with respect to the use of these two features separately. Although the single γ_j feature does not offer good performance, it greatly improves the classification when used with ω_i and β_i. Instead, the combination of features related to the hand movements (u_w, v_w, z_w) are unable by themselves to classify the hand gestures but, if combined with the features of the joint angles, allow the method to achieve high performance (as discussed below).

4.1.4 Hand Gesture Recognition on the ASL Dataset

To evaluate the method, we used very popular metrics: *Accuracy*, *Precision*, *Recall*, and *F1-score*. These metrics can be considered a de facto standard to measure the quality of this class of algorithms [162]. The results are presented in Table 4.2. According to the tests performed to recognize the different hand gestures, and to better analyse the proposed method, also the confusion matrix was computed (Fig. 4.9). Each column of the matrix represents the instances in a predicted gesture, instead each row represents the instances in a current gesture. The main diagonal of the matrix represents the instances correctly classified by the stacked LSTMs. The elements below the diagonal are the false positives, i.e., the gestures that are incorrectly classified within a class of interest. The elements above the diagonal are the false negatives, i.e., the gestures incorrectly classified as not belonging to a class of interest. The distinction of some gestures is very hard, since they are very similar to other gestures in the dataset. Despite this, the proposed method does not suffer of ambiguity issues. The only exceptions are given by the gestures 6 with W (Fig. 4.2a) and 2 with V (Fig. 4.2b). The variations in their joint angles are minimal and difficult to see even to the human eye. Moreover, the LMC device fails to capture these variations. For this reason, these gestures have been grouped in the same class. To quantify the difficulty in recognizing these gestures, we have also performed tests without grouping these classes, thus obtaining an accuracy of 91.5178%. This decrease, with respect to the overall accuracy of 96.4102%, is due to the incorrect classifications related to the classes: 2, 6, v, and W. In Fig. 4.3, the Train/Validation plots are shown. The first plot (Fig. 4.3a) shows the Train/Validation accuracy over the iterations, instead the second plot (Fig. 4.3b) contains the loss curves that represent the sum of the errors provided for each training or validation instance. In this work, the loss curves are calculated as maximum-likelihood loss function, described in Section 3.2.5. Instead, the curves of accuracy represent the training or validation instances correctly recognized. After a certain number of iterations ($\sim$ 125000), the validation accuracy curve converges.



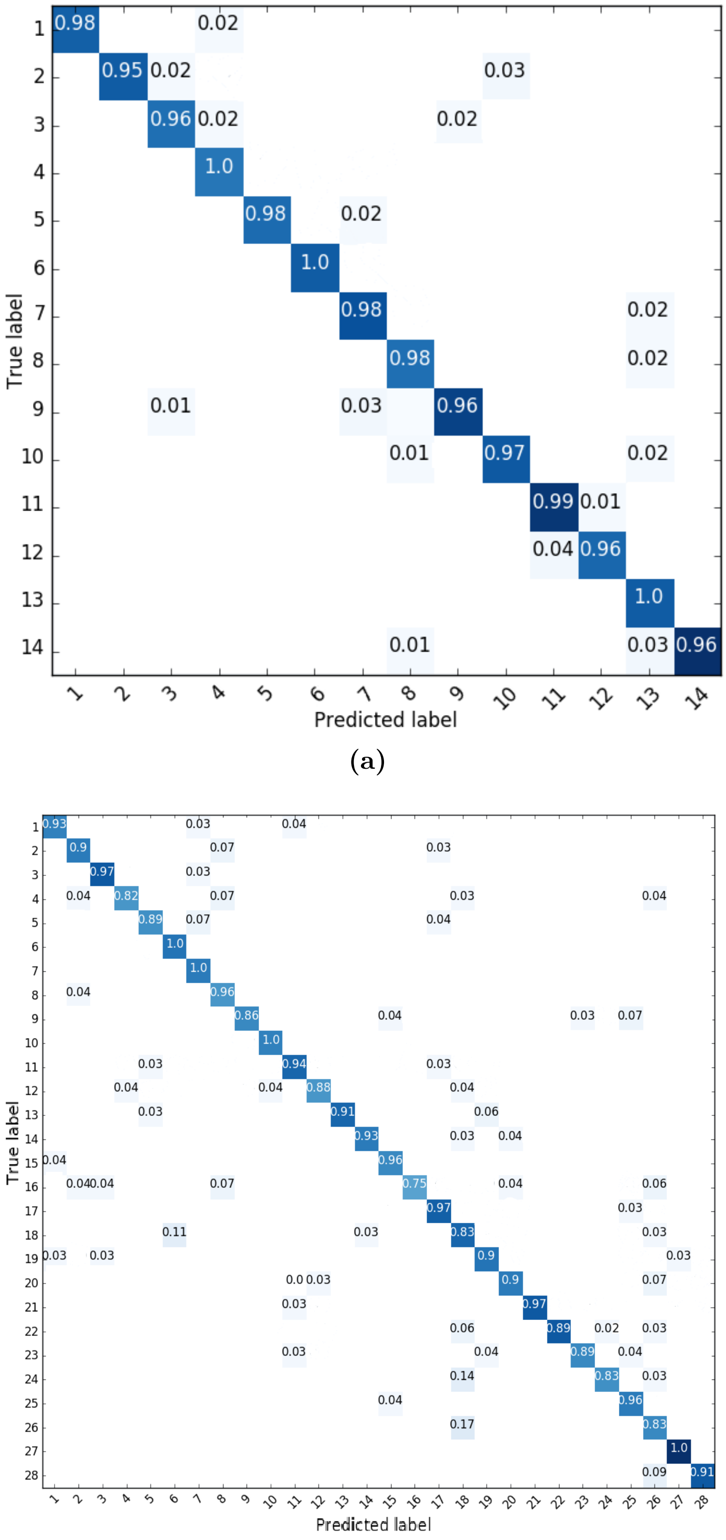
(a)

(b)

Figure 4.3. Pairs of gestures joined into a common class: (a) 6 and W hand gestures, 2 and V hand gestures.Train/Val Curves: the progress of training and validation over the iterations based on: (a) the accuracy and (b) the loss. After 100000 iterations, the test accuracy curve converges. The x-axis represents the progress of training/validation stage and the y-axis represents the number of training iterations.

4.1.5 Comparisons

We compared the proposed method with significant works of the current state-of-the-art presented in [160, 36, 125, 121, 34] on the SHREC dataset [34]. The SHREC



(a)

(b)

Figure 4.4. Confusion matrices obtained on the SHREC dataset: (a) with 14 hand gestures and (b) with 28 hand gestures.

Features	Accuracy 14 Gestures	Accuracy 28 Gestures
Proposed Method	**97.62%**	**91.43%**
De Smedt et al. [160]	88.24%	81.90%
De Smedt et al. [34]	82.90%	71.90%
Ohn-Bar et al. [121]	83.85%	76.53%
HON4D [125]	78.53%	74.03%
Devanne et al. [36]	79.61%	62.00%

Table 4.3. Comparison of the accuracy measure among significant state-of-the-art approaches on the SHREC dataset.

dataset was selected since: (a) it provides different types of data to allow comparisons between methods based on different acquisition sensors; (b) it allows the classification of hand gestures with different degrees of complexity; (c) it provides data that allow to extract all the features necessary for the proposed method. Notice that, the SHREC dataset contains very challenging semaphoric hand gesture sequences, and the evaluation of the method on this type of gestures is one of the main targets of the presented work.

The SHREC dataset is composed by 14 dynamic hand gestures performed by 28 participants (all the participants were right-handed). The hand gestures were captured by the Intel RealSense short range depth camera. Each gesture was performed between 1 and 10 times by each participant in two ways: using one finger and the whole hand. Therefore, the dataset is composed by 2800 sequences. The depth image, with a resolution of $640x480$, and the coordinates of 22 hand joints (both in the 2D depth image space and in the 3D world space) were saved for each frame of each sequence in the dataset. For the proposed method, we only needed of the 3D coordinates of the joints from which we derived the features of our interest. The depth images and hand skeletons were captured at 30 frames per second (fps) and the length of the sample gestures ranges from 20 to 170 frames. Since some sequences of the dataset are very short, to avoid a sampling with a very low T value, we used the padding technique to increase the length of these sequences to an acceptable value of T (i.e., $T = 100$). As shown in Table 4.3, the proposed method outperforms the accuracy values of the other works both in the dataset divided into 14 hand gesture classes and in the dataset divided into 28 hand gesture classes.

The confusion matrices related to the tests are shown in Fig. 4.4. By analysing these matrices, it can be observed that the method can accurately classify the hand gestures made by using only one finger, instead, when these gestures are made by using the whole hand, some mismatches can occur. In detail, the gesture 16 ($SWIPE$ $LEFT$) is, sometimes, erroneously classified, while, the gesture 18 ($SWIPE$ UP) can be confused with the gesture 26 ($SWIPE$ V). By carefully analysing the variations of the feature values, it is noticeable that the angles obtained from these instances are similar, moreover the movements of the hand in space are not substantial. In addition, some of these sequences are composed of few frames. Despite these isolated cases, we can state that the proposed method achieves excellent performance. This result demonstrates how the stacked LSTMs and the selected features are a very powerful solution in recognizing different types of challenging hand gestures.

4.2 Action recognition experiments

In this section, exhaustive action recognition experiments on a wide range of settings and conditions are reported. All the experiments were performed on an Intel i7 2.80 GHz CPU with 16GB RAM and a GeForce GTX 1070 graphics card. The two-branch network and the RMSprop algorithm were implemented by using the Keras framework and the TensorFlow back-end[3]. Each LSTM of the proposed architecture was realized by using 50 units, based on observations performed on KTH [153] and UCF Sports [164] datasets. Each dataset was split into 70% and 30% for training and validation sets, respectively. Increasing of this number of units neither decreased the loss nor increased the accuracy, over the epochs.

4.2.1 Datasets

Regarding the evaluation of the method on RGB data, the following four datasets were used:

- **KTH** [153], containing 6 types of challenging human actions (i.e., walking, jogging, running, boxing, hand waving, and hand clapping) performed several times by 25 actors in indoor and outdoor scenarios. In addition to the comparison with the current literature, this benchmark was also used to test the robustness of the method in presence of partial body occlusions. Besides, KTH was used to study different RNN architectures and to choose the best configuration for the proposed pipeline.

- **Weizmann** [47], containing 90 low-resolution testing video sequences that show 9 different persons performing 10 natural actions (i.e., run, walk, skip, jumping-jack, jump-forward-on-two-legs, jump-in-place-on-two-legs, gallopsideways, wave-two-hands, wave-one-hand, and bend). In addition to the comparison with other key works of the current state-of-the-art, this benchmark was used to evaluate the method performance when very few video sequences are used in the training stage.

- **UCF Sports** [164], containing 150 sequences, at a resolution of 720 × 480, that show different actions collected from various sports (e.g, diving, golf swing, kicking, lifting, riding horse, running, skateboarding, swing-bench, swing-side, and walking). This dataset was mainly used to prove the robustness of the method in presence of both partial body occlusions (of the lower part) and noisy data caused by incorrect 2D skeleton extraction.

- **IXMAS** [193], composed of 13 daily-live motions, each performed 3 times by 11 actors. In the dataset, the actors assumed a wide range of positions and orientations. This dataset was mainly used to evaluate the method under different camera angles and positions.

- **HMDB51** [85], composed of 7000 clips distributed in 51 action classes. In the dataset, the actors perform general facial and body actions, including actions with object manipulation. This dataset was mainly used to evaluate

[3] https://www.tensorflow.org/

Layer	# Units
LSTM	256
LSTM	128
LSTM	64
LSTM	20 (# features)

(a)

Layer	# Units
LSTM	256
LSTM	128
LSTM	128
Dense (LeakyRelu [104])	512
Dense (LeakyRelu, Dropout=20%)	256
Dense (sigmoid)	1

(b)

Table 4.4. Layers and units used in the proposed implementation for the: (a) Generator and (b) Discriminator. In the dense layer, the type of activation function and the percentage of dropout (if used) are indicated.

the performance of the method, integrating the 3D-DenseNet, on large RGB datasets.

- **UCF101** [165], composed of 13320 clips, collected from YouTube, divided in 101 action classes. This dataset presents a large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background and illumination conditions. This dataset was mainly used to evaluate the performance of the method, integrating the 3D-DenseNet, on large RGB datasets.

- **Kinetics400** [74], composed of 400 human action classes, with at least 400 video clips for each class. The clips last around 10s and were collected from YouTube. The actions cover a broad range of classes including human-object interactions, such as playing instruments, as well as human-human interactions like shaking hands.

To show that the accuracy of the proposed method is fully comparable to that of works based on 3D skeletons, additional comparisons on UT-Kinect [196] and NTU-RGB+D [155] datasets were also performed:

- **UT-Kinect** consists of data acquired with a single stationary Kinect sensor. It comprises 10 action classes (i.e., walk, sit down, stand up, pick up, carry, throw, push, pull, wave hands, and clap hands) performed by 10 actors, where each actor performs every action twice.

- **NTU-RGB+D** consists of 56880 action samples, divided into 60 different classes, containing, for each sample, RGB videos, depth map sequences, 3D skeletal data, and infrared videos. This dataset was acquired by 3 Kinect seonsors concurrently.

For the tests on the two just reported datasets, our method only used the RGB information to generate the 2D skeletons. Finally, to fully exploit the proposed architecture and features, on the KTH, UCF Sports, and UT-Kinect datasets, which present few samples, results both on the original sequences and on the latter enriched by C-RNN-GANs [113] are presented.

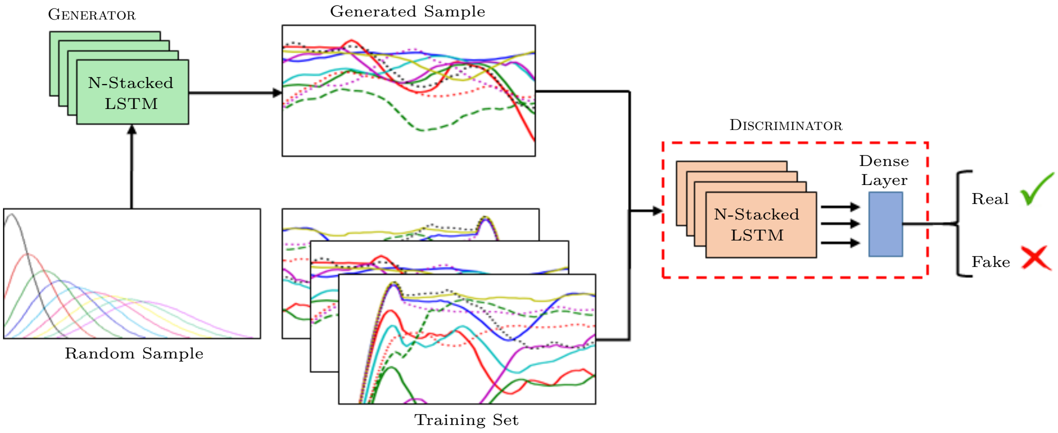


Figure 4.5. Architecture of the C-RNN-GAN. The Generator produces sequences of continuous data events starting from random samples derived by a Gaussian distribution. The Discriminator is trained to distinguish between real action sequence data and generated data.

Architecture	Accuracy
2-Branch, 3-Stacked LSTMs	**98.33%**
2-Branch, 3-Stacked Phased LSTMs	95.00%
3-Stacked LSTMs	96.67%
3-Stacked Phased LSTMs	93.14%

(a)

Architecture	Accuracy
2-Branch, 3-Stacked LSTMs	**100%**
2-Branch, 3-Stacked Phased LSTMs	86.67%

(b)

Table 4.5. Comparison among different LSTM and phased LSTM architectures on: (a) KTH and (b) Weizmann datasets.

4.2.2 Data Augmentation by C-RNN-GANs

Usually, deep neural networks need a large amount of data to be trained. Despite this, some challenging datasets, as KTH, UCF Sports, and UT-Kinect, do not contain a sufficient number of video sequences. This situation can lead to the overfitting problem [166], which can be addressed by dropout techniques. In this work, in addition to a classic dropout solution, a C-RNN-GANs is used to perform data augmentation on the three datasets reported above. C-RNN-GANs are able to learn how to produce synthetic data sequences that are indistinguishable from real data. This goal is obtained through the adversarial work performed by two networks: *Generator* and *Discriminator* (Fig. 4.5). The *Generator* takes as input a random sample from a latent space (represented, in our case, by random Gaussian distributions) and generates a real-looking skeleton feature sequences of a specific action class. Instead, the *Discriminator* is trained to tell apart sequences from the used ground truth dataset with the fake ones given by the *Generator*. In Table 4.4 details about layers and units that compose our implementation of *Generator* (Table 4.4a) and *Discriminator* (Table 4.4b) are shown. Notice that, the ground truth set of sequences used in the *Discriminator* training does not contain those sequences used later in the test phase of our method, because their presence could both influence the creation of synthetic sequences and increase the risk of an

KTH Class	Two-Branch Stacked LSTM	Single Branch (Upper or Lower)
Walking	94.44%	100%
Jogging	94.44%	97.22%
Running	97.22%	100%
Hand Clapping	97.22%	100%
Hand Waving	97.22%	100%
Boxing	100%	100%

Table 4.6. Results on the modified video sequences of the KTH dataset on the basis of the accuracy metric by using the whole two-branch stacked LSTM network, analysing the entire skeleton, and the single upper or lower branch (as reported in Sec. 3.3.4), focusing on the not occluded parts of the body.

overlapping between training and testing sets. Following this consideration, for each dataset reported above, we generated new data according to their experimentation protocols.

In the KTH dataset, the training set composed of sequences of 16 random subjects, as described in [126], was used to generate 10 new skeletal data for each action. To reduce the dependency of the random selection, we produced multiple different random splits and, afterwards, we repeated the data augmentation process for each split. Lastly, in the test phase, the final result was obtained as the average of all runs performed on all splits. In the UCF Sports dataset, for each training set derived from the use of the leave-one-out cross validation protocol, reported in [86], we doubled the data contained for each class. Only for the lifting class (the class with less data) the amount of sequences was quadrupled, thus obtaining 20 videos for each training set in the dataset. About the UT-Kinect dataset, the C-RNN-GAN was used to synthesized 15 extra sequences for each class, where each training set was established by the protocol discussed in [180]. The data augmentation process was not applied on IXMAS, NTU-RGB+D, and Weizmann datasets, since the first two datasets were composed of a large number of sequences, while the third was specifically used to analyse the method performance in extreme situations with few data.

4.2.3 LSTM Versus Phased LSTM

In a first set of tests, we compared our solution with similar models based on the recently proposed phased LSTM units [116], which allow to considerably reduce the training time of a network. In particular, we compared the following four reference architectures: 3-stacked phased LSTMs, 3-stacked LSTMs, two-branch with 3-stacked phased LSTMs for each branch, and two-branch with 3-stacked LSTMs for each branch. The tests were performed on the KTH dataset (without augmented data) using the standard partition and following the protocol specified in [153]. Although all architectures achieved remarkable results, Tab. 4.5a shows that models based on LSTMs perform better than those based on phased LSTMs. This is due to the fact that phased LSTM units require of an extensive amount of training data to achieve good performance. Such results were also confirmed by additional

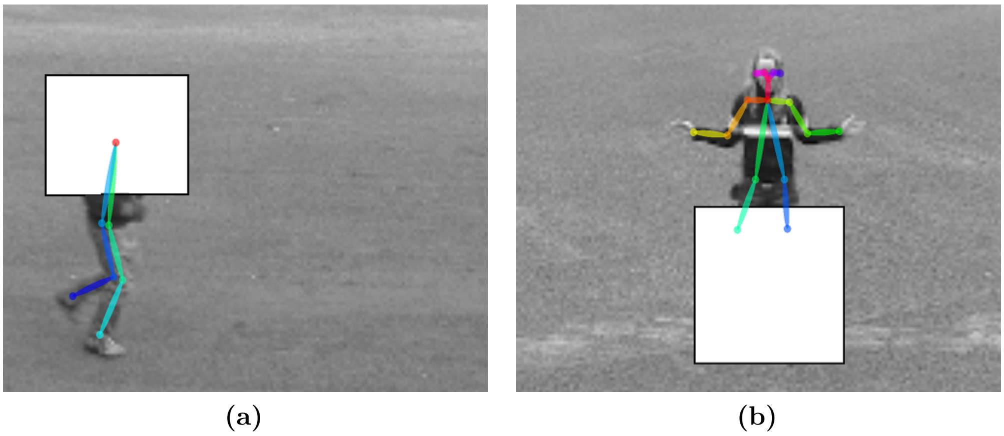


Figure 4.6. Examples of skeleton joints extracted from the modified video sequences of the KTH dataset to test the method under partial occlusions of: (a) upper and (b) lower body parts.

tests performed on the Weizmann dataset (see Tab. 4.5b), a dataset that contains less data than KTH, thus exhibiting a decrease in performance of the phased LSTM based model. The results reported above have motivated our choice in using classical LSTM units for the proposed architecture, thus providing a method that can work well even in situations with limited amount of data. Following empirical tests, in our pipeline the number of stacked LSTMs was set to 3. The increase of this number, at the expense of an increase in training time, did not lead to a significantly better accuracy.

4.2.4 Partial Occlusion Tests

The architecture of the proposed method was designed to estimate a human action even in presence of partially covered subjects. To show the effectiveness of the method, also in these cases, several tests were performed. Since public datasets in the action recognition state-of-the-art only present subjects with small occlusions, we created a customized collection of video sequences on the basis of the KTH dataset. The modification of the video frames was achieved by using the following simple steps:

- The tracking algorithm proposed in [204] was used to obtain a bounding box $\beta_{i,t}$ for each actor i inside each frame at the time t for all the video sequences collected in the KTH dataset;

- Each $\beta_{i,t}$ was divided into two parts: upper and lower;

- Finally, for each video sequence, if the action was focused on the lower body part (i.e., running, jogging, or walking) a square patch was applied to occlude the upper bounding box part. Conversely, if the action was focused on the upper body part (i.e., hand clapping, hand waving, or boxing) a square patch was applied to occlude the lower bounding box part.

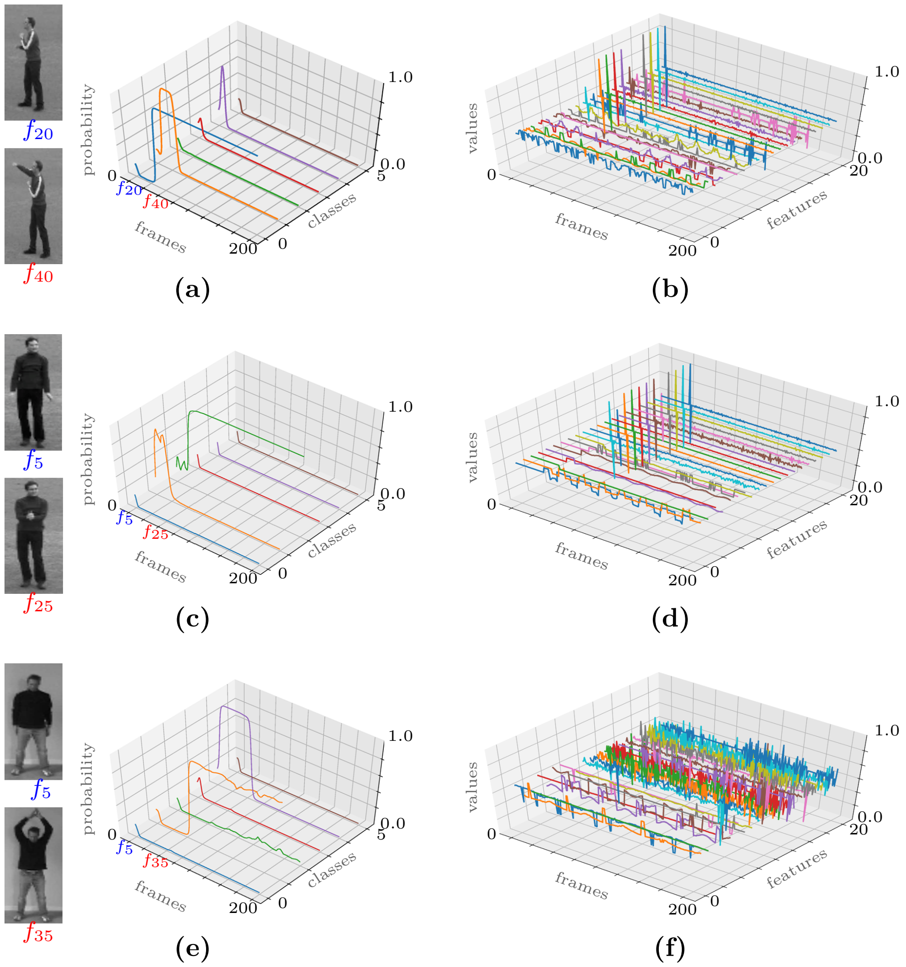


Figure 4.7. Examples of: (a)(c)(e) classification and (b)(d)(f) feature values for boxing, hand clapping, and hand waving. In all the plots, x-axis is the time. In the classification plots (left column), y-axis and z-axis are class probability and class indices, respectively. In the feature analysis plots (right column), y-axis and z-axis are feature values and feature indices, respectively.

In Fig. 4.6, examples of the just reported approach are shown. Notice that, on the KTH database, we considered only occlusions of the lower and upper body parts for the following reasons: a) a such modified dataset can be easily obtained; b) the UCF Sports dataset, used in our experimentation, already includes subjects with partial occlusions of individual limbs. In this case, we were interested in analysing larger occlusions.

Following the protocol established in [126], in Tab. 4.6 the evaluation of the method on the modified KTH dataset is shown. The results highlight that the use of only a specific branch to recognize a targeted subset of actions can bring a greater accuracy in presence of occlusions. In fact, the remaining not considered classes would be impossible to recognize without the view of the hidden portion of the body.

Parisi et al. [126]	94.91%
Ji et al. [67]	90.20%
Wang et al. [184]	94.20%
Gao et al. [43]	95.04%
Proposed Method	98.33%
Proposed Method (with a.d.)	99.50%

(a)

Parisi et al. [126]	100%
Bregonzio et al. [16]	96.66%
Thurau et al. [173]	94.40%
Proposed Method	100%

(b)

Wang et al. [185]	85.60%
Le et al. [90]	86.50%
Wang et al. [184]	89.10%
Weinzaepfel et al. [194]	90.50%
Abdulmunem et al. [2]	90.90%
Rahmani et al. [138]	93.80%
Hou et al. [61]	95.70%
Proposed Method	86.60%
Proposed Method (with a.d.)	88.20%
Proposed Method (with 3D-DenseNet)	99.60%

(c)

Table 4.7. Comparisons with RGB based methods on the: (a) KTH, (b) Weizmann, and (c) UCF Sports datasets (if present, the a.d. term highlights that the results were obtained by also including augmented data). In addition, in (c), the results of the proposed network combined with the 3D-DenseNet are reported.

4.2.5 Feature Analysis

A key aspect of the skeleton features used in the proposed method is their ability in acquiring detailed information about a recognized action, going beyond the simple classification task. This gives us the opportunity to have itemized data of the analysed human movements to have a deeper understanding of the whole action. In Fig. 4.7 an example is reported. In particular, in Fig. 4.7a, the subject starts to throw punches within the frame f_t when $t = 40$. Similarly, in Fig. 4.7c, the subject starts clapping the hands at time $t = 25$. In all-class probability plot (i.e., Fig. 4.7a, Fig. 4.7c, and Fig. 4.7e), class indices are defined as follows: 0) Boxing, 1) Hand Clapping, 2) Hand Waving, 3) Jogging, 4) Running, and 5) Walking. Notice that, besides, in Fig. 4.7a, the prediction changes over time, in fact before the arm is completely extended, the feet/arms motion may suggest the beginning of a running action. Likewise, as shown in Fig. 4.7b, by analysing the variations of the joint acceleration and angle, how often a subject moves the fists can be computed. Finally, it is also interesting to observe the correlation between classification and feature plots. In Fig. 4.7d and Fig. 4.7f, for example, we can see how the variations in arm acceleration and angle can define periodic oscillations that encode the motion pattern (i.e., rhythmic movement). Instead, in Fig. 4.7b and Fig. 4.7d, how feature peaks at small time values can identify the transition from an idle state to a sudden

Layers	Output size	Parameters
Convolution	$56 \times 56 \times 72$	$7 \times 7 \times 7$ conv, stride 2
Pooling	$28 \times 28 \times 36$	$3 \times 3 \times 3$ max pool, stride 2
3D Dense Block (1)	$28 \times 28 \times 36$	$\begin{bmatrix} 1 \times 1 \times 1 \text{ conv} \\ 3 \times 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	$14 \times 14 \times 8$	$1 \times 1 \times 1$ conv
	$14 \times 14 \times 8$	$1 \times 1 \times 1$ average pool, stride2
3D-Dense Block (2)	$14 \times 14 \times 18$	$\begin{bmatrix} 1 \times 1 \times 1 \text{ conv} \\ 3 \times 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	$14 \times 14 \times 8$	$1 \times 1 \times 1$ conv
	$7 \times 7 \times 9$	$1 \times 1 \times 1$ max pool, stride2
3D-Dense Block (3)	$7 \times 7 \times 9$	$\begin{bmatrix} 1 \times 1 \times 1 \text{ conv} \\ 3 \times 3 \times 3 \text{ conv} \end{bmatrix} \times 32$
Transition Layer (3)	$7 \times 7 \times 9$	$1 \times 1 \times 1$ conv
	$3 \times 3 \times 4$	$1 \times 1 \times 1$ max pool, stride2
Pooling	$1 \times 1 \times 1$	$3 \times 3 \times 4$ global max pool

Table 4.8. Layers and parameters used in the 3D-DenseNet model. Each convolutional layer inside the dense block and the transition layer consists of batch normalization, ReLU, and convolution.

motion. All the kinds of information reported above can be used in more complex applications (e.g., video surveillance, event recognition) to interpret very complex interactions among subjects.

4.2.6 Quantitative Comparisons

The proposed method has obtained remarkable results with respect to the current literature in human action recognition. Regarding RGB based approaches, extensive comparisons on KTH, Weizmann, UCF Sports, HDBM51, UCF101, IXMAS, and NTU-RGB+D datasets were performed. For the KTH benchmark, a standard protocol described in [126], where 16 randomly selected subjects were used for training and other 9 subjects for testing, was followed. As reported in Section 4.2.2,

Method	HDBM51	UCF101
Graph distillation [102]	N.A.	80.30%
Zhang et al. [202]	55.3%	87.5%
ML-HDP+iDT+sTDD [176]	N.A.	89.3%
Opt Flow+RGB+iDT [179]	67.2%	92.7%
Li et al. [94]	N.A.	93.6%
Hou et al. [61]	N.A.	94.4%
Tu et al. [177]	73.3%	96.3%
Tu et al. [177] (Pretrained on Kinetics)	80.9%	97.9%
Our Method (with 3D-DenseNet)	72.1%	96.2%

(a)

Method	Kinetics400
RGB-I3D [22]	72.9%
FLOW-I3D [22]	63.5%
Two Stream ConvNet [159]	61%
Two Stream ARTNet [187]	71.3%
Two Stream I3D [22]	74.9%
TSN (2D ConvNet) [189]	73.9%
TSN (I3D) [189]	75.7%
Our Method (with 3D-DenseNet)	75.1 %

(b)

Table 4.9. Comparisons with the state-of-the-art methods combining the proposed architecture and 3D-DenseNet on (a) HDBM51, (a) UCF101 datasets and (b) Kinetics400.

Camera 00	80.50%
Camera 01	76.35%
Camera 02	73.20%
Camera 03	78.00%
All Camera	92.60%

(a)

Histograms of 3D Joints [196]	90.90%
Spatio-Temporal Local Features [209]	87.90%
Lie Group [180]	97.10%
Hankel-HMM [100]	86.80%
Manifold PCA [3]	94.90%
Graph-Joint-LSTM [99]	97.00%
Proposed Method	94.60%
Proposed Method (with a.d.)	95.70%

(b)

Table 4.10. (a) Results on the IXMAS dataset. (b) Comparisons with RGB-D based methods on the UT-Kinect dataset (if present, the a.d. term highlights that the results were obtained by also including augmented data).

HON4D [125]	30.56%
Super Normal Vector [198]	31.82%
HOG2 [121]	32.24%
Skeletal Quads [40]	41.36%
Shuffle and Learn [111]	47.50%
Key Poses + SVM [27]	48.90%
Lie Group [180]	50.08%
FTP Dynamic Skeletons [62]	60.23%
P-LSTM [155]	62.93%
ST-LSTM [99]	69.20%
DSSCA-SSLM [156]	74.86%
Joint Distance Maps + CNN [93]	76.20%
ResNet-56 [128]	78.20%
LSTM+CNN [25]	82.89%
Multitask Deep Learning [103]	85.50%
DGNN [157]	89.90%
Proposed Method	69.53%

(a)

HON4D [125]	7.26%
Super Normal Vector [198]	13.61%
HOG2 [121]	22.27%
Skeletal Quads [40]	41.36%
Lie Group [180]	52.76%
Key Poses + SVM [27]	57.70%
FTP Dynamic Skeletons [62]	65.22%
P-LSTM [155]	70.27%
ST-LSTM [99]	77.70%
Joint Distance Maps + CNN [93]	82.30%
ResNet-56 [128]	85.60%
LSTM+CNN [25]	90.10%
DGNN [157]	96.10%
Proposed Method	78.6%

(b)

Table 4.11. Comparisons with the state-of-the-art methods on the NTU-RGB+D dataset by using: (a) cross-subject evaluation criteria and (b) cross-view evaluation criteria.

multiple splits were generated and the final result was obtained by the average of all runs on all splits. As shown in Tab. 4.7a, in this benchmark, we outperform the two best results previously obtained in [126] and [43]. As introduced in the above sections, the proposed method is also able to obtain excellent results in situations with limited amount of data, as shown on the Weizmann dataset (see Tab. 4.7b), where our results are fully comparable with key works of the present state-of-the-art [126]. Also in these evaluations a protocol, named leave-one-out cross-validation, described in [126], was followed. The UCF Sports dataset (Tab. 4.7c) can be considered very challenging due to the presence of occlusions, multiple subjects, and complex actions. Particularly critical is the lifting class, where skeletons cannot be correctly extracted. Anyway, excluding this class from the performance computation, the method reached an accuracy of the 91%. For the evaluation on this dataset, the protocol reported in [86], where each sample is used in turn as a test video, was followed. Unlike the previous tests, in these estimations a dropout of 30% on the last dense level nodes of the network was applied to reduce the overfitting problem. As reported in Tab. 4.7a and Tab. 4.7c, the use of C-RNN-GANs for the data augmentation phase can provide a concrete expedient, in term of accuracy, especially for those datasets whose data is not entirely sufficient for deep learning applications. In Tab.4.7c is still possible to notice how the 3D-DenseNet significantly improves the classification accuracy of the proposed method, handling missing skeletal data and integrating scene information. The parameters used in the construction of the 3D-DenseNet model, shown in Tab.4.8, are obtained through empirical experiments carried out on the UCF101 dataset; using 70% of the video sequences as training set and the remaining 30% as validation set. For each video, $n = 20$ frames were sampled, by dividing the sequence into n slots and taking the first frame for each slot. The combination between the proposed approach and 3D-DenseNet was further tested on a large amount of data and number of classes, by using the HDBM51, UCF101 and Kinetics400 datasets. As can be seen from Tab.4.9a, following the same experimental protocols defined in [177], the accuracy of our method is comparable with the current state-of-the-art best results on the HDBM51 and UCF101 datasets. However, the obtained accuracy is far from the one reported in [177], where the network uses a pre-training procedure by using the Kinetics400 dataset. Instead, we avoided using of pre-training in order not to influence the validity of results in any way. This is because the Kinetics400 is a huge dataset and many of the sequences contained in it are similar to the ones of the HDBM51 and UCF101 datasets. For each dataset used, we chose to use only the training data provided by standard evaluation protocols, or the augmented data obtained from the latter. For the Kinetics400, to perform further challenging experiments, the 1-top protocol specified in [74], was used and the obtained result is reported in Tab.4.9b. Also, in this case, the method is comparable with the current state of the art in terms of accuracy. In the HDBM51, UCF101 and Kinetics400 datasets, the large amount of videos with both missing skeletal data and face close-ups, makes impossible performing tests using the architecture composed only by the two-branch stacked LSTMs.

Moreover, a supplementary challenge is introduced by the IXMAS dataset, where video sequences are acquired from five cameras at different angles and perspectives with respect to the subjects that perform actions. Evaluations are performed by using the leave-one-camera-out cross-validation, where all sequences coming from a camera are used to perform tests and remaining sequences are used for the training. Due to the impossibility to obtain a reliable skeleton from the sequences acquired from a nadir camera, in the training step we did not consider the streams captured by the camera with $id = 04$. In Tab.4.10a, the results, on each single camera view, are shown. The result of the last row was obtained by using the leave-one-subject-out cross-validation on all video sequences of all four cameras.

A final set of qualitative comparisons was performed with RGB-D based methods, which are designed to make full use of depth information. We point out that our pipeline is entirely based on RGB information, thus making our approach applicable in uncontrolled real-world scenarios where depth sensors can present drawbacks and limitations. Similarly to the UCF Sports dataset, for these tests a dropout of 30% on the dense layer was applied. Comparisons on the UT-Kinect dataset (Tab. 4.10b) was, once again, evaluated by the leave-one-subject-out cross-validation, according to the protocol reported in [180]. While, comparisons on the NTU-RGB+D dataset were performed using both cross-subject (Tab. 4.11a) and cross-view (Tab. 4.11b) evaluation criteria, as described in [155]. The values reported in Tab. 4.10b and Tab. 4.11 show that the accuracy of the proposed method is fully comparable also with methods that use depth information, thus highlighting again the versatility and robustness of the proposed method. Among all the methods shown in Tab. 4.11, The DGNN [157] gets the best result. This type of network, which manages skeleton information like a directed acyclic graph (DAG), could also be used on 2D skeleton information, becoming our direct competitor. However, this method does not provide solutions to manage occlusions (i.e., missing skeleton joints), multiple people in the scene and face close-ups; in this way, unlike the solution we propose, it may not be suitable for managing real applications, such as active video surveillance. As for the previous experiments, Tab. 4.10b underlines how the use of synthetic data, for the training step, can significantly improve the classification results.

4.2.7 Two-Branch Stacked LSTM versus 3D-DenseNet

In this section a comparison between the two-branch stacked and the 3D-DenseNet networks is presented to show how, in some cases, skeletal analysis can offer advantages over the use of 3D-CNN. The experiments were performed on a custom dataset, created by merging all the KTH video sequences and a UCF101 class subset, the latter consisting of 3394 video sequences and 26 classes. As a result, the whole custom dataset consists of 5785 video sequences divided in 32 classes. These videos are chosen because, for each person in the scene, it is always possible to acquire the entire skeleton with few noise. The experiments, performed using 70% of samples as training and the remaining data as test, indicate that the two-branch stacked LSTM, obtaining 95.12% accuracy, exceeds the 92.46% accuracy obtained from the 3D-DenseNet. The 3D-DenseNet performs classification errors especially on classes taken from the KTH dataset, all of which have a homogeneous background and the people in the video sequences do not interact with any object in the scene. In conclusion, in case of correct skeletal data acquisition, the architecture based on LSTM and joint information behaves better than a CNN-based architecture focused solely on RGB images.

4.3 Non-acted body affect recognition experiments

In this section, implementation details and exhaustive experiments on several configurations of the proposed non-acted body affect recognition module are reported. We also compare our method with key works of the non-acted affective recognition literature.

4.3.1 Dataset

The benchmark used in the experiments is the UCLIC Affective Body Posture and Motion database [81]. This collection is composed by 37 sequences capturing 11 standing humans playing Nintendo Wii Sports games for a minimum of thirty minutes. The chosen format is the BioVision Hierarchy (BVH), which provides both a skeleton hierarchy and its motion data. The hierarchy comprises all relevant joints for the skeleton, including head, neck, chest, collar, shoulders, elbows, wrists, hips, knees, and ankles. About the motion data, for each frame of the recording, the starting X, Y, and Z positions of the hierarchy root are stored, as well as the various joints rotations. Starting from these values, the 3D location of all body joints are computed. Kleinsmith et al. [81] analysed the whole dataset and hand-picked 103 frames across all sequences, representing skeleton configurations in which emotional expressiveness can be identified, thus defining key postures inside the available recordings. The authors, following an established protocol, labelled the frames of the UCLIC dataset according to human observations. The affects identified inside the sequences, observable while participants play the games, were classified as: concentration, triumph, frustration, and defeat. Finally, human observations were used to define a base rate for human capabilities, thus estimating the classification accuracy for each of these affect classes. In this work, in order to associate movements evolving through time to single body postures, we retained sequences of poses starting from the selected key postures. In more details, 103 sequences were retrieved using the preceding and subsequent 50 frames of a recording, with respect to a given key posture. To each sequence was then associated the same label as the analysed key posture, effectively associating movement to the right affect.

4.3.2 Implementation details

All the experiments were performed on an 6-Core Intel i7 2.60GHz CPU with 32GB RAM. The proposed network was implemented using the TensorFlow [1] framework. The following assessments were performed using the 10-cross validation average, according the protocol used in [81].

Both LSTM and MLP branches use 64 hidden units in each layer, where the LSTM consists in 3-stacked layers and the MLP is composed of 3 hidden dense layers. To avoid the over-fitting problem, a 30% dropout [166] probability is applied in each layer of both branches. In the stacked LSTM the dropout is the recurrent one. The training step was performed using 1500 epochs, with a learning rate of 0.01 and a batch size of 5. The choice of the LSTM units is based on the results obtained by comparing the most common recurrent units used in the state-of-the-art, as shown in Tab. 4.12. This procedure was performed because, as reported in several works (e.g., [70, 26]), the performance of gated recurrent units may depend heavily on both dataset and corresponding task. So a trial test phase was required to choose the right cell. Although the more recent units (i.e., GRU [70], UGRNN [28], and NAS [11]) have obtained higher scores on single affective categories, such as concentrated and defeated, the LSTM was chosen since able to achieve the best average percentage overall. Finally, this hardware and network configuration, given an input sequence, is able to work in real-time once the system is trained.

4.3.3 Model and feature analysis

A key aspect of the proposed method was feature selection. The description of several feature groups, tested out to find the most effective combination, is reported below:

M0: this set is represented by a vector of 277 entries, which includes all the temporal

Unit	C	T	F	D
LSTM	70.00%	**62.86%**	**20.00%**	60.00%
RNN	64.45%	48.57%	20.00%	**72.00%**
GRU	76.67%	31.43%	10.00%	40.00%
UGRNN	**77.78%**	51.43%	0.00%	52.00%
NAS	72.22%	52.42%	0.00%	64.00%

Table 4.12. Experimental results for each UCLIC dataset class, using the 10-cross validation protocol, obtained by varying recurrent neural network units; where C = concentrating, T = triumphant, F = frustrated, and D = defeated.

global features; In more details, 61 entries are related to the introduced meta-features, while the remaining 216 entries represent the average X, Y, and Z values of classic features (i.e., velocity, acceleration, snap, and jerk), computed over the 18 available joints;

M1: this set is composed of 331 features vectors of length 101. The size of each vector is defined by the 50 preceding and subsequent frames, with respect to the analysed key posture. This window size was chosen to ease the computation of the model, as well as to avoid representing movements that happen too far in time from a selected posture. In more details, this feature set is composed of three vectors groups of sizes 61, 216, and 54, representing, respectively, meta-features, classic features, and 3D joints positions;

R0: this set is represented by the raw Euler joint rotation angles of all available joints, stored in the UCLIC dataset. This set of rotations uses the same time window of M1 (i.e., 101 rotation frames);

R1: this set represents the R0 features normalized in $[0, 1]$ taking into account the joint movement range. This set of rotations uses the same time window of M1.

The obtained results, shown in Tab. 4.13, highlight how the simultaneous use of the two branches is more effective with respect to the single branches. This behaviour is easily explained since by using both branches (i.e., employing both temporal local and global features), the model can use both a summary and detailed information of a motion to describe an emotion. Moreover, tests emphasize a better classification of the concentrated, frustrated, and defeated affects whenever the normalized rotations are considered inside the temporal local features on both single LSTM branch and full architecture. This outcome is due to the fact that, thanks to smoother values (i.e., $[0, 1]$ normalized rotations), the LSTM can better handle body posture motions in the time evolution scenario we devised. Another important aspect, again backed-up by Tab. 4.13, is the relevance of time-based information. It should be noted that all the LSTM based models exceed the MLP performance, which does not consider a detailed motion information but only its overview. A rationale behind this behaviour can be attributed, for example, to speed variations on the human joint configuration depending on the current affective state. Indeed, as shown in Fig. 4.8, a comparison between triumphant and concentrated affects can

Model	Features	C	T	F	D
B2	M0	67.78%	62.86%	15.00%	62.00%
B1	M1	72,23%	62.86%	20.00%	62.00%
B1	R0	70.00%	62.86%	20.00%	60.00%
B1	R1	72,23%	65.71%	20.00%	64.00%
B1	R0+M1	75.56%	65.71%	30.00%	64.00%
B1	R1+M1	76.67%	65.71%	20.00%	64.00%
B1 + B2	R0+M1, M0	77.78%	68.57%	30.00%	64.00%
B1 + B2	R1+M1, M0	**78.89%**	**68.57%**	**35.00%**	**68.00%**

Table 4.13. Experimental results for each UCLIC dataset class, obtained by using the first branch B1 (based on the stacked LSTM), the second branch B2 (based on the MLP), and the whole network, varying the features groups.

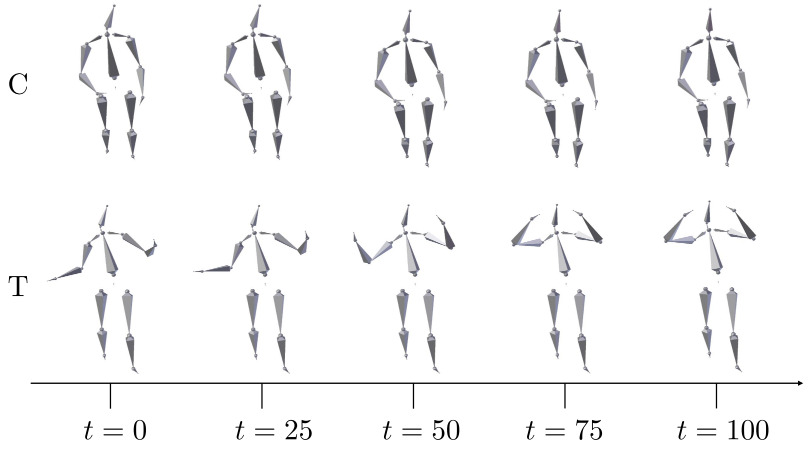
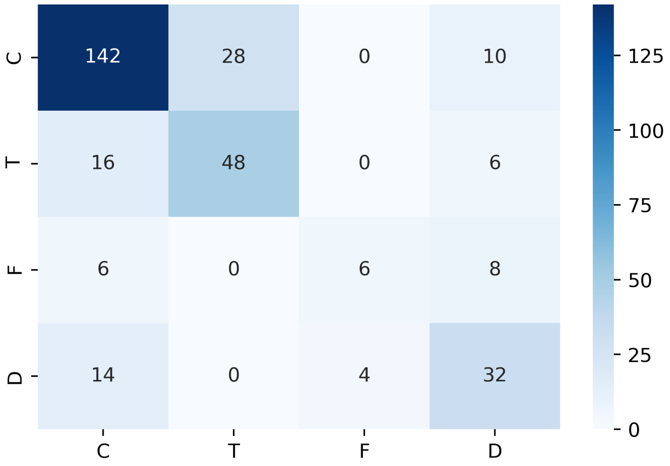


Figure 4.8. Body motion comparison between concentrated and triumphant affects on the 101 analysed time window.

be considered, where the former causes the human subject to perform more dynamic movements, while the latter leads to a more static motion sequence. Finally, an overview of the system classification performance, over the UCLIC dataset, is shown in Fig. 4.9. As expected, most of the results lie on the confusion matrix diagonal. The misclassified samples are usually associated with the concentrated class, due to human observers using this affect as the neutral one in the referenced dataset. During the training step, the network learns this pattern and, following the human behaviour, uses the concentrated category as the default one. Furthermore, the frustrated affect has the highest misclassification rates since, in the UCLIC dataset, only 5 samples for this emotion are available and the identified key postures differ from one another, as shown in Fig. 4.10. Thus resulting in a really hard affect to recognize.

To conclude the model analysis, in Fig. 4.11 are shown three sampled sequences, where the top and central row represent the frustrated emotion, while the bottom row shows the concentrated affect. The proposed model is able to correctly classify the first and third sequences while, for the second one, there is a misclassification



]

Figure 4.9. Confusion matrix of the proposed solution according to the affect set on the UCLIC dataset over a 10-cross validation run. The rows and columns of the confusion matrix denote predictions and ground truth, respectively.

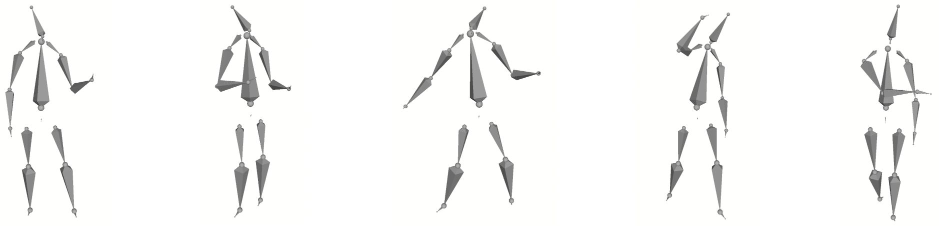


Figure 4.10. UCLIC dataset frustrated affect key postures.

indicating the emotion as concentrated. The correct classification for the first sample is feasible since the model is able to grasp the sudden movement (e.g., via the smooth-jerk feature) and low symmetry of the body, which can be often seen in a frustrated person. Regarding the correct classification of the third sample, the model can correctly represent the low body movement and its high symmetry, which can usually be observed in a person focusing on a given task. Finally, concerning the second misclassified sample, the error is caused since first, there are few examples for this affect, and second, the sequence shows various characteristics of the concentrated emotion (e.g., high legs and arms symmetry, low values for features analysing motions, etc.). This second sequence can be easily misclassified, even where there are cues like the backward tilting head and the upward symmetry of the hands.

4.3.4 Data augmentation

A common issue that afflicts numerous datasets is the low number of labelled samples. In particular, the UCLIC dataset suffers this aspect, as observable by the class related to the frustrated affect, thus making the correct classification by automated

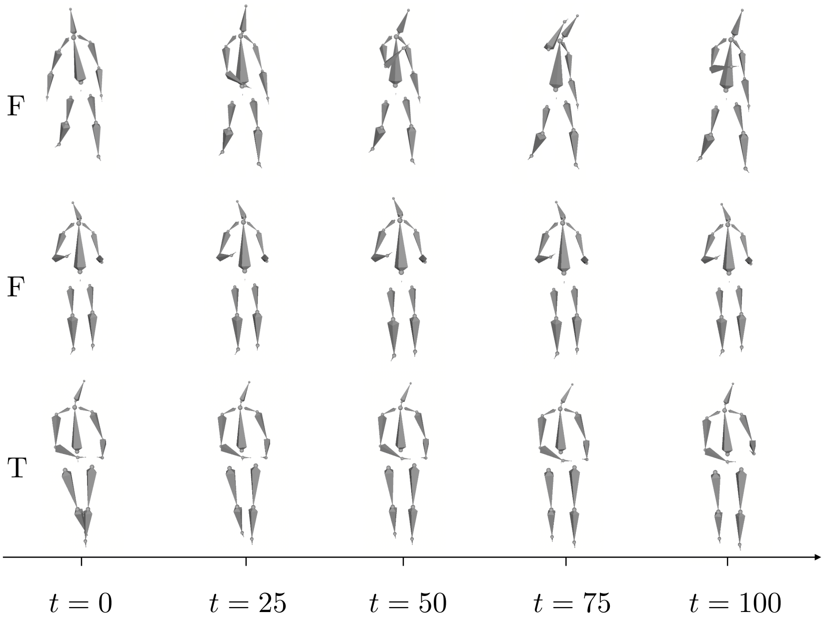


Figure 4.11. UCLIC dataset affect examples. Top and center rows show the frustrated emotion, while bottom row represents the concentrated affect.

systems a hard task. To overcome this problem, an automatic data augmentation procedure was devised with the aim to increase the number of samples and to obtain a uniform distribution of data among the various emotion categories. The conceived procedure was designed as follows: starting from the original 3D skeleton joint rotation sequences of the UCLIC dataset, noise was applied to frames near the labelled key postures, following a uniform Gaussian distribution with low standard deviation (i.e., ±5° per axis angle), thus generating new joint rotation sequences. This new motion data was then normalized to avoid meaningless rotations, which are, normally, either unfeasible or extremely rare (e.g., an head turned backwards, or an hyper-extended elbow) to be used inside the dataset. With this simple approach the procedure was able to generate synthetic data sequences that are consistent with the real data, thus achieving a uniform distribution of samples among the four emotion categories and reaching 250 samples instead of being limited to only 103. To avoid an over-fitting scenario and wrong recognition rates due to similar samples being wrongly distributed in both training and test sets, the synthetic sequences were placed in the same set of the original sample they were generated from. If an original sample is contained in the test set, during the 10-cross validation approach, its associated forged sequences are not considered for either the training nor the test sets. In Tab. 4.14 the results using the augmented data are reported. Models trained on the synthetic sequences easily outperform their counterparts trained on the original dataset. These improvements also affect the more challenging frustrated set, clearly demonstrating that having more real samples would improve the classification rates, especially with our proposed network. Indeed, this augmented model, can also correctly classify the second sequence of Fig. 4.11, since, thanks to having more samples in the training set, the model is able to grasp other cues of the motion such as the upward hands symmetry and backward tilting head.

Type of Dataset	Features	C	T	F	D
Base	R0	70.00%	62.86%	20.00%	60.00%
	R1	72,23%	65.71%	20.00%	64.00%
	R0+M1, M0	77.78%	68.57%	30.00%	64.00%
	R1+M1, M0	78.89%	68.57%	35.00%	68.00%
Augmented	R0	73.28%	67.59%	10.00%	61.00%
	R1	74.95%	67.82%	20.00%	61.25%
	R0+M1, M0	76.67%	79.05%	50.84%	67.00%
	R1+M1, M0	82.22%	80.95%	51.67%	70.00%

Table 4.14. Experimental result comparison, for each UCLIC dataset class, among the various proposed network models, with or without the augmented data, using the 10-cross validation protocol.

system	C	T	F	D
human base rate	57.00%	64.00%	39.00%	61.00%
Kleinsmith et al. [81]	65.30%	61.90%	16.00%	64.70%
our	**78.89%**	**68.57%**	**35.00%**	**68.00%**

Table 4.15. Experimental comparisons on the UCLIC dataset using the 10-cross validation protocol.

4.3.5 Comparisons

The proposed method is compared to the current literature in non-acted body affect recognition. The results on the UCLIC dataset, shown in Tab. 4.15, highlight how our method exceeds the pioneer work of Kleinsmith et al. [81] in the classification of each class. Especially in the recognition of the frustrated affect, that represents one of the major challenges offered by this dataset, the accuracy is greatly increased. Our system is able to improve performances thanks to the devised multi-branch architecture, which can fully leverage both local and global temporal features. Indeed, the model can better associate body movements to affects due to its fine-grained (i.e., local via LSTM) and coarse (i.e., global via MLP) description of the motion. Moreover, our solution is in line with respect to the reported human base rate.

In Tab. 4.16 the overall results based on the average of the accuracy rates, obtained on each class, are shown. One of the competitors does not consider the accuracy related to the frustrated class, this is due to the protocol used in [44] that excludes this class because of the small number of samples. Even if we do not consider these overall values useful to understand the actual performance of a method, we have reported them to perform a comparison with [44]. It should be noted that, even using this measure, our system achieves the best results, especially when considering the model trained on augmented data. Indeed, this model is able to achieve accuracy increases of roughly 12%, even when analysing all of four affects.

system	NoFrustrated	Overall
Garber et al. [44]	66.50%	-
Kleinsmith et al. [81]	66.33%	59.22%
our LSTM R1+M1	68.80%	56.60%
our Full R1+M1, M0	**71.82%**	**62.61%**
ours Augmented	**77.72%**	**71.21%**

Table 4.16. Experimental comparisons on the UCLIC dataset with and without the frustrated class based on the overall accuracy.

4.4 Conclusions

In this chapter, extensive tests were reported for each module of the framework. In the gesture recognition module experiments, the proposed method outperforms competing works on the SHREC dataset. A new dataset was also provided, based on a large subset of the ASL, to analyse the robustness of the extracted features and the behaviour of the network when the number of stacked LSTMs change. In the action recognition experimental section, comparative experiments on RGB data showed that the proposed method outperforms the current literature on three benchmarks. While, in the remaining datasets, the method shows robust performance even in presence of noisy, or limited, data and perspective changes. Additional tests on two sets of RGB-D data showed that the method is also comparable with methods that use 3D skeletons, thus making our approach widely applicable in uncontrolled indoor and outdoor environments. Moreover, this section presented an approach based on C-RNN-GAN to support the datasets with a limited amount of data. Supplementary comparisons showed that, although the use of a 3D-CNN offers a substantial contribution, it does not exceed the performance of the 2-Branch stacked LSTM network in case of correct skeletal data acquisition. Finally, in the experiments on body affect recognition, an exhaustive experimental phase was performed on the UCLIC dataset. Tests using the different feature groups were also performed in order to show that the configuration used is the best. An augmentation data phase, carried out by applying Gaussian noise, was also illustrated, together with its results. The state-of-the-art comparison highlighted how the proposed method surpasses the other works, and is also in line with human recognition rates.

Chapter 5

Conclusions

The following section concludes this thesis work by first resuming the contribution to the state-of-the-art, and then by showing the future developments of the proposed algorithms

5.1 Contribution

In this thesis, an original framework to analyse human actions and body language is proposed. The framework is composed by three main modules: the hand gesture recognition, action recognition and non-acted body affect recognition modules.

In the hand gesture recognition module, an affective set of discriminative features based on both joint angles and fingertip positions is used in combination with an LSTM-RNN to obtain high accuracy results. The method we propose outperforms competing works on the SHREC dataset. This thesis also provides a new dataset, based on a large subset of the ASL, to train and test the effectiveness of approaches similar to that we present. This dataset has been also used to analyse the robustness of the extracted features and the behaviour of the network when the number of stacked LSTMs change.

To follow, in the action recognition module, a method based on 2D skeleton and two-branch stacked RNNs with LSTM cells is proposed. The method presents several key contributions, including an alternative approach based on 2D skeletons and RGB videos, an original pipeline based on lightweight features and 2D skeleton joints, the tracking of multiple people, and the robust management of partial body occlusions. The method, moreover, includes a supplementary network based on 3D-DenseNet to handle the missing skeleton data, and an approach based on C-RNN-GAN to support those datasets with a limited amount of data. Extensive evaluations on KTH, Weizmann, UCF Sports, IXMAS, HMDB51, UCF101, UT-Kinect, and NTU-RGB+D datasets show the effectiveness of the method. In particular, comparisons with several key works on the first two datasets highlight that the method outperforms the current literature achieving a recognition rate of 99.5% and 100%, respectively. Further investigations on the third and fourth datasets underline the successfulness of the method even in presence of incorrect 2D skeleton extraction and significant changes in camera angles and positions, respectively. Moreover, additional tests on the first and third datasets also show the robustness of the method when partial body occlusions occur. Supplementary experiments on the third, fifth, and sixth datasets highlight how the combination between our architecture and the 3D CNN-based network improves the accuracy of the proposed method, by handling missing skeletal data cases. Comparisons on the last two datasets show that the accuracy of the proposed method based on 2D skeletons is fully comparable to that of works based on 3D skeletons. Despite the excellent results, the few samples available in KTH, UCF Sports, and UT-Kinect datasets could not allow a full exploitation of the proposed architecture and features. For this reason, only for them, the thesis also reports experimental results obtained by using an effective data augmentation technique, in which Continuous Recurrent Neural Networks with Generative Adversarial Networks (C-RNN-GANs) are used to generate suitable synthetic action sequences.

Finally, as concern the body affect recognition module, an original combination of local and global temporal features is used, with a custom deep neural network architecture, to realize a non-acted body affect recognition method. An exhaustive experimental phase was performed on the only benchmark dataset available in the current literature. The obtained results shown how the proposed solution outperforms key works in this topic, thus demonstrating how time based features can improve the classification performance with respect to the single postures.

5.2 Future developments

In order to improve the proposed framework, we are currently working on several aspects. Starting from the hand gesture recognition module, we want to integrate an analysis of the upper body skeleton into the current pipeline, in order to be able to manage more complex cases, such as the Italian Sign Language (LIS). In fact, each term of these non-verbal languages is influenced both by the finger movements and by the hand position relative to the body. The studies carried out for the realization of the second module (Sec. 3.3) could help in the realization of this step, thus making the framework a valid support tool for people with muteness.

As concern the non-acted body affect recognition module, we are planning to improve the UCLIC dataset by integrating new real samples, especially for the frustrated class, and by defining new affect classes. There are two main problems to manage to achieve this goal: the first one regards the creation of a valid acquisition protocol that allows to capture the genuine emotions of the people, without conditioning them; while, the second one, regards the labelling of the body movements performed by people, which requires the support of expert researchers in the field of body language.

Moreover, we are currently working on a new module focused on human hand rehabilitation. Using the anomaly detection technique based on LSTM Auto-Encoders (AEs) [14], in this new module we will try to recognize the degree of hand disability of the patient, training the network only on 3D skeletal data obtained by healthy patients. Preliminary tests have highlighted the validity of the idea, showing how this technique allows to identify the fingers that performed wrong movements during the execution of the assigned rehabilitation exercise.

Bibliography

[1] Abadi, M. and et al. TensorFlow: Large-scale machine learning on heterogeneous systems (2015). Software available from tensorflow.org. Available from: `https://www.tensorflow.org/`.

[2] Abdulmunem, A., Lai, Y.-K., and Sun, X. Saliency guided local and global descriptors for effective action recognition. *Computational Visual Media*, **2** (2016), 97.

[3] Anirudh, R., Turaga, P., Su, J., and Srivastava, A. Elastic functional coding of human actions: From vector-fields to latent variables. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 3147–3155 (2015).

[4] Argyle, M., Alkema, F., and Gilmour, R. The communication of friendly and hostile attitudes by verbal and non-verbal signals. *European Journal of Social Psychology*, **1** (1971), 385.

[5] Avola, D., Bernardi, M., Cinque, L., Foresti, G. L., and Massaroni, C. Adaptive bootstrapping management by keypoint clustering for background initialization. *Pattern Recognition Letters*, **100** (2017), 110.

[6] Avola, D., Bernardi, M., Cinque, L., Foresti, G. L., and Massaroni, C. Exploiting recurrent neural networks and leap motion controller for the recognition of sign language and semaphoric hand gestures. *IEEE Transactions on Multimedia*, **21** (2019), 234.

[7] Avola, D., Cascio, M., Cinque, L., Foresti, G. L., Massaroni, C., and Rodolà, E. 2d skeleton-based action recognition via two-branch stacked lstm-rnns. *IEEE Transactions on Multimedia*, (2019), 1.

[8] Avola, D., Cinque, L., Foresti, G. L., Massaroni, C., and Pannone, D. A keypoint-based method for background modeling and foreground detection using a ptz camera. *Pattern Recognition Letters*, **96** (2017), 96.

[9] Avola, D., Spezialetti, M., and Placidi, G. Design of an efficient framework for fast prototyping of customized human–computer interfaces and virtual environments for rehabilitation. *Computer Methods and Programs in Biomedicine*, **110** (2013), 490 .

[10] Bagozzi, R. P., Gopinath, M., and Nyer, P. U. The role of emotions in marketing. *Journal of the Academy of Marketing Science*, **27** (1999), 184.

[11] Barret, Z. and Quoc V., L. Neural architecture search with reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pp. 1–16 (2017).

[12] Barrientos, F. A. and Canny, J. F. Cursive: Controlling expressive avatar gesture using pen gesture. pp. 113–119. Proceedings of the International Conference on Collaborative Virtual Environments (2002).

[13] Bishop, C. *Pattern Recognition and Machine Learning.* Springer (2006).

[14] Bontemps, L., Cao, V. L., McDermott, J., and Le-Khac, N.-A. Collective anomaly detection based on long short-term memory recurrent neural networks. In *Future Data and Security Engineering* (edited by T. K. Dang, R. Wagner, J. Küng, N. Thoai, M. Takizawa, and E. Neuhold), pp. 141–152 (2016).

[15] Bradski, G. and Kaehler, A. *Learning OpenCV: Computer Vision in C++ with the OpenCV Library.* O'Reilly Media, Inc. (2013).

[16] Bregonzio, M., Xiang, T., and Gong, S. Fusing appearance and distribution information of interest points for action recognition. *Pattern Recognition*, **45** (2012), 1220.

[17] Brendel, W. and Todorovic, S. Learning spatiotemporal graphs of human activities. In *International Conference on Computer Vision (ICCV)*, pp. 778–785 (2011).

[18] Bridle, J. S. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. *Neurocomputing: Algorithms, Architectures and Applications*, (1990), 227.

[19] Calinon, S. and Billard, A. Incremental learning of gestures by imitation in a humanoid robot. pp. 255–262. Proceedings of the ACM/IEEE International Conference on Human-robot Interaction (2007).

[20] Cambria, E. Affective computing and sentiment analysis. *IEEE Intelligent Systems*, **31** (2016), 102.

[21] Cao, Z., Simon, T., Wei, S. E., and Sheikh, Y. Realtime multi-person 2d pose estimation using part affinity fields. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1302–1310 (2017).

[22] Carreira, J. and Zisserman, A. Quo vadis, action recognition? a new model and the kinetics dataset. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4724–4733 (2017).

[23] Chaudhry, R., Ofli, F., Kurillo, G., Bajcsy, R., and Vidal, R. Bio-inspired dynamic 3d discriminative skeletal features for human action recognition. pp. 471–478. IEEE Conference on Computer Vision and Pattern Recognition Workshops (2013).

[24] Cheng, H., Yang, L., and Liu, Z. Survey on 3d hand gesture recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, **26** (2016), 1659.

[25] Chuankun Li, Pichao Wang, Shuang Wang, Yonghong Hou, and Wanqing Li. Skeleton-based action recognition using lstm and cnn. In *International Conference on Multimedia Expo Workshops (ICMEW)*, pp. 585–590 (2017).

[26] Chung, J., Çaglar Gülçehre, Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proceedings of the International Conference on Neural Information Processing Systems (NIPS)*, pp. 1–9 (2014).

[27] Cippitelli, E., Gambi, E., Spinsante, S., and Florez-Revuelta, F. Evaluation of a skeleton-based method for human activity recognition on a large-scale rgb-d dataset. In *Technologies for Active and Assisted Living (TechAAL)*, pp. 1–6 (2016).

[28] Collins, J., Sohl-Dickstein, J., and Sussillo, D. Capacity and trainability in recurrent neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pp. 1–17 (2017).

[29] Corneanu, C. A., Simón, M. O., Cohn, J. F., and Guerrero, S. E. Survey on rgb, 3d, thermal, and multimodal approaches for facial expression recognition: History, trends, and affect-related applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **38** (2016), 1548.

[30] Coulson, M. Attributing emotion to static body postures: Recognition accuracy, confusions, and viewpoint dependence. *Journal of Nonverbal Behavior*, **28** (2004), 117.

[31] Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., and Taylor, J. G. Emotion recognition in human-computer interaction. *IEEE Signal Processing Magazine*, **18** (2001), 32.

[32] Damásio, A. *Descartes' error: emotion, reason, and the human brain.* Quill (1994).

[33] Das Dawn, D. and Shaikh, S. H. A comprehensive survey of human action recognition with spatio-temporal interest point (stip) detector. *The Visual Computer*, **32** (2016), 289.

[34] De Smedt, Q., Wannous, H., Vandeborre, J.-P., Guerry, J., Le Saux, B., and Filliat, D. Shrec'17 track: 3d hand gesture recognition using a depth and skeletal dataset. pp. 1–7. Eurographics Workshop on 3D Object Retrieval (2017).

[35] del Rincón, J. M., Santofimia, M. J., and Nebel, J.-C. Common-sense reasoning for human action recognition. *Pattern Recognition Letters*, **34** (2013), 1849.

[36] Devanne, M., Wannous, H., Berretti, S., Pala, P., Daoudi, M., and Bimbo, A. D. 3-d human action recognition by shape analysis of motion trajectories on riemannian manifold. *IEEE Transactions on Cybernetics*, **45** (2015), 1340.

[37] Du, Y., Wang, W., and Wang, L. Hierarchical recurrent neural network for skeleton based action recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1110–1118 (2015).

[38] Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, **12** (2011), 2121.

[39] Ekman, P. and Friesen, W. V. *Facial Action Coding System (FACS): Manual.* Consulting Psychologist Press (1978).

[40] Evangelidis, G., Singh, G., and Horaud, R. Skeletal quads: Human action recognition using joint quadruples. In *International Conference on Pattern Recognition (ICPR)*, pp. 4513–4518 (2014).

[41] Eyben, F., Wöllmer, M., Schuller, B., and Graves, A. From speech to letters - using a novel neural network architecture for grapheme based asr. pp. 376–380. IEEE Workshop on Automatic Speech Recognition Understanding (2009).

[42] Fabio, D., Mauro, D., and Pietro, Z. Combining multiple depth-based descriptors for hand gesture recognition. *Pattern Recognition Letters*, **50** (2014), 101 .

[43] Gao, Z., Chen, M.-y., Hauptmann, A. G., and Cai, A. Comparing evaluation protocols on the kth dataset. In *Human Behavior Understanding (HBU)*, pp. 88–100 (2010).

[44] Garber-Barron, M. and Sei, M. Using body movement and posture for emotion detection in non-acted scenarios. In *Proceedings of the International Conference on Fuzzy Systems (FUZZ)*, pp. 1–8 (2012).

[45] Glowinski, D., Dael, N., Camurri, A., Volpe, G., Mortillaro, M., and Scherer, K. Toward a minimal representation of affective gestures. *IEEE Transactions on Affective Computing*, **2** (2011), 106.

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

[47] Gorelick, L., Blank, M., Shechtman, E., Irani, M., and Basri, R. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence*, **29** (2007), 2247.

[48] Goza, S. M., Ambrose, R. O., Diftler, M. A., and Spain, I. M. Telepresence control of the nasa/darpa robonaut on a mobility platform. pp. 623–629. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2004).

[49] Grafsgaard, J. F., Boyer, K. E., Wiebe, E. N., and Lester, J. C. Analyzing posture and affect in task-oriented tutoring. In *FLAIRS Conference* (2012).

[50] Graves, A. *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer (2012).

[51] Graves, A., Jaitly, N., and r. Mohamed, A. Hybrid speech recognition with deep bidirectional lstm. pp. 273–278. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU) (2013).

[52] Graves, A., Mohamed, A., and Hinton, G. Speech recognition with deep recurrent neural networks. In *International Conference on Acoustics, Speech and Signal Processing (ASSP)*, pp. 6645–6649 (2013).

[53] Greenwald, M., Cook, E., and J. Lang, P. Affective judgment and psychophysiological response: Dimensional covariation in the evaluation of pictorial stimuli. *Journal of Psychophysiology*, **3** (1989), 51.

[54] Griffin, H. J., Aung, M. S. H., Romera-Paredes, B., McLoughlin, C., McKeown, G., Curran, W., and Bianchi-Berthouze, N. Perception and automatic recognition of laughter from whole-body motion: Continuous and categorical perspectives. *IEEE Transactions on Affective Computing*, **6** (2015), 165.

[55] Guna, J., Jakus, G., Pogačnik, M., Tomažič, S., and Sodnik, J. An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking. *Sensors*, **14** (2014), 3702.

[56] Hand, D. and Yu, K. Idiot's bayes: Not so stupid after all? *International Statistical Review*, **69** (2007), 385.

[57] Hecht-Nielsen. Theory of the backpropagation neural network. In *International 1989 Joint Conference on Neural Networks*, pp. 593–605 vol.1 (1989).

[58] Hermans, M. and Schrauwen, B. Training and analyzing deep recurrent neural networks. In *Proceedings of the International Conference on Neural Information Processing Systems (NIPS)*, pp. 190–198 (2013).

[59] Hinton, G., et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, **29** (2012), 82.

[60] Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computing*, **9** (1997), 1735.

[61] Hou, R., Chen, C., and Shah, M. Tube convolutional neural network (t-cnn) for action detection in videos. In *International Conference on Computer Vision (ICCV)*, pp. 5823–5832 (2017).

[62] Hu, J., Zheng, W., Lai, J., and Zhang, J. Jointly learning heterogeneous features for rgb-d activity recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5344–5352 (2015).

[63] Huang, G., Liu, Z., v. d. Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Computer Vision and Pattern Recognition (CVPR)* (2017).

[64] Huang, Z., Wan, C., Probst, T., and Gool, L. V. Deep learning on lie groups for skeleton-based action recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1243–1252 (2017).

[65] Ibrahim, M. S., Muralidharan, S., Deng, Z., Vahdat, A., and Mori, G. A hierarchical deep temporal model for group activity recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1971–1980 (2016).

[66] Jain, S. and Aggarwal, J. K. Facial expression recognition with temporal modeling of shapes. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*, pp. 1642–1649 (2011).

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

[68] Jia, J., Zhou, S., Yin, Y., Wu, B., Chen, W., Meng, F., and Wang, Y. Inferring emotions from large-scale internet voice data. *IEEE Transactions on Multimedia*, (2018), 1.

[69] Jing, C., Potgieter, J., Noble, F., and Wang, R. A comparison and analysis of rgb-d cameras' depth performance for robotics application. In *Mechatronics and Machine Vision in Practice (M2VIP)*, pp. 1–6 (2017).

[70] Jozefowicz, R., Zaremba, W., and Sutskever, I. An empirical exploration of recurrent network architectures. In *Proceedings of the International Conference on International Conference on Machine Learning - Volume 37*, pp. 2342–2350 (2015).

[71] Kapoor, A., Burleson, W., and Picard, R. W. Automatic prediction of frustration. *International Journal of Human-Computer Studies*, **65** (2007), 724 .

[72] Kapur, A., Kapur, A., Virji-Babul, N., Tzanetakis, G., and Driessen, P. F. Gesture-based affective computing on motion capture data. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pp. 1–7 (2005).

[73] Karg, M., Samadani, A., Gorbet, R., Kühnlenz, K., Hoey, J., and Kulić, D. Body movements for affective expression: A survey of automatic recognition and generation. *IEEE Transactions on Affective Computing*, **4** (2013), 341.

[74] Kay, W., et al. The kinetics human action video dataset. *arXiv*, **abs/1705.06950** (2017).

[75] Kendon, A. *Visible Action as Utterance.* Cambridge University Press (2004).

[76] Kim, T., Lee, J., and Nam, J. Comparison and analysis of samplecnn architectures for audio classification. *IEEE Journal of Selected Topics in Signal Processing*, **13** (2019), 285.

[77] Kim, T., Shakhnarovich, G., and Livescu, K. Fingerspelling recognition with semi-markov conditional random fields. pp. 1521–1528. IEEE International Conference on Computer Vision (ICCV) (2013).

[78] Kim, Y. Convolutional neural networks for sentence classification. In *Empirical Methods in Natural Language Processing* (2014).

[79] Kleinsmith, A. and Bianchi-Berthouze, N. Recognizing affective dimensions from body posture. In *Affective Computing and Intelligent Interaction* (edited by A. C. R. Paiva, R. Prada, and R. W. Picard), pp. 48–58 (2007).

[80] Kleinsmith, A. and Bianchi-Berthouze, N. Affective body expression perception and recognition: A survey. *IEEE Transactions on Affective Computing*, **4** (2013), 15.

[81] Kleinsmith, A., Bianchi-Berthouze, N., and Steed, A. Automatic recognition of non-acted affective postures. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **41** (2011), 1027.

[82] Kleinsmith, A., Silva, P. R. D., and Bianchi-Berthouze, N. Cross-cultural differences in recognizing affect from body posture. *Interacting with Computers*, **18** (2006), 1371.

[83] Korpusik, M. and Glass, J. Spoken language understanding for a nutrition dialogue system. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **25** (2017), 1450.

[84] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, pp. 1097–1105 (2012).

[85] Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., and Serre, T. HMDB: a large video database for human motion recognition. In *International Conference on Computer Vision (ICCV)* (2011).

[86] Lan, T., Wang, Y., and Mori, G. Discriminative figure-centric models for joint action localization and recognition. In *International Conference on Computer Vision (ICCV)*, pp. 2003–2010 (2011).

[87] Lange, J. and Lappe, M. The role of spatial and temporal information in biological motion perception. *Advances in cognitive psychology*, **3** (2007), 419.

[88] Lavee, G., Rudzsky, M., and Rivlin, E. Propagating certainty in petri nets for activity recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, **23** (2013), 326.

[89] Laxton, B., Lim, J., and Kriegman, D. Leveraging temporal, contextual and ordering constraints for recognizing complex activities in video. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8 (2007).

[90] Le, Q. V., Zou, W. Y., Yeung, S. Y., and Ng, A. Y. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 3361–3368 (2011).

[91] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86** (1998), 2278.

[92] Lee, D.-H. and Hong, K.-S. Game interface using hand gesture recognition. pp. 1092–1097. International Conference on Computer Sciences and Convergence Information Technology (ICCIT) (2010).

[93] Li, C., Hou, Y., Wang, P., and Li, W. Joint distance maps based action recognition with convolutional neural networks. *IEEE Signal Processing Letters*, **24** (2017), 624.

[94] Li, D., Yao, T., Duan, L., Mei, T., and Rui, Y. Unified spatio-temporal attention networks for action recognition in videos. *IEEE Transactions on Multimedia*, **21** (2019), 416.

[95] Li, K. and Fu, Y. Prediction of human activity by discovering temporal sequence patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **36** (2014), 1644.

[96] Li, P., Ling, H., Li, X., and Liao, C. 3d hand pose estimation using randomized decision forest with segmentation index points. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 819–827 (2015).

[97] Li, X., Hu, W., Shen, C., Zhang, Z., Dick, A., and Hengel, A. V. D. A survey of appearance models in visual object tracking. *ACM Trans. Intell. Syst. Technol.*, **4** (2013), 2157.

[98] Li, Y., Li, W., Mahadevan, V., and Vasconcelos, N. Vlad3: Encoding dynamics of deep features for action recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1951–1960 (2016).

[99] Liu, J., Shahroudy, A., Xu, D., and Wang, G. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European Conference on Computer Vision (ECCV)*, pp. 816–833 (2016).

[100] Lo Presti, L., La Cascia, M., Sclaroff, S., and Camps, O. Gesture modeling by hanklet-based hidden markov model. In *Asian Conference on Computer Vision (ACCV)*, pp. 529–546 (2015).

[101] Lu, W., Tong, Z., and Chu, J. Dynamic hand gesture recognition with leap motion controller. *IEEE Signal Processing Letters*, **23** (2016), 1188.

[102] Luo, Z., Hsieh, J.-T., Jiang, L., Niebles, J. C., and Fei-Fei, L. Graph distillation for action detection with privileged modalities. In *ECCV* (edited by V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss), pp. 174–192 (2018).

[103] Luvizon, D. C., Picard, D., and Tabia, H. 2d/3d pose estimation and action recognition using multitask deep learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5137–5146 (2018).

[104] Maas, L., Hannun, A. Y., and Ng., A. Y. Rectifier nonlinearities improve neural network acousticmodels. In *International Conference on Machine Learning (ICML)*, pp. 1–6 (2016).

[105] Marin, G., Dominio, F., and Zanuttigh, P. Hand gesture recognition with jointly calibrated leap motion and depth sensor. *Multimedia Tools Appl.*, **75** (2016), 14991.

[106] Mavani, V., Raman, S., and Miyapuram, K. P. Facial expression recognition using visual saliency and deep learning. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*, pp. 2783–2788 (2017).

[107] McCallum, A., Nigam, K., Rennie, J., and Seymore, K. A machine learning approach to building domain-specific search engines. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2*, pp. 662–667 (1999).

[108] Meeren, H. K. M., van Heijnsbergen, C. C. R. J., and de Gelder, B. Rapid perceptual integration of facial expression and emotional body language. *Proceedings of the National Academy of Sciences of the United States of America*, **102** (2005), 16518.

[109] Mehrabian, A. Significance of posture and position in the communication of attitude and status relationships. *Psychological Bulletin*, **71** (1969), 359.

[110] Michael, A. *Bodily Communication.* Methuen (1988).

[111] Misra, I., Zitnick, C. L., and Hebert, M. Shuffle and learn: Unsupervised learning using temporal order verification. In *European Conference on Computer Vision (ECCV)*, pp. 527–544 (2016).

[112] Moeslund, T. B., Hilton, A., and Krüger, V. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, **104** (2006), 90.

[113] Mogren, O. C-rnn-gan: A continuous recurrent neural network with adversarial training. In *Neural Information Processing Systems (NIPS)*, pp. 1–6 (2016).

[114] Mollahosseini, A., Hasani, B., and Mahoor, M. H. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, **10** (2019), 18.

[115] Naguri, C. R. and Bunescu, R. C. Recognition of dynamic hand gestures from 3d motion data using lstm and cnn architectures. In *IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 1130–1133 (2017).

[116] Neil, D., Pfeiffer, M., and Liu, S.-C. Phased lstm: Accelerating recurrent network training for long or event-based sequences. In *Advances In Neural Information Processing Systems (NIPS)*, pp. 3882–3890 (2016).

[117] Nelson, B., Barreno, M., Chi, F. J., Joseph, A. D., Rubinstein, B. I. P., Saini, U., Sutton, C., Tygar, J. D., and Xia, K. Exploiting machine learning to subvert your spam filter. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, pp. 7:1–7:9 (2008).

[118] Neverova, N., Wolf, C., Taylor, G., and Nebout, F. Moddrop: adaptive multi-modal gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32** (2016), 1692.

[119] Noroozi, F., Kaminska, D., Corneanu, C., Sapinski, T., Escalera, S., and Anbarjafari, G. Survey on emotional body gesture recognition. *IEEE Transactions on Affective Computing*, (2018), 1.

[120] Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., and Bajcsy, R. Sequence of the most informative joints (smij): A new representation for human skeletal action recognition. *Journal of Visual Communication and Image Representation*, **25** (2014), 24.

[121] Ohn-Bar, E. and Trivedi, M. M. Joint angles similarities and hog2 for action recognition. pp. 465–470. IEEE Conference on Computer Vision and Pattern Recognition Workshops (2013).

[122] Ohn-Bar, E. and Trivedi, M. M. The power is in your hands: 3d analysis of hand gestures in naturalistic video. pp. 912–917. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2013).

[123] Ohn-Bar, E. and Trivedi, M. M. Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE Transactions on Intelligent Transportation Systems*, **15** (2014), 2368.

[124] Onofri, L., Soda, P., Pechenizkiy, M., and Iannello, G. A survey on using domain and contextual knowledge for human activity recognition in video streams. *Expert Systems with Applications*, **63** (2016), 97.

[125] Oreifej, O. and Liu, Z. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. pp. 716–723. IEEE Conference on Computer Vision and Pattern Recognition (2013).

[126] Parisi, G. I., Tani, J., Weber, C., and Wermter, S. Lifelong learning of human actions with deep neural network self-organization. *Neural Networks*, **96** (2017), 137.

[127] Pei, M., Jia, Y., and Zhu, S. C. Parsing video events with goal inference and intent prediction. In *International Conference on Computer Vision (ICCV)*, pp. 487–494 (2011).

[128] Pham, H.-H., Khoudour, L., Crouzil, A., Zegers, P., and Velastin, S. Exploiting deep residual networks for human action recognition from skeletal data. *Computer Vision and Image Understanding*, **170** (2018), 51.

[129] Pierce, J. S. and Pausch, R. Comparing voodoo dolls and homer: Exploring the importance of feedback in virtual environments. pp. 105–112. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2002).

[130] Placidi, G., Avola, D., Iacoviello, D., and Cinque, L. Overall design and implementation of the virtual glove. *Computers in Biology and Medicine*, **43** (2013), 1927.

[131] Pollick, F., Paterson, H., and Mamassian, P. Combining faces and movements to recognize affect. *Journal of Vision*, **4** (2010), 232.

[132] Poppe, R. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, **108** (2007), 4.

[133] Poppe, R. A survey on vision-based human action recognition. *Image and Vision Computing*, **28** (2010), 976.

[134] Presti, L. L. and Cascia, M. L. 3d skeleton-based human action classification: A survey. *Pattern Recognition*, **53** (2016), 130.

[135] Qi, J., Wang, Z., Lin, X., and Li, C. Learning complex spatio-temporal configurations of body joints for online activity recognition. *IEEE Transactions on Human-Machine Systems*, **48** (2018), 637.

[136] Qiu, Z., Yao, T., and Mei, T. Learning deep spatio-temporal dependence for semantic video segmentation. *IEEE Transactions on Multimedia*, **20** (2018), 939.

[137] Quek, F., McNeill, D., Bryll, R., Duncan, S., Ma, X.-F., Kirbas, C., McCullough, K. E., and Ansari, R. Multimodal human discourse: Gesture and speech. *ACM Trans. Comput.-Hum. Interact.*, **9** (2002), 171.

[138] Rahmani, H., Mian, A., and Shah, M. Learning a deep model for human action recognition from novel viewpoints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **40** (2018), 667.

[139] Rautaray, S. S. and Agrawal, A. Interaction with virtual game through hand gesture recognition. pp. 244–247. 2011 International Conference on Multimedia, Signal Processing and Communication Technologies (IMPACT) (2011).

[140] Reale, M. J., Canavan, S., Yin, L., Hu, K., and Hung, T. A multi-gesture interaction system using a 3-d iris disk model for gaze estimation and an active appearance model for 3-d hand pointing. *IEEE Transactions on Multimedia*, **13** (2011), 474.

[141] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, **39** (2017), 1137.

[142] Ren, Z., Yuan, J., Meng, J., and Zhang, Z. Robust part-based hand gesture recognition using kinect sensor. *IEEE Transactions on Multimedia*, **15** (2013), 1110.

[143] Ribeiro, M., Lazzaretti, A. E., and Lopes, H. S. A study of deep convolutional auto-encoders for anomaly detection in videos. *Pattern Recognition Letters*, **110** (2018), 1.

[144] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, **65** (1958), 386.

[145] Rosten, E. and Drummond, T. Machine learning for high-speed corner detection. In *Computer Vision – ECCV 2006* (edited by A. Leonardis, H. Bischof, and A. Pinz), pp. 430–443 (2006).

[146] Roy, D. and Mohan, C. K. Snatch theft detection in unconstrained surveillance videos using action attribute modelling. *Pattern Recognition Letters*, **108** (2018), 1.

[147] Sak, H., Senior, A. W., and Beaufays, F. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. In *Proceedings of the Annual International Conference of the International Speech Communication Association (ISCA)*, pp. 338–342 (2014).

[148] Savitzky, A. and Golay, M. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, **36** (1964), 1627.

[149] Savva, N., Scarinzi, A., and Bianchi-Berthouze, N. Continuous recognition of player's affective body expression as dynamic quality of aesthetic experience. *IEEE Transactions on Computational Intelligence and AI in Games*, **4** (2012), 199.

[150] Schaaff, K. and Schultz, T. Towards emotion recognition from electroencephalographic signals. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction and Workshops (ACII)*, pp. 1–6 (2009).

[151] Schaefer, A. M., Udluft, S., and Zimmermann, H.-G. Learning long-term dependencies with recurrent neural networks. *Neurocomput.*, **71** (2008), 2481.

[152] Schramm, R., Jung, C. R., and Miranda, E. R. Dynamic time warping for music conducting gestures evaluation. *IEEE Transactions on Multimedia*, **17** (2015), 243.

[153] Schuldt, C., Laptev, I., and Caputo, B. Recognizing human actions: a local svm approach. In *International Conference on Pattern Recognition (ICPR)*, pp. 32–36 (2004).

[154] Sebastiani, F. Machine learning in automated text categorization. *ACM Comput. Surv.*, **34** (2002), 1.

[155] Shahroudy, A., Liu, J., Ng, T., and Wang, G. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1010–1019 (2016).

[156] Shahroudy, A., Ng, T., Gong, Y., and Wang, G. Deep multimodal feature analysis for action recognition in rgb+d videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **40** (2018), 1045.

[157] Shi, L., Zhang, Y., Cheng, J., and Lu, H. Skeleton-based action recognition with directed graph neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7904–7913 (2019).

[158] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. Real-time human pose recognition in parts from single depth images. pp. 1297–1304. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2011).

[159] Simonyan, K. and Zisserman, A. Two-stream convolutional networks for action recognition in videos. In *Proceedings of the International Conference on Neural Information Processing Systems*, pp. 568–576 (2014).

[160] Smedt, Q. D., Wannous, H., and Vandeborre, J. P. Skeleton-based dynamic hand gesture recognition. pp. 1206–1214. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2016).

[161] Sohn, M.-K., Lee, S.-H., Kim, D.-J., Kim, B., and Kim, H. A comparison of 3d hand gesture recognition using dynamic time warping. pp. 418–422. Proceedings of the Conference on Image and Vision Computing New Zealand (2012).

[162] Sokolova, M. and Lapalme, G. A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, **45** (2009), 427.

[163] Song, T., Zheng, W., Song, P., and Cui, Z. Eeg emotion recognition using dynamical graph convolutional neural networks. *IEEE Transactions on Affective Computing*, (2018).

[164] Soomro, K. and Zamir, A. R. Action recognition in realistic sports videos. In *Computer Vision in Sports (CVS)*, pp. 181–208 (2014).

[165] Soomro, K., Zamir, A. R., Shah, M., Soomro, K., Zamir, A. R., and Shah, M. Ucf101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, (2012).

[166] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, **15** (2014), 1929.

[167] Sun, L., Jia, K., Chen, K., Yeung, D.-Y., Shi, B. E., and Savarese, S. Lattice long short-term memory for human action recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pp. 2147–2156 (2017).

[168] Sun, Y., Zheng, L., Yang, Y., Tian, Q., and Wang, S. Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In *European Conference on Computer Vision (ECCV)* (2018).

[169] Sundermeyer, M., Ney, H., and Schlüter, R. From feedforward to recurrent lstm neural networks for language modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **23** (2015), 517.

[170] Suryanarayan, P., Subramanian, A., and Mandalapu, D. Dynamic hand pose recognition using depth data. pp. 3105–3108. International Conference on Pattern Recognition (ICPR) (2010).

[171] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9 (2015).

[172] Tamara, A. and Belén, L.-P. A longitudinal analysis of the relationship between positive and negative affect and health. *Journal of Psychology*, **5** (2014), 859.

[173] Thurau, C. and Hlavac, V. Pose primitive based human action recognition in videos or still images. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8 (2008).

[174] Tran, S. D. and Davis, L. S. Event modeling and recognition using markov logic networks. In *European Conference on Computer Vision (ECCV)*, pp. 610–623 (2008).

[175] Truong, A., Boujut, H., and Zaharia, T. Laban descriptors for gesture recognition and emotional analysis. *The Visual Computer*, **32** (2016), 83.

[176] Tu, N. A., Huynh-The, T., Khan, K. U., and Lee, Y. Ml-hdp: A hierarchical bayesian nonparametric model for recognizing human actions in video. *IEEE Transactions on Circuits and Systems for Video Technology*, **29** (2019), 800.

[177] Tu, Z., Li, H., Zhang, D., Dauwels, J., Li, B., and Yuan, J. Action-stage emphasized spatiotemporal vlad for video action recognition. *IEEE Transactions on Image Processing*, **28** (2019), 2799.

[178] Van den Stock, J., Righart, R., and de Gelder, B. Body expressions influence recognition of emotions in the face and voice. *Emotion*, **7** (2007), 487.

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

[180] Vemulapalli, R., Arrate, F., and Chellappa, R. Human action recognition by representing 3d skeletons as points in a lie group. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 588–595 (2014).

[181] Vikram, S., Li, L., and Russell, S. Writing and sketching in the air, recognizing and controlling on the fly. pp. 1179–1184. Extended Abstracts on Human Factors in Computing Systems (2013).

[182] Wang, C., Liu, Z., and Chan, S. C. Superpixel-based hand gesture recognition with kinect depth camera. *IEEE Transactions on Multimedia*, **17** (2015), 29.

[183] Wang, D., Ouyang, W., Li, W., and Xu, D. Dividing and aggregating network for multi-view action recognition. In *European Conference on Computer Vision (ECCV)*, pp. 457–473 (2018).

[184] Wang, H., Kläser, A., Schmid, C., and Liu, C. L. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 3169–3176 (2011).

[185] Wang, H., Ullah, M. M., Klaser, A., Laptev, I., and Schmid, C. Evaluation of local spatio-temporal features for action recognition. In *British Machine Vision Conference (BMVC)*, pp. 1–11 (2009).

[186] Wang, J., Liu, Z., Chorowski, J., Chen, Z., and Wu, Y. Robust 3d action recognition with random occupancy patterns. pp. 872–885. Proceedings of the European Conference on Computer Vision (2012).

[187] Wang, L., Li, W., Li, W., and Van Gool, L. Appearance-and-relation networks for video classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1430–1439 (2018).

[188] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Van Gool, L. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision (ECCV)*, pp. 20–36 (2016).

[189] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Van Gool, L. Temporal segment networks for action recognition in videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **41** (2019), 2740.

[190] Wang, S. B., Quattoni, A., Morency, L. P., Demirdjian, D., and Darrell, T. Hidden conditional random fields for gesture recognition. vol. 2, pp. 1521–1527. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2006).

[191] Wang, X. and Ji, Q. Incorporating contextual knowledge to dynamic bayesian networks for event recognition. In *International Conference on Pattern Recognition (ICPR)*, pp. 3378–3381 (2012).

[192] Weichert, F., Bachmann, D., Rudak, B., and Fisseler, D. Analysis of the accuracy and robustness of the leap motion controller. *Sensors*, **13** (2013), 6380.

[193] Weinland, D., Boyer, E., and Ronfard, R. Action recognition from arbitrary views using 3d exemplars. In *International Conference on Computer Vision (ICCV)*, pp. 1–7 (2007).

[194] Weinzaepfel, P., Harchaoui, Z., and Schmid, C. Learning to track for spatio-temporal action localization. In *International Conference on Computer Vision (ICCV)*, pp. 3164–3172 (2015).

[195] Wu, D., Pigou, L., Kindermans, P., Le, N. D., Shao, L., Dambre, J., and Odobez, J. Deep dynamic neural networks for multimodal gesture segmentation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **38** (2016), 1583.

[196] Xia, L., Chen, C., and Aggarwal, J. View invariant human action recognition using histograms of 3d joints. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 20–27 (2012).

[197] Yan, S., Xiong, Y., and Lin, D. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI* (2018).

[198] Yang, X. and Tian, Y. Super normal vector for activity recognition using depth sequences. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 804–811 (2014).

[199] Yao, H., Ge, C., Xue, J., and Zheng, N. A high spatial resolution depth sensing method based on binocular structured light. *Sensors*, **17** (2017), 1.

[200] Zang, J., Wang, L., Liu, Z., Zhang, Q., Hua, G., and Zheng, N. Attention-based temporal weighted convolutional neural network for action recognition. In *Artificial Intelligence Applications and Innovations*, pp. 97–108 (2018).

[201] Zeng, Z., Pantic, M., Roisman, G. I., and Huang, T. S. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **31** (2009), 39.

[202] Zhang, B., Wang, L., Wang, Z., Qiao, Y., and Wang, H. Real-time action recognition with deeply transferred motion vector cnns. *IEEE Transactions on Image Processing*, **27** (2018), 2326.

[203] Zhang, C. and Tian, Y. Histogram of 3d facets: A depth descriptor for human action and hand gesture recognition. *Computer Vision and Image Understanding*, **139** (2015), 29 .

[204] Zhang, K., Zhang, L., and Yang, M.-H. Real-time compressive tracking. In *European Conference on Computer Vision (ECCV)*, pp. 864–877 (2012).

[205] Zhang, Z. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, **19** (2012), 4.

[206] Zhao, W., Chellappa, R., Phillips, P. J., and Rosenfeld, A. Face recognition: A literature survey. *ACM Computing Surveys*, **35** (2003), 399.

[207] Zhirong Wu, Song, S., Khosla, A., Fisher Yu, Linguang Zhang, Xiaoou Tang, and Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1912–1920 (2015).

[208] Zhu, H. M. and Pun, C. M. Real-time hand gesture recognition from depth image sequences. pp. 49–52. International Conference on Computer Graphics, Imaging and Visualization (CGIV) (2012).

[209] Zhu, Y., Chen, W., and Guo, G. Fusing spatiotemporal features and joints for 3d action recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 486–491 (2013).