Comparison of motion-based approaches for multi-modal action and gesture recognition from RGB-D

Hugo Bertiche Argila

A thesis submitted in fulfillment for the
Master in Artificial Intelligence

in the

Facultad d'Informàtica de Barcelona
Universitat Politècnica de Catalunya

Supervisor: Prof. Sergio Escalera Guerrero
Co-supervisor: Maryam Asadi

May 2017

Abstract

Automatic action and gesture recognition research field has growth in interest over the last few years. Action recognition can be understood as the automatic classification of generic human actions or activities, such as walking, reading, jumping, etc. while gesture recognition focuses on the analysis of more concrete movements, usually from the upper body, which have a meaning by their own, as waving, saluting, negating, etc. Such interest on the domain comes mainly from its many applications, which include, human-computer interaction, ambient assisted living systems, health care monitoring systems, surveillance, communications, entertainment, etc. This concrete domain shares many similarities with object recognition from still images, nevertheless, it has shown a special characteristic that turns it into a very challenging task. That is, the temporal evolution of actions and gestures. The scenario that is found nowadays into the author's community is a competition on finding out how to deal with this extra dimensionality. Therefore, the project starts with an exhaustive state-of-the-art analysis, where the most common approaches for dealing with time are summarized. Hand-crafted features rely on the extension of 2D descriptors, such as HoG or SIFT to a third dimension (time) and also the definition of descriptors based on motion features, such as optical flow or scene flow, meanwhile, deep learning models can be categorized into four non-mutually exclusive categories according on how they deal with time: 2D CNNs that perform recognition on still images from videos, averaging results for each of them, 2D CNNs applied over motion features, 3D CNNs able to compute 3D convolutions over 2 spatial dimension and 1 temporal dimension and neural networks which can model temporal evolution, such as RNN and LSTM. After reviewing the literature, a selection and testing of some of this methods is performed to find the direction in which should point the future research on the domain. Additionally, the recent increase on availability of depth sensors (Microsoft's Kinnect V1 and V2) allow the exploration of multi-modal techniques that take advantage of multiple data sources (RGB and depth). The domain's background has shown how many algorithms can benefit from this extra modality, by itself or combining with classical RGB. For these reasons, it is mandatory to test as well techniques that rely on multi-modal data, to do so, one of the algorithms selected has been modified to use both, RGB data and depth maps. Hand-crafted algorithms still compete with deep learning approaches in this challenging domain, as neural networks require a much higher complexity to deal with the extra temporal dimension, which implies an increase of the number of parameters to learn by the model, therefore, larger datasets and computational resources are necessary, nevertheless, for this domain, datasets are still sparse and few, that is why many authors propose different workarounds, like pre-training on image recognition datasets or multi-task learning that allows the models to learn from several datasets at once. Due to this situation, the algorithms tested into the scope of this project are of both types, hand-crafted features and deep based models. Also, a late fusion strategy is tested to see how well can be combined the results of both kind of approaches. Finally, the results obtained are compared with other state-of-the-art techniques applied over the same datasets along with a conclusion on the topic.

Contents

Chapter 1

Introduction

Automatic action and gesture recognition are fundamental parts of human behavior analysis, which has significantly grown in interest over the last few years. The former is understood as recognizing generic human actions or activities (e.g. walking, reading, jumping...), meanwhile the latter focuses on the analysis of more concrete movement, usually from the upper body, which have a meaning by their own (e.g. waving, saluting, negating, ...). Both tasks share many similarities, as both are based on body posture and movement through a set of video frames.

The interest in this field of research is due to its many domains of application. Starting with human computer interaction, where the capability of machines to understand gestures enhances the user experience [1]. Following ambient assisted living systems, which improve the quality of independent living of elderly or disabled people by using action recognition to ensure their safety and well-being [2]. Similarly, health care monitoring systems rely also on action recognition for medical purposes (fall detection, seizures, ...) to reduce the personnel's response time [3]. Also in the security and surveillance domain, where reducing the need of human operators makes possible the inspection of the increasing volume of security camera videos [4]. In communications, as gesture recognition allows automatic sign language interpretation [5]. And finally, entertainment systems, such as kinect, which allows the user to interact with its gaming device or computer via gestures.

The problem being addressed is somehow similar to object recognition or image classification, nevertheless, it has a special aspect that characterizes it, i.e., the temporal dimension. Distinct approaches deal differently with time, and how are extended the already existing methodologies to this new dimension shall have a direct impact on its performance. This extra dimension significantly increases the difficulty of the problem due to the required amount of data to process and the necessary complexity of the models. Different strategies on how to add the temporal dimension had been proposed, video summarization [6], aggregation of local frame-level features into mid-level video representations [7], or temporal sequence modeling [8], among others.

Action and gesture recognition is a highly challenging research topic. It became a focus of research for computer vision and pattern recognition fields around two decades ago [9]. Classical approaches relied on hand-crafted features over RGB data. Since the appearing of the topics, substantial progress has been made on both tasks. Furthermore, deep learning based models, as in many other computer vision tasks, have been applied for action and gesture recognition, outperforming state-of-the-art techniques [10][11].

Additionally, nowadays, depth sensors availability has greatly increased, yielding new data sources for

multi-modal approaches to address both, action and gesture recognition. This has called the attention of many authors [12]. The main advantage shown by depth sensors is their capability to capture 3D structural information of the objects, which might be more discriminative for action and gesture recognition. Another characteristic is its invariance to illumination changes, even in complete darkness, which is very useful for applications such as surveillance or patient monitoring, that need to be working 24/7. Depth maps had been successfully used for tasks such as skeleton estimation [13], which in turns facilitates automatic action and gesture recognition tasks.

Object recognition hand-crafted features had been extended to spatio-temporal features for action recognition by using 3D patches over the video frames instead of regular 2D patches over single images, as shown in [7]. For the newly available data source, depth maps, hand-crafted features had been successfully used as well for 3D object recognition [14]. Therefore, by combining both concepts, it is possible to have as well spatio-temporal features over depth maps, which provide information about the shape of the objects over the time dimension, as it has been previously done in [15]. Moreover, the combination of RGB and depth data makes possible the estimation of a 3D motion field, i.e., scene flow [16], as a counter part of the classic 2D optical flow, computed over RGB data. Scene flow advantages are the possibility to track movements along Z axis (camera axis), and, as it is computed on real world coordinates, becomes invariant to the distance from the camera to the moving object (apart from depth sensor error, which increases with distance), and it has already being used on deep learning based models for action recognition [17]. Moreover, it allows the computation of 3D motion-based features by extending the concept of Histogram of Oriented Optical Flow.

Summarizing, action and gesture recognition is a difficult task which classically has been addressed using RGB data only. Nevertheless, depth sensors are becoming more and more accessible, and many authors had successfully developed methodologies based on this data type which enhances the performance of the models. Furthermore, the combination of both RGB and depth data provides significant advances in fields like object recognition [18], and also in action and gesture recognition [19] [20]. Therefore, it is expected that exploring this new set of possibilities will show the path to follow in the future on how to develop better models for this task. And this is the goal of this document, to explore and analyze the performance of the different existing methodologies to define the best approach for this challenging problem.

The structure of the document is as follows. Chapter 2 focuses on state of the art on motion features and Convolutional Neural Networks applied on automatic action and gesture recognition. Next, chapter 3 describes the methodologies selected for the experiments and their configurations. Finally, chapter 4 presents the results obtained along a discussion and a final conclusion.

Chapter 2

Background

In the past years, many authors had published different methods for automatic action and gesture recognition. This field of research has grown considerably, yet still a long way to go. Although generally deep based models significantly outperform hand-crafted features, in this challenging domain, there is still many state of the art approaches that do not rely on learnt features. This methodologies usually extend 2D image descriptors to 3D descriptors to model spatio-temporal features [21] [7] [15], another technique consists on the computation of descriptors over motion features for implicit modeling of movement [22] [23], and the combination of these descriptors over densely sampled trajectories [23] [24] [25].

Deep learning models have to face with an extra dimension (temporal dimension), which hugely increases both data amount to be processed and model complexity, key aspects for the training of parametric deep learning networks. The problem was first addressed using RNN, but these models showed some major mathematical difficulties [26]. Later on, authors developed the long short-term memory (LSTM) cells for RNN [27], which are nowadays an important element on deep based models for image sequence modeling for action and gesture recognition [28] [29]. Additional strategies include implicit spatio-temporal modeling of features using 3D convolutional networks [30] [31], pre-computed motion-based features [10] [32], or the combination of multiple visual [33].

Based on this, it is possible to separate action and gesture recognition state of the art techniques in two groups: hand-crafted and deep based. Inside the latter, depending on how they deal with the temporal dimension, they can be grouped into four non-mutually exclusive categories: 2D CNNs which exploit spatial information, 2D CNNs fed on motion features like optical flow, 3D CNNs which use 3D convolutional filters to learn spatio-temporal features and 2D convolutional networks which are applied to single frames, followed by a temporal sequence modeling. Four groups can be combined as well with hand-crafted features to boost their performance [34].

2.1 Hand-crafted features

As stated before, hand-crafted features for automatic action and gesture recognition are temporal extensions of the classical hand-crafted features used in computer vision for object recognition and similar tasks.

2.1.1 Extended Spatio-Temporal Features

In [21], authors proposed an extension of the SIFT descriptor to 3D (being time the third dimension). To do so, besides the computation of image gradient on axis X and Y, it is also performed a gradient estimation over the time dimension. This gives a three-dimensional field vector, which in turn allows the construction of a 2D histogram (3D orientations represented with θ, ϕ and magnitude). Just as 2D SIFT, the 3D neighborhood around a point of interest can be rotated so that the dominant orientation has $\theta = 0$ and $\phi = 0$, then, sub-histograms are built around the interest point in 3D cells of $4 \times 4 \times 4$ which encode spatio-temporal information, thus creating a descriptor invariant to orientation (time orientation as well), which presumably can better generalize the underlying information to discriminate actions and gestures.

Similarly, authors of [35] introduce the idea of HOG-3D. To achieve scale invariance, they propose the computation of the descriptor over different spatial and temporal scales. Nevertheless, this scale variations imply a geometric increase on the data to compute. That is why they also describe an extension of the popular technique of integral images to integral videos to reduce computational cost. Finally, instead of a 2D histogram (which has distortion issues on the vertical dimension), they use the tessellation of a sphere into a dodecahedron and into an icosahedron, yielding bins with equivalent solid angle, and therefore, no distortion or weighting.

Finally, in [15], the HON descriptor, which encodes the 3D information of the objects, is extended to include the time dimension as well. HON descriptor is computed over depth maps, and it has an additional dimension, therefore, it is said to be able to better capture the information than other descriptors such as HOG, applied to RGB data. Just as SIFT and HOG descriptors can be extended to a new dimension (temporal), also is HON, by adding as well the depth temporal gradient. Nevertheless, surface normal vectors already are in 3D, therefore, their extension yields 4-dimensional vectors. In order to quantize them into bins, the authors use a regular polychoron, the four-dimensional analogue of the polygon, with 120 vertices (final dimensionality of the histogram).

Another approaches for dealing with the temporal dimension is, as seen in [24], to compute regular 2D descriptors, such as HOG, in every frame and later on construct histograms over 3D cells, aggregating therefore the 2D orientations of several frames into the same histogram.

2.1.2 Flow-Based Features

The descriptors explained until now are able to capture the spatio-temporal appearance of the objects, but they are not able to encode the information about the movement through the video frames, or at least not directly. In order to do so, appear motion features, which are based on the detection and quantization of the displacement of the objects on videos.

When talking about RGB data, it is not possible to estimate the real movement of the objects in the 3D real world. Nevertheless, it is possible to track the pixels and compute their displacements. This is called optical flow. Given two consecutive frames of a video, optical flow is a motion field that translates each pixel from the first frame to its corresponding location on the second frame. Optical flow has proven to be useful for automatic action and gesture recognition before [36] [25] [24] [23]. Among its applications there is pixel tracking for trajectory construction and also the creation of new descriptors. Analogously to HOG descriptor, appears the Histogram of Oriented Optical Flow (HOOF or HOF). Optical flow is a two-dimensional motion field, just like image gradient, and it is possible to use the same procedure used for HOG to construct a histogram, sometimes with an additional bin to count occurrences with small

magnitude (almost no movement). This descriptor has been also successfully applied in the domain of action and gesture recognition [22] [25] [24]. Additionally, [23] proposes a descriptor called Motion Boundary Histogram (MBH), this is computed from the gradient of the optical flow, and as its name indicates, highlights the boundaries on the motion field. This descriptor has outperformed the HOF descriptor due to its robustness to camera motion, which, as it is close to be constant through the whole image, is suppressed by the computation of the gradient. The construction of the descriptor is also very similar to the construction of HOG, but optical flow has two components, X and Y, therefore, gradient has to be computed over both dimensions, so MBH is actually the concatenation of MBHx and MBHy, which correspond to the histograms of both optical flow gradients.

Once depth sensors made it into this scenario, new possibilities for motion features arises. Microsoft's depth sensors, Kinect V1 and Kinect V2, provide of depth maps in millimeters. Then, given the Field Of View (FOV) of the camera and applying trigonometry, it is possible to create a correspondence along pixels and real world coordinates (relative to camera position). Therefore, the concept of optical flow can be extended to scene flow [37], which can be understood as the real world 3D motion field. From this perspective, optical flow can be seen as the projection of scene flow into the camera's image plane. Analogously to optical flow, it can be used to track points and construct their trajectories, and also to compute new descriptors. Just as HOF, for scene flow it is possible to compute a Histogram of Oriented Scene Flow (HOSF). As this is a 3D motion field, the methodology to construct the histogram is the same as the aforementioned 3D gradients for hand-crafted features over videos (SIFT 3D, HOG 3D, ...). Authors in [16] showed that scene flow can be fast and efficiently computed through the use of a GPU device, achieving a real-time performance, which is a very important advantage for certain applications that might require such characteristic.

2.2 Deep learning

The most crucial challenge in deep-based models for automatic human action and gesture recognition is how to deal with the temporal dimension. According to this, the deep models found in literature can be grouped into four non-mutually exclusive categories, as stated before.

2.2.1 2D Convolutional Neural Networks

This category consist on 2D CNNs, which basically rely on appearance (spatial information). These approaches sample one or more frames from the whole video and apply pre-trained 2D models on each of these frames separately. Then, by averaging the results of the sampled frames, a label for the action is obtained. The main advantage of this approach is the possibility to use pre-trained models on larger image datasets. Gesture recognition methods often fall in this category [38] [39] [40]. Some works further explore the possibility of using several frames as input. Authors of [41] analyzed the several alternatives for considering multiple frames in a 2D model; nevertheless, their work concluded that there was not gain in performance over averaging single frame predictions. Instead, [42] randomly samples video frames from K equal width temporal segments, obtain K class score predictions, compute the consensus scores, and use these in the loss function to learn from video representations directly, instead from one frame or one stack of frames. Authors in [43] convolve each frame of the video sequence to obtain frame-level CNN features. They then perform spatio-temporal pooling on pre-defined spatial regions over the set

of randomly sampled frames (50-120 depending on the sequence) in order to construct a video-level representation, which is later $L2$-normalized and classified using SVM. [44] proposes to model scene, object, and more generic feature representations using separate convolutional streams. For each frame, the three obtained representations are averaged and input to a three-layer fully connected network which provides the final output. In [45], authors collapse the videos into dynamic images, that can be fed into CNNs for image classification, by using rank pooling [46]. Dynamic images are simply the parameters of a ranking function that are learned to order the video frames. In [47], the authors propose a CNN, not to classify actions in depth data directly, but to model poses in a view-invariant high-dimensional space. For this purpose, they generate a synthetic dataset of 3D poses from motion capture data that are later fit with a puppet model and projected to depth maps. The network is first trained to differentiate among hundreds of poses to, then, use the features of the penultimate fully-connected layer for action classification in a non-deep action recognition approach. Authors of [48] exploit the combination of CNNs and LSTM for interactional object parsing on individual frames. Note LSTMs are not used for temporal sequence modeling but for refining object detections. For the action detection task, they then use object detections for pooling improved dense trajectories extracted on temporal segments.

2.2.2 Motion-based features

Researchers found that motion based features, such as optical flow, were a rich cue that could be fed directly as a network input. There are accurate and efficient methods to compute these kind of features, some of them by exploiting GPU capabilities [49]. The use of optical flow demonstrated to boost the performance of CNNs on action recognition-related tasks [10] [50] [51] [52].

Authors in [10] present a two-stream CNN which incorporated both spatial (video frames) and temporal networks (pre-computed optical flow), and show that the temporal networks trained on dense optical flow are able to obtain very good performance in spite of having limited training data. Along the same lines, in [53], authors propose a two-stream (spatial and temporal) net for non-action classification in temporal action localization. Similarly, in [54] the same architecture is used for key-volume mining and classification in this case for spatio-temporal localization of actions. In [55], authors extract both appearance and motion deep features from some detected body parts instead of whole video frames. They then compute for each body part the min/max aggregation of their descriptors over time. The final representation consists of the concatenation of pooled body part descriptors on both appearance and motion cues, which is comparable to the size of a Fisher vector. [50] uses the magnitude of optical flow vectors as a multiplicative factor for the features from the last convolutional layer. This reinforces the attention of the network on the moving objects when fine-tuning the fully connected layers. [51] explores motion vectors (obtained from video compression) to replace dense optical flow. They adopt a knowledge transfer strategy from optical flow CNN to the motion vector CNN to compensate the lack of detail and noisiness of motion vectors.

Authors of [28] use a multi-stream network to obtain frame-level features. To the full-frame spatial and motion streams from [10], they add two other actor-centered (spatial and motion) streams that compute the features in the actor's surrounding bounding box obtained by a human detector algorithm. Moreover, motion features are not stacks of optical flow maps between pairs of consecutive frames, but among a central frame and neighboring ones (avoiding object's displacement along the stacked flow maps). [52] and [56] propose a similar approach for action localization. They first generate action region proposals

from RGB frames using, respectively, selective search [57] and EdgeBoxes [58]. Regions are then linked and described with static and motion CNN features. However, high quality proposals can be obtained from motion. [59] shows region proposals generated by a region proposal network (RPN) [60] from motion (optical flow) were complementary to the ones generated by an appearance RPN. Note some of the works in other sections use pre-computed motion features, which is not mutually exclusive with using motion features approaches. [61] uses stacks of 60 pre-computed optical flow maps as inputs for the 3D convolutions, largely improving results obtained using raw video frames. Authors in [62] compute motion like image representations from depth data by accumulating absolute depth differences of contiguous frames, namely hierarchical depth motion maps (HDMM).

In the literature there exist several methods which extend the deep-based methods with the popular dense trajectory features. Authors of [34] introduce a video representation called Trajectory-pooled Deep-convolutional Descriptor (TDD), which consists on extending the state-of-the-art descriptors along the trajectories with deep descriptors pooled from normalized CNN feature maps. [63] proposes a method based on a concatenation of iDT feature (HOG, HOF, MBHx, MBHy descriptors) and Fisher vector encoding and CNN features (VGG19). For CNN features they use VGG19 CNN to capture appearance features and VLAD encoding to encore/pool convolutional feature maps. [64] utilize dense trajectories, and hence motion-based features, in order to learn view-invariant representations of actions. In order to model this variance, they generate a synthetic dataset of actions with 3D puppets from MoCap data that are projected to multiple 2D viewpoints from which fisher vectors of dense trajectories are used for learning a CNN model. During its training, an output layer is placed with as many neurons as training sequences so fisher vectors from different 2D viewpoints give same response. Afterwards, the concatenation of responses in intermediate layers (except for last one) provide the view-invariant representation for actions.

Differently from other works, in [65] jointly estimates optical flow and recognize actions in a multi-task learning setup. Their models consists in a residual network based on FlowNet [66] with extra additional classification layers, which learns to do both estimate optical flow and perform the classification task.

2.2.3 3D Convolutional Neural Networks

The early work of [30] introduced the novelty of inferring temporal information from raw RGB data directly by performing 3D convolutions on stacks of multiple adjacent video frames, namely 3D ConvNets. Since then, many authors try to either further improve this kind of models [31][67] [68] [69] [70] [19] or use them in combination with other hybrid deep-oriented models[71] [72] [73] [74] [75] [76].

In particular,[31] propose 3D convolutions with more modern deep architectures and fixed 3x3x3 convolution kernel size for all layers, that makes 3D convnets more suitable for large-scale video classification. In general, 3D ConvNets can be expensive to train because of the large number of parameters, especially when training with bigger datasets such as 1-M sports dataset [41] (which can take up to one month). Authors of [68] factorize the 3D convolutional kernel learning into a sequential process of learning 2D spatial convolutions in lower convolutional layers followed by learning 1D temporal convolutions in upper layers. [67] proposed initializing 3D convolutional weights using 2D convolutional weights from spatial CNN trained on ImageNET. This not only speeds up the training but also alleviates the overfitting problem on small datasets. [61] extends the length of input clips from 16 to 60 frames in order to model videos with longer temporal dimension during 3D convolutions, but reduces the input's spatial resolution to maintain

the model complexity. In [70], authors introduce a more compact 3D ConvNet for egocentric action recognition by applying 3D convolutions and 3D pooling only at the first layer. However, they do not use raw RGB frames, but stack optical flow. In the context of depth data, [19] proposes re-scaling depth image sequences to a 3D cuboid and the use of 3D convolutions to extract spatio-temporal features. The network consists of two pairs of convolutional and 3D max pooling followed by a two-layer fully-connected layer net.

3D convolutions are often used in more cumbersome hybrid deep-based approaches. Authors in [69] propose a multi-stage CNN, in this case for temporal action localization, consisting of three 3D convnets [31]: a proposal generation network that learns to differentiate background from action segments, a classification network that aims at discriminating among actions and serves as initialization for a third network, the localization network with a loss function that considers temporal overlap with the ground truth annotations. In [62], authors apply 3D ConvNets to action recognition from depth data. The authors train a separate 3D ConvNet for each Cartesian plane each of which fed with a stack of depth images constructed from different 3D rotations and temporal scales. [33] proves the combination of both 2D and 3D ConvNet can leverage the performance when performing egocentric action recognition. Authors in [76] use 3D convolutions from [31] to model short-term action features on a hierarchical framework in which linear dynamic systems (LDS) and VLAD descriptors are used to, respectively, model/represent medium- and long-range dynamics.

2.2.4 Temporal deep learning models: RNN and LSTM

The application of temporal sequence modeling techniques, such as LSTM, to action recognition show promising results in the past [77] [78]. Earlier works did not try to explicitly model the temporal information, but aggregated the class predictions got from individual frame predictions. For instance, in [10], sample 25 equally spaced frames (and their crops and flips) from each video and then average their predicted scores.

Today, we find the combination of recurrent networks, mostly LSTM, with CNN models for the task of action recognition. In [79], authors propose a new gating scheme for LSTM that takes into account abrupt changes in the internal cell states, namely differential RNN. They use different order derivatives to model the potential saliency of observed motion patterns in actions sequences. [28] presented a bi-directional LSTM, which demonstrated to improve the simpler uni-directional LSTMs. [80] introduces a fully end-to-end approach on a RNN agent which interacts with a video over time. The agent observes a frame and provides a detection decision (confidence and begin-end), to whether or not emit a prediction, and where to look next. While back-propagation is used to train the detection decision outputs, REINFORCE is required to train the other two (non-differentiable) agent policies. In [81] authors propose a deep architecture which uses 3D skeleton sequences to regularize an LSTM network (LSTM+CNN) on the video. The regularization process is done by using the output of the encoder LSTM (grounded on 3D human-skeleton training data) and by modifying the standard BPTT algorithm in order to address the constraint optimization in the joint learning of LSTM+CNN. In their most recent work,[82] explores contexts as early as possible and leverage evolution of hierarchical local features. For this, they introduce a novel architecture called deep alternative neural network (DANN) stacking alternative layers, where each alternative layer consists of a volumetric convolutional layer followed by a recurrent layer. [83] introduces a novel Fisher Vector representation for sequences derived from RNNs. Features are extracted

from input data via VGG/C3D CNN. Then a PCA/CCA dimension reduction and L2 normalization are applied and sequential feature are extracted via RNN. Finally, another PCA+L2-norm step is applied before the final classification.

Authors of [29] extend the traditional LSTM into two concurrent domains, i.e, spatio-temporal long short-term memory (ST-LSTM). In this tree structure each joint of the network receive contextual information from both neighboring joints and previous frame. [84] proposes a part aware extension of LSTM for action recognition by splitting the memory cell of the LSTM into part-based sub-cells. These sub-cells can yield the models learn the long-term patterns specifically for each part. Finally, the output of each unit is the combination of all sub-cells.

Chapter 3

Methods

In this chapter of the document the methodologies to be used will be explained in detail. On the previous chapter was mentioned that in automatic action and gesture recognition domain, hand-crafted features and deep learning models still fight for the best performance. Therefore, for this document, it has been selected two methods, pertaining to each of the categories. For the hand-crafted features side, Dense Trajectories [23] has been chosen for being widely used by the authors community. On the other hand, as deep based model, two-stream convolution neural network [10] has been chosen due to its demonstrated high performance within the domain. Finally, a late fusion strategy is applied to both approaches in order to combine their strengths into a more robust model.

3.1 Dense Trajectories

First introduced in 2011 by [24], it presented a novel framework to obtain trajectories along video frames densely sampled, improving both quality and quantity of these trajectories. Additionally, they introduce the Motion Boundary Histogram (MBH) (mentioned in the previous section), which outperforms previous descriptors due to its robustness against camera motion. Moreover, in [25], authors further improved this algorithm to increase its performance on unconstrained, realistic videos by computing homographies between pairs of frames to correct the effects of the camera motion. Nevertheless, due to the nature of the datasets used in this work (static camera), this improvement would not only introduce errors, but also increase the computational time of the algorithm, later on, in a section devoted to this issue, it is explained why this happens.

Dense Trajectories provide a video representation based on densely sampled trajectories and a set of descriptors: HOG for spatial appearance, HOF for first-order motion information and MBH for second-order motion information. Trajectories capture local information of the video. By densely sampling it is ensured a good coverage of foreground motion as well as surrounding context information. It is important to notice that the extracted descriptors are not computed on a 3D fashion, as HOG3D or 3D-SIFT. As 2D space domain and 1D time domain show different characteristics, it is more intuitive to treat them differently as well. That is why they propose a trajectory-based approach which implicitly models temporal dimension.

The algorithm is computed over a multi-scale pyramid of at most 8 layers (depending on video resolution), with a scale stride of $\sqrt{2}$. The first step of the algorithm consists of the dense sample of feature points over the first frame on a grid space by $W = 5$ pixels. This procedure ensures feature

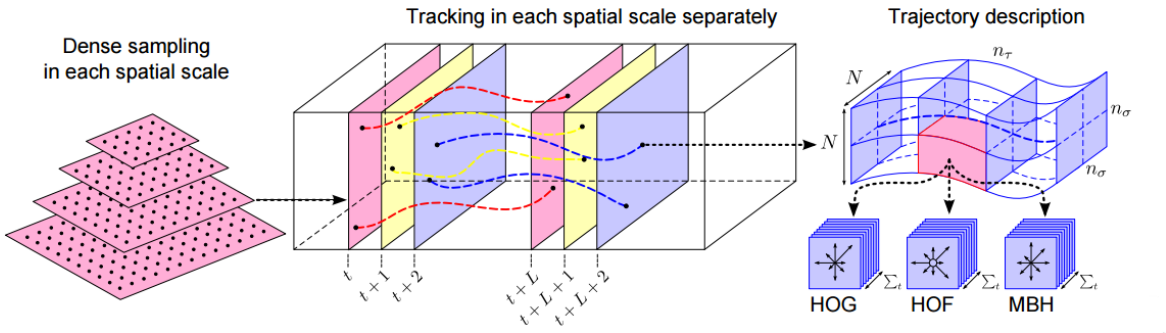


Figure 3.1: Visualization of the procedure followed by Dense Trajectory's algorithm.

points cover all spatial positions and scales. Points over homogeneous areas are rejected due to tracking unfeasibility. Then, dense optical flow is computed for the current frame with respect to the next one. This allows the tracking of these densely sampled points by adding the optical flow (displacement) to the coordinates of the corresponding point. Then, for the following frame, the points obtained are once again displaced by the corresponding optical flow of that frame. Following this procedure, dense trajectories are built from the initial densely sampled feature points. As the whole process is repeated at several scales, trajectories on each level are tracked, allowing then the encoding of larger motions.

As static trajectories encode no motion information, these are later on filtered on the post-processing step by thresholding the variances of the coordinates x and y of the points of the trajectory. If none of the variances is above a given value, trajectory is rejected for being static. Similarly, trajectories with sudden large displacements are rejected as well by, most likely, being erroneous. The concrete criteria is if the displacement vector among two frames is above the 70% of the trajectory's total length. Moreover, trajectories with variances too high, are filtered as well for being inconsistent.

Trajectories sampled are tracked along $L = 15$ frames. Once a trajectory achieves a length of 15 is considered completed and is filtered as valid or invalid. Feature points are sampled every frame, but rejecting points that already belong to an ongoing trajectory. Around each trajectory, it is constructed a window of $N \times N$ pixels, with $N = 32$, along the $L = 15$ frames of the trajectory. This creates a space-time volume around the trajectory, which is then divided into $n_\sigma \times n_\sigma \times n_t$ cells, where $n_\sigma = 2$ and $n_t = 3$. For each sub-division of the spatio-temporal grid, descriptors are computed (HOG, HOF and MBH), then the final descriptor is the result of the concatenation of the histograms of each cell. Fig.3.1 shows a graphical representation of this procedure. First, feature points are densely sampled over the multi-scale pyramid, then, trajectories are constructed by using dense optical flow, and finally, descriptors are computed around the trajectory, on a spatio-temporal volume divided into sub-cells.

For the classic HOG descriptor, orientations are quantified into 8 bins with orientations and magnitudes used as weights, therefore, the concatenation of the descriptor for each cell yields a final descriptor size of $n_{bins} \times n_\sigma \times n_\sigma \times n_t = 96$. For HOF, an additional zero bin is added (leading to a total of 9 bins), for those pixels whose optical flow value is below a threshold, so the final descriptor size in this case is 108. Lately, MBH descriptor is actually the concatenation of two descriptors, MBHx and MBHy, both of which are constructed in the same fashion as HOG descriptor, having each of them a total of 8 orientation bins, so the final size of the MBH descriptor is $96 \times 2 = 192$. As it is mentioned before, HOG descriptor encodes spatial appearance while HOF captures first-order motion information. On the other hand, MBHx

and MBHy correspond to the X and Y derivatives of optical flow, therefore, the second-order motion information they encode is the relative motion of pixels, highlighting regions of the image where the optical flow changes; i.e., motion boundaries.

3.1.1 Improved Dense Trajectories

As aforementioned, this improved algorithm increases the robustness against camera motion by correcting its effects. In order to do so, they densely sample points and key points between a pair of frames, then, using RANSAC matching, they find a homography, which applied on optical flow obtains a new motion field free of camera motion. Furthermore, they filter all the trajectories with small displacements, assuming that are due to camera motion as well. This procedure though, showed bad performance on frames dominated by human motion, as the RANSAC matching algorithm failed to properly estimate the homography if key points extracted from moving humans were used. For this reason, using a human detector, matches inside bounding boxes corresponding to humans are discarded.

However, the dataset used in this work, MSRDailyAct3D, does not contain camera motion. Therefore, trying to estimate camera motion is unnecessarily increasing computational time. Moreover, iDT (improved Dense Trajectories) code does not include the human detector, as it is not owned by the authors. Then, unless bounding boxes for humans are provided, errors will be introduced when estimating the homography, furthermore, even with bounding boxes, it might be possible that small errors are still being introduced. As it would not make sense to apply a human detector to avoid a wrong homography for camera motion estimation when there is no camera motion, instead of iDT, the previous version of the algorithm, Dense Trajectories, has been chosen. Additionally, the filter they apply to suppress trajectories due to camera motion (small displacement trajectories) is incorrectly rejecting valid trajectories, and therefore, losing information.

3.2 Multi-modal Dense Trajectories

Dense trajectories algorithm has been developed to work with RGB data only. Nevertheless, due to the increasing in depth sensors availability, previous works have shown the benefits of combining several data sources [18] [19] [20]. Thus, performance enhancement is expected by exploiting RGB and depth data, and combining the strengths of both sources for dense trajectories.

3.2.1 Trajectories

Recalling the previous section, Dense Trajectories are constructed based on the dense optical flow extracted from the video. Although this methodology can accurately track pixels, shows an important drawback. The correspondence along pixels and real world spatial coordinates directly depends on the distance to the camera, therefore, the same movement, performed at different points of the space will produce different trajectory lengths in pixels. Despite using a multi-scale approach, the trajectory rejecting criteria is based on thresholds measured in pixels. This hinders the classification of trajectories as valid or invalids. Moreover, optical flow can only track motion on X and Y axes, discarding then trajectories in which most of the motion is along Z axis.

Since the availability of depth maps, these problems can be addressed by using scene flow instead of optical flow. Using the public code of [16], scene flow is precomputed for the dataset, to be later on used

in the modified algorithm. Scene flow spatial units are meters instead of pixels, achieving invariance to camera distance. Furthermore, scene flow is a 3-dimensional motion field, therefore, movements along Z axis can be tracked as well. Finally, there is an additional advantage on using scene flow. Depth sensors Kinect V1 and V2 have a maximum range of $4000mm$, all points beyond this distance are assigned a pixel value of 0, therefore, as scene flow is computed using these depth maps, the background is automatically removed (unless it is closer than 4 meters).

Pixel tracking

To be able to track pixels with scene flow, as it is precomputed, it would be necessary to project the displacements back to the image plane again. Nevertheless, to do so, it is necessary to have the exact camera parameters (focal lengths). Although using an approximation of these parameters would provide a valid approximation to the pixel displacement, the next scene flow vector of the trajectory would be incorrectly sampled, increasing the error (the error propagates through the trajectory). In Fig. 3.2, a graphical explanation of this effect is shown. In this figure, green arrows represent the properly projected scene flow, while red arrows represent a projection with error, it can be seen that sampling the wrong scene flow and projecting it wrongly increases the trajectory error at each step. The solution to this, then, is to use the optical flow already computed by the algorithm for pixel tracking, as it is accurate enough, and then for each pixel of the trajectory, sample the scene flow, constructing a 3D trajectory on real world coordinates. This procedure avoids using camera parameters, suppressing a source of error, and is equivalent, as optical flow is the projection of scene flow in the image plane. This gives then 2 trajectories, one in pixels, used for the computation of descriptors, and the other one in meters, used for trajectory validation, as it is invariant to camera position.

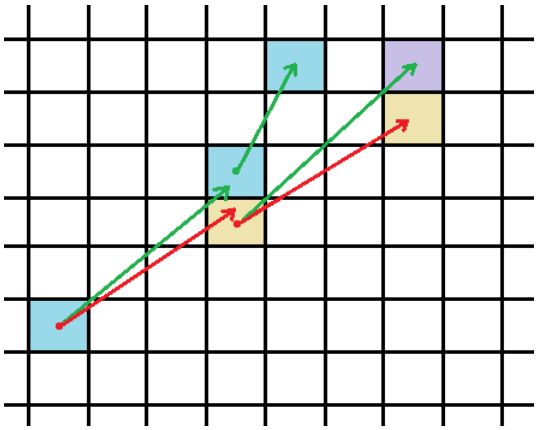


Figure 3.2: Visual representation of the effects of a bad projection of scene flow for pixel tracking. Green arrows show the correct path, while red arrows show how the projection error propagates through the trajectory.

Noise filtering

There is another drawback of using scene flow. In Fig.3.3, it can be seen that around the edges of objects scene flow is very noisy. This is due to the noise found on the depth maps around objects edges. Therefore, it is necessary to filter trajectories that are result of this noise. In order to do so, the following reasoning is applied: noise is random, consequently, trajectories constructed from noisy scene flow will have random

changes of direction. To quantize this, for each pair of consecutive scene flow vectors of the trajectory (total of 15 vectors per trajectory, yielding 14 pairs) the inner product is computed, if the result is below 0, it implies a change of direction of more than 90 degrees. As assumed random noise, the chances of two random vectors having a negative inner product is 50%, then, the probability that a trajectory produced by noise is not filtered with this criteria is of $P = 0.5^{14} < 0.01\%$. Nevertheless, valid trajectories might have this subtle changes on direction as well, and results showed this criteria filtered too many trajectories. That's why experimentally was set an occurrence criteria of at least 4 subtle changes on direction to be filtered as noise, as with this value valid trajectories were not filtered and chances of noisy trajectories being not filtered are still lower than 0.1%.

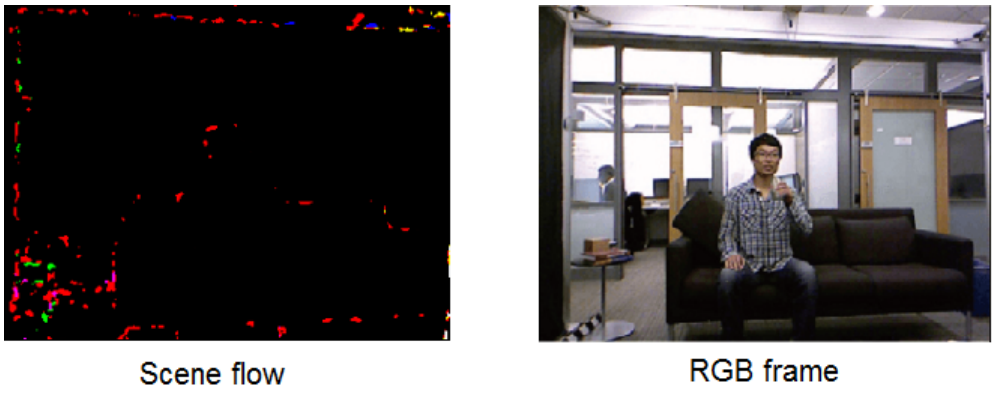


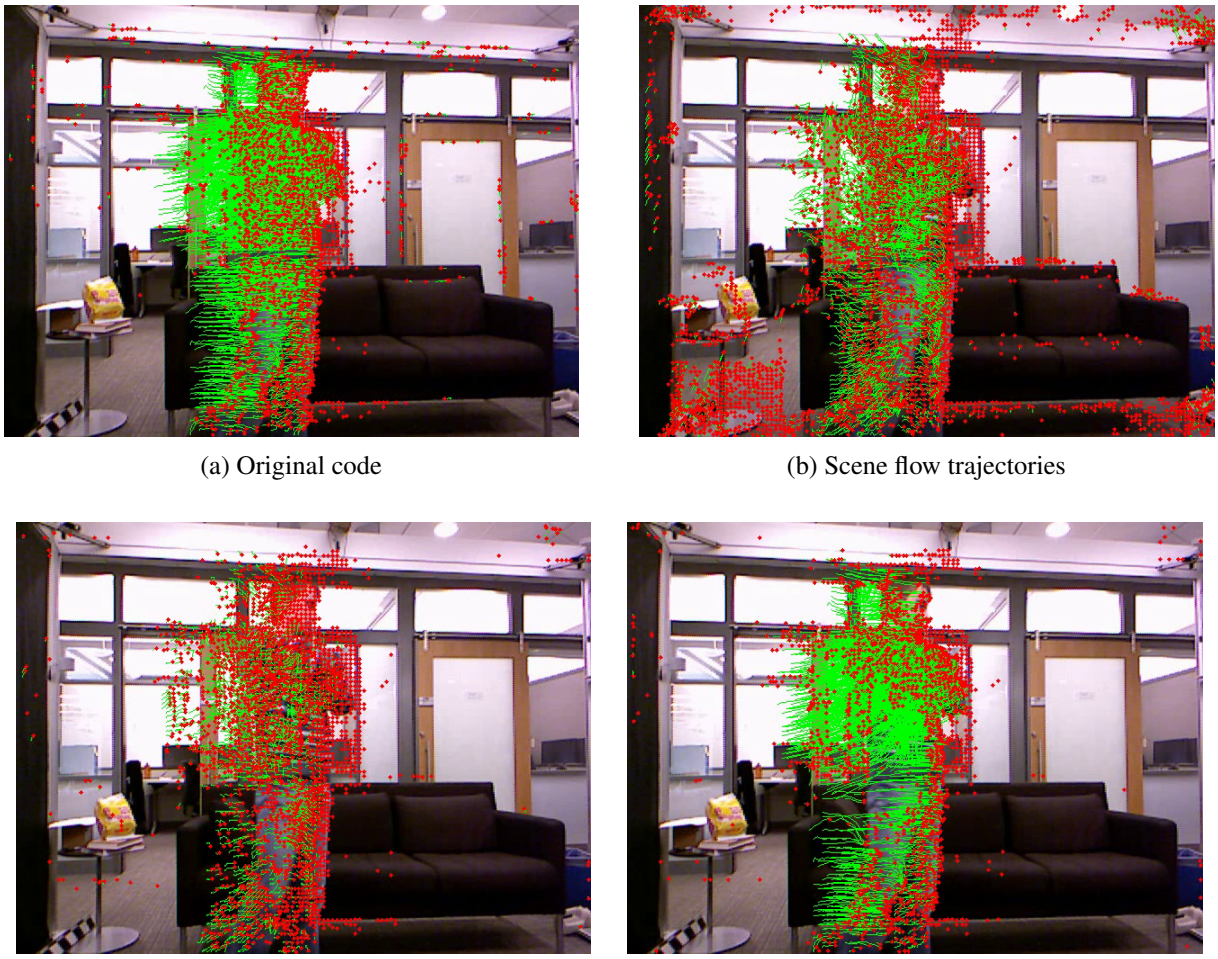
Figure 3.3: Visual representation of a sample of Scene Flow computed in a MSRDailyAct3D dataset video with its corresponding RGB frame.

Final configuration

Therefore, the resulting configuration for multi-modal trajectories is: dense optical flow for pixel tracking, as a precise value is necessary to construct reliable trajectories, around of which descriptors shall be extracted, and scene flow for trajectory filtering, as its real world coordinates achieve invariance to camera distance and also allows the detection of trajectories along Z axis, avoiding then the rejection of trajectories in which most of motion is along this axis. In Fig. 3.4 appears the effect of the different possible configurations for trajectory construction and filtering. In sub-figure (a) appears the original configuration of Dense Trajectories, optical flow for tracking and filtering, on (b) is shown the results of using scene flow for both, pixel tracking and trajectory filtering, it can be seen that trajectory's quality is worse than previous example, plus, there is a lot of noise, in (c) are drawn the effects of the designed noise filter, finally, in (d) are shown the trajectories built with optical flow and filtered with scene flow. Although it is hard to see from still images, the quality of optical flow trajectories is better, and the final configuration keeps less trajectories, as for this concrete example, the subject is closer to the camera, therefore, trajectories in pixels appeared larger, wrongly introducing more motion to the algorithm.

3.2.2 Descriptors

Following the philosophy of computation over spatio-temporal cells and concatenation of the resulting histograms, two new descriptors had been added to construct the final descriptor by taking advantage of the new source of information



(a) Original code

(b) Scene flow trajectories

(c) Scene flow trajectories with noise filter

(d) Optical flow trajectories with scene flow filtering

Figure 3.4: Effect of the different possible configurations for trajectory construction.

The first additional descriptor is histogram of normal vector (HON) [14]. Each normal is represented by two angles θ and ϕ. The surfaces seen by the camera will always have normals facing the camera as well, that is: $0 < \theta < \pi$ and $-\pi/2 < \phi < \pi/2$. To construct the histogram, for simplicity, a 2D histogram is used instead of a sphere tessellation histogram. Then, for each angle, 5 bins are needed, each of them separated $\pi/4$ radians, which leads to a total of 25 bins for each sub-histogram. The final descriptor is the concatenation of these sub-histograms along the 12 spatio-temporal cells, yielding a total descriptor with size of 300. As HON encodes 3-dimensional information about the surfaces of the objects, it is expected to capture the spatial appearance better than HOG, which relies only on 2-dimensional information.

If HON can be understood as an extension of HOG to a third spatial dimension, as both capture spatial information of the objects, the other additional descriptor can be understood as the analogous extension for HOF, the Histogram of Oriented Scene Flow (HOSF). In this case, the vectors are already computed, so it is only necessary to construct the histograms. Nevertheless, scene flow vector can face any direction, that is: $0 < \theta < \pi$ and $-\pi < \phi < \pi$. Therefore, to build the histogram, θ still needs only 5 bins, but ϕ will require 8 bins to fully represent the whole circumference. This leads to a total of number of 40 bins, however, since an additional bin is included in HOF for low-magnitude vectors, then, there will be 41 bins per cell for HOSF. Final descriptor size is then 492. Like HON, it is expected that HOSF can encode motion information better than HOF and will be able to give a more discriminative representation.

3.3 Two-stream Convolutional Neural Network

Proposed by [10], it is an extension of the deep Convolutional Neural Networks, which nowadays is the state-of-the-art on still image representation, to the domain of automatic action and gesture recognition. This new architecture performs the recognition by processing two different streams (spatial and temporal) at the same time, and combining both by late fusion. The first stream captures spatial information from still frames of the video whereas the temporal stream uses as input the dense optical flow extracted from the same video. Both of the streams are implemented as a regular Convolutional Neural Network. This supposes an advantage as the spatial stream network can be pre-trained on larger datasets and fine tuned for the action or gesture recognition application.

The idea behind this proposed architecture is the two-stream hypothesis [85]. The same suggests that human visual cortex is indeed composed of two pathways: the ventral stream (which performs object recognition) and the dorsal stream (which recognizes motion). Nonetheless, authors did not further researched this connection.

3.3.1 Architecture

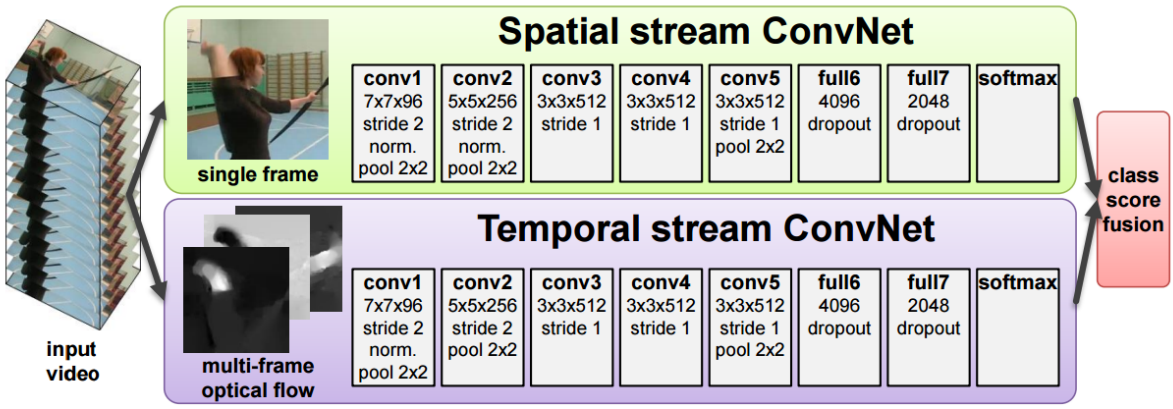


Figure 3.5: Two-stream ConvNet architecture.

As aforementioned, the architecture of the two-stream convolutional neural network is actually a composition of two different deep ConvNets, which its softmax scores are combined by late fusion. In Fig. 3.5, the detailed architecture of the model is depicted.

Spatial stream

The spatial recognition stream ConvNet receives video frames as its input, and then performs action recognition from still images. Static appearance of the video can be discriminative enough by itself, as some actions are strongly associated to some objects (e.g. playing guitar, reading a book, drinking, ...). Additionally, as this architecture is basically an image classification model, it can benefit from the recent advances in large-scale image recognition methods [86], and pre-train the network on a large dataset, to later fine tune its parameters for the concrete domain of action or gesture recognition.

Temporal stream

On the other hand, the temporal recognition stream ConvNet takes as input the stacks of optical flow displacement fields along several video frames. This kind of input explicitly defines the motion of the video, which makes the task easier, as the neural network does not have to learn to implicitly estimate motion. Authors of the method propose different possibilities on how to stack optical flow.

Optical flow stacking. A dense optical flow is a vector displacement field between a pair of frames. Each vector $\mathbf{d}(x, y)$ corresponds to the displacement of the pixel in the position (x, y) to its position in the next frame. This configuration proposes to encode each component of the vector $\mathbf{d}$, d_x and d_y as image channels. Therefore, being w and h the width and the height of the video, for an stack of L frames, the optical flow input volume dimension is $w \times h \times 2L$. Where each point $(x, y) \in \mathbb{R}^{w \times h}$ corresponds to a pixel position, odd channels store the horizontal components of optical flow, while even channels store the vertical one.

Trajectory stacking. Inspired by trajectory-based descriptors. Instead of sampling optical flow at the same location at each frame, flow is sampled along motion trajectories. The first element of the stack would be the same, but in the next one, the value for the position (x, y) is the flow sampled at the position $(x + d_x, y + d_y)$, where $\mathbf{d}$ is the optical flow of the first frame at location (x, y). Following this procedure through the L frames will result in an input volume of trajectories.

Bi-directional optical flow. Optical flow represents the displacements from frame t to frame $t + 1$. It is possible to conceive then an extension to a bi-directional flow by computing an additional set of displacements in the opposite direction that connect a frame t with the frame $t - 1$. This way, for L frames, $L/2$ forward optical flow is stacked for frames t to $t + L/2$, and $L/2$ backwards optical flow for frames t to $t - L/2$. This conception can be constructed using both of previous approaches, optical flow stacking and trajectory stacking.

Mean flow subtraction. For deep learning models, it is beneficial to use zero-centered input, as it allows the model to better exploit the rectification non-linearities. Optical flow can take both positive and negative values, and as movement to one direction is as probable as movement in the opposite direction, it can be naturally centered. Nevertheless, it is probable that between a pair of frames, movement in one direction is dominant. So, for each displacement vector $\mathbf{d}$, the mean value is subtracted.

3.4 Fusion

As a final addition to the methodologies presented, a fusion strategy is tested as well. It is expected to combine the strengths of both approaches, hand-crafted and deep learning models. In order to do so, different possibilities of fusion arise. Authors in [87] propose the use of features learnt through the training of deep learning networks to later on use this filters as descriptors for the Improved Dense Trajectories algorithm. Nevertheless, due to the complexity of the implementation and the computational time required to learn the filters, a more simple fusion strategy has been chosen. Instead, a late fusion is performed. For each of the models and samples there is a confidence value per class, the values of both models are concatenated and used as a new feature vector for classification with an SVM. This configuration should guarantee that the accuracy is at least the minimum of both models, therefore, it is likely an increase on the performance.

Chapter 4

Results

This chapter is structured as follow: first an analysis of the datasets used for the previously explained methods, and a review of how they should be evaluated, then, a detailed explanation of the experiments performed, the results of these experiments along with a discussion and lately a final conclusion on the topic of the document.

4.1 Datasets

In this section shall be described the datasets to be used in this project: MSRDailyAct3D and Montlabano II. Their characteristics shall be discussed and how the sames might behave with the selected algorithms to test. Additionally, for each dataset is detailed the evaluation methodology chosen.

4.1.1 MSRDailyAct3D

This dataset has been provided by Microsoft and it consists of sixteen different actions captured with Kinect device. Each sample is composed of an RGB video and a depth map per frame. The actions presented on the videos are: drink, eat, read book, write on paper, use laptop, play game, call cellphone, use vacuum cleaner, cheer up, sit still, walking, sit down, toss paper, lay down on sofa, stand up and play guitar. Each of the actions is performed twice by 10 different subjects, leading to 20 samples per action, a total of 320 samples. This low number of samples makes the task more challenging as the models will tend to overfit if not treated carefully. In Fig. 4.1 some samples from this dataset are shown. As it can be seen, all the videos have been recorded in the same room, therefore they have the same background and surrounding context.

Evaluation

As aforementioned, MSRDailyAct3D is composed of 16 actions performed by 10 subjects, each repeated twice. The dataset is then balanced in the number of classes and subjects. From this, two alternatives of evaluation arise. The first one is the leave-one-out approach, in such case, models should be trained on 9 subjects and tested on the last one, the process is then repeated 10 times, one for subject, and the results are averaged (this is possible because the dataset is balanced). On the other hand, there is the possibility of training on half the subjects and testing on the other half. The later evaluation metric is preferred because deep learning models take long to train, therefore, repeating the process for each subject is a very time

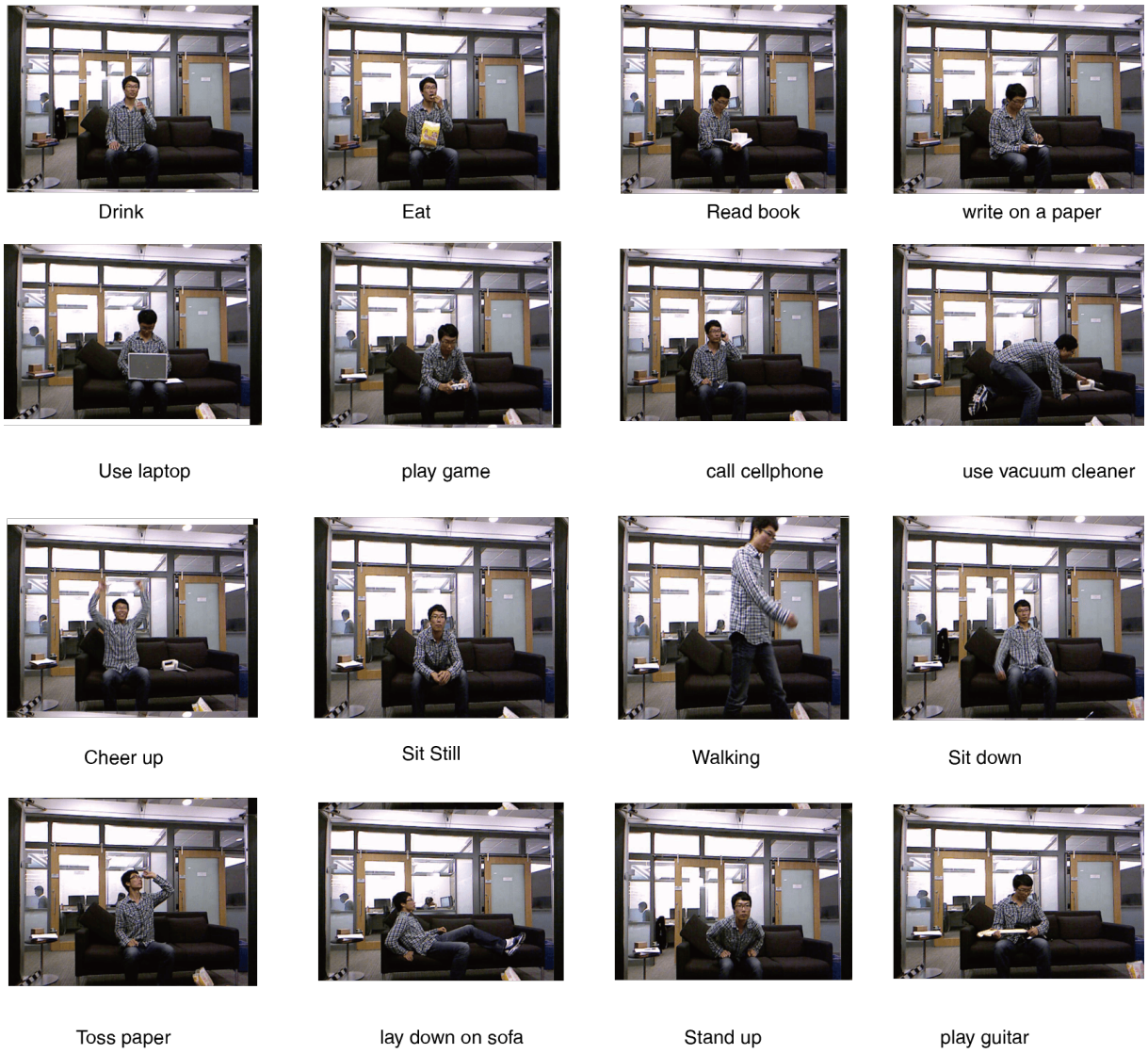


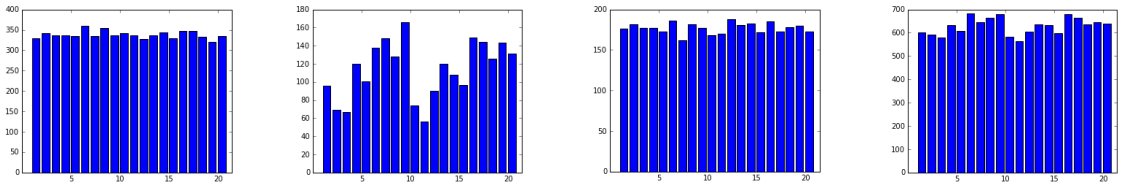
Figure 4.1: Frame samples of each of the actions which compose MSRDailyAct3D dataset.

consuming task. The final partition of the dataset is then, odd subjects (1, 3, 5, ...) for training, and the rest for testing.

4.1.2 Montalbano II

This dataset is composed of 940 sample videos of subjects performing 20 different Italian gestures. Like MSRDailyAct3D, videos had been recorded with Kinect device, therefore, both RGB and depth data are available. Each of the videos though consist of several gestures, as this dataset is used for gesture detection as well. Nevertheless, within the dataset, labels are provided in triplets: label, starting frame and ending frame of the gesture, for each video. Therefore, it is possible to previously split all of the videos into single gesture samples. This leads to a total of 12.575 samples. This dataset is already divided in three subsets, train, validation and test, in Fig. 4.2 is shown the number of samples per gesture, although each class has a different amount of instances, they are similar, therefore, it can be considered a balanced dataset, except for the validation subset. In Fig. 4.3 is shown a frame of a sample of each gesture along with its labels. As can be observed, in this dataset, the background is not always the same, furthermore,

subject's distance to camera changes from sample to sample, as some of the subjects' bodies appear completely while other subjects have their feet outside the image plane. Due to the 4 extra classes, it is expected to be a more difficult task to classify Montalbano II correctly, on the other hand though, the large number of samples should allow the models to generalize better.



(a) Training. (b) Validation. (c) Test. (d) All.

Figure 4.2: Number of samples per class in Montalbano II dataset.

Figure 4.3: Sample frame for each of the gestures of the Montalbano II dataset.

Evaluation

As mentioned above, the Montalbano II dataset is already divided in train, validation and test. The train subset goes from samples 1 to 470, which once split into gestures has a total of 6.761 samples. Validation subset goes from samples 471 to 700, making a total of 2.271 gestures after splitting. Finally, test subset goes from sample 701 to sample 940, which in turn becomes a total of 3.543 different gesture videos. The validation set is used in deep models for fine-tuning and performance tracking, nevertheless, for the hand-crafted methods to be tested, this is not necessary.

4.2 Dense Trajectories

In this section the results of the Dense Trajectories algorithm shall be presented and discussed for both datasets. First of all, it is important to highlight that this algorithm relies on motion detection for a trajectory based descriptor extraction. This might hinder the performance on the classification of low movement actions or gestures.

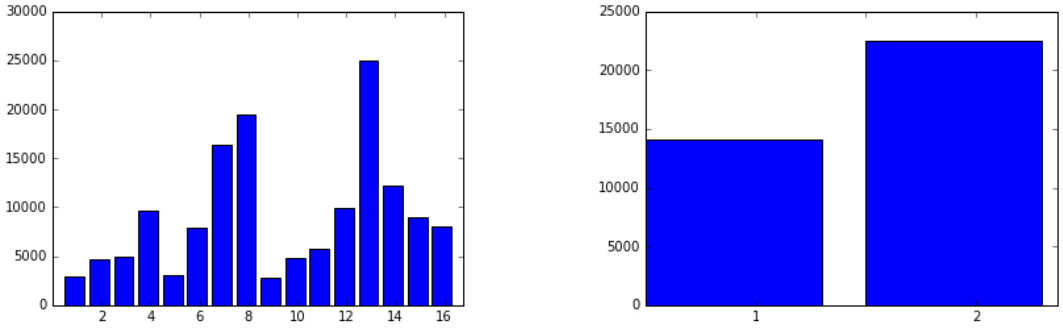
4.2.1 Experiments

For the experimental part of this method, the algorithm is applied to each video of the datasets, where each video corresponds to an action or gesture. This leads to a set of hundreds, thousands or even tens of thousands of trajectories, depending on the quantity of movement of the action, for each video. Similarly to the authors of the method [24], a bag-of-features approach is chosen. First, for each descriptor separately, PCA is applied, keeping 16, 32 and 64 dimensions, due to the spatio-temporal consecutiveness of the cells over which histograms are computed, it is expected that PCA properly eliminates redundant information, then a Gaussian Mixture Model (GMM) is fitted to the transformed data with a total number of components of 32, 64, 128 or 256. After that, each video can be encoded into Fisher Vectors with the GMM obtained for each one of the descriptors (HOG, HOF or MBH). Finally, classification is performed with a multi-class SVM with different kernels and parameters in order to obtain the most suitable configuration. First, the different combinations of PCAs and GMM are tested in order to determine a good encoding that will be used for the rest of the experiments, after that, single descriptors encodings are to be tested, then combinations of them by training several SVMs and aggregating class confidence values.

4.2.2 MSRDailyAct3D

Following the reasoning above, the first step is to make an analysis of the trajectories detected by the algorithm. In Fig. 4.4 are shown a couple of histograms corresponding respectively to the average number of trajectories per action and per example. Regarding actions, it can be observed that action 7, 8 and 13 show a great amount of motion, which correspond to actions 'use vacuum cleaner', 'cheer up' and 'walking'. On the other hand, actions 1, 5 and 9 show the less motion, and correspond, respectively, to 'drink', 'writing on a paper' and 'still'. The latter actions (low motion) might lack of an appropriate video representation, therefore, the algorithm may have trouble classifying it, nevertheless, two of these actions, 'drink' and 'writing on a paper' might be discriminated by the appearance, as both are object dependent. Regarding the examples, as aforementioned, each action is repeated twice per subject, and in many cases (except actions in which is not possible), example 1 is performed sitting on the couch, while example 2 is performed standing. As can be expected, standing examples have a larger amount of trajectories, which is due to two reasons, first, the whole body is moving, so therefore, there is more motion, and second, standing in front of the couch implies less distance to camera, therefore, same movements will have larger trajectories measured in pixels, as the closer to the camera, the larger is the projection of a trajectory (inversely proportional to the depth).

As explained in the previous section, each of the descriptors obtained through the algorithm is treated separately. First, as dimensionality is high, PCA is applied to ease the fitting of Gaussian Mixture Models. Once this is done, for each video and descriptor, a Fisher Vector encoding is computed. Finally, a multi-class SVM is trained. For combinations of descriptors, a multi-class SVM is fitted for each of them



(a) Mean number of trajectories per action.

(b) Mean number of trajectories per example.

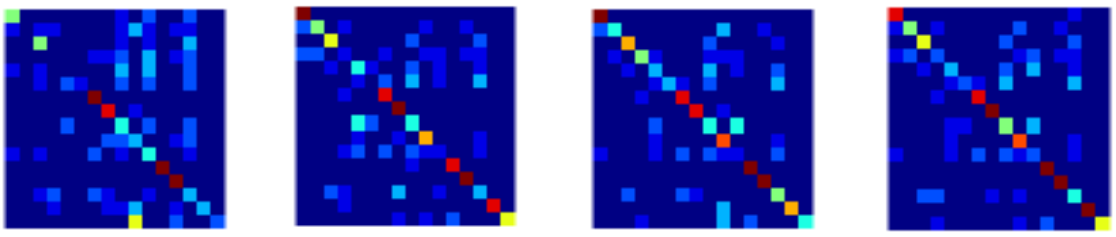
Figure 4.4: Histograms showing the trajectory distribution according to both action and example.

and class confidences are aggregated, then, the final label is the maximum of the resulting scores. After performing several experiments, the selected final configuration for testing is a linear SVM with $C = 100$ for descriptors codified with Gaussian Mixture Models of 32 components after applying a PCA with 32 dimensions.

Table 4.1: Results obtained with Dense Trajectories algorithm per descriptor and combinations of them.

HOG	43.125%	HOG + HOF	58.125%
HOF	58.75%	HOG + MBH	58.75%
MBH	61.875%	HOF + MBH	62.5%
HOG + HOF + MBH		**63.125%**	

In Table 4.1 are shown the results obtained with each of the descriptors and with combinations of them. All of the results were obtained using the same configuration, in order to compare the capability to encode discriminative information of the descriptors. As it can be seen, spatial information (HOG) performance is worse than motion features (HOF and MBH), furthermore, combining a motion descriptor with HOG seems to hinder the accuracy. Nevertheless, the combination of all of them seems to slightly enhance the classification.



(a) HOG

(b) HOF

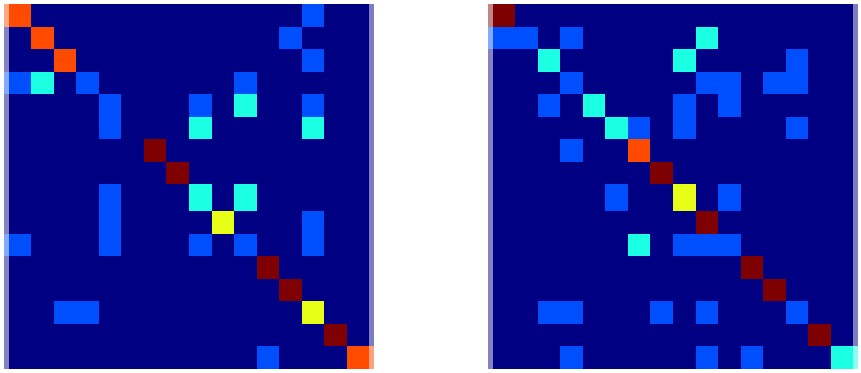
(c) MBH

(d) All

Figure 4.5: Visual representation of confusion matrices for each descriptor separately and for the combination of all of them.

In Fig. 4.5 are shown the confusion matrices obtained for each of the descriptors and for the combination of all of them. Dark blue indicates no population while dark red means a perfect classification. HOG seems to be able to properly discriminate actions 'use vacuum cleaner', 'cheer up', 'lay down on sofa' and 'walking'. It is possible to state then that these actions are the most spatial dependent, which seems coherent, as all of these actions imply a very differentiated posture, while on other actions such

as 'eat', 'drink', 'call cellphone', etc. posture is very similar. On the other hand, HOF shows a good discriminative encoding for the same actions and also for 'drink', 'read book', 'sit still', 'stand up' and 'sit down'. For the first two, it can be concluded that their motion is characteristic of the action, while for the other three, it seems quite clear that are very motion dependent. For MBH we see a similar performance, with an slight improvement in actions 'call cellphone' and 'play guitar'. Finally, the last confusion matrix, where all descriptors are combined show their combined strengths, keeping a good accuracy on the classes where single descriptors performed well.



(a) Example 1 (b) Example 2

Figure 4.6: Visual representation of confusion matrices per example of the action.

As mentioned on the analysis of the dataset, each action is repeated twice by each subject, one sitting down and the other standing up (if the action allows it), exceptions to this are actions 'use vacuum cleaner', 'walking', 'stand up' and 'sit down'. It is expected that this intra-class variation shall have an effect on the performance. In Fig. 4.6 are shown the confusion matrices of actions performed sitting down (example 1) and standing up (example 2). In general it can be noticed that examples 2 are harder to classify. This might be due to the bigger amount of motion of the whole body, without any focus, giving then less discriminative video representations. While actions performed sitting down has motion only in the critical parts of the actions. Furthermore, the bigger amount of trajectories produced by standing up samples biases the data distribution, which directly affects the fitting of the Gaussian Mixture Models and the consequent Fisher Vector encoding, hindering the classification of whole body motion actions. Although the difference on accuracies for each kind of samples is not very large, (65% for 'Example 1' and 61.25% for 'Example 2'), it is still quite significant.

4.2.3 Montalbano II

Following the same procedure as for MSRDailyAct3D, first step is to analyze the amount of motion in the samples. In Fig. 4.7 is depicted the distribution of number of trajectories per gesture. Although it can be seen that is not a homogeneous distribution, is far more balanced than the distribution shown for MSRDailyAct3D. This was expected if we understand gestures as constrained actions where motion is mainly from head, face and arms. Nevertheless, this dataset contains 4 extra classes compared to the previous one, which makes the classification task more complex. On the other hand, the number of samples of this dataset is much larger, which should help the model to better generalize.

The experiments to run are very similar to those performed over MSRDailyAct3D dataset. First of all, each descriptor is treated separately, running first a PCA and fitting a Gaussian Mixture Model, after that,

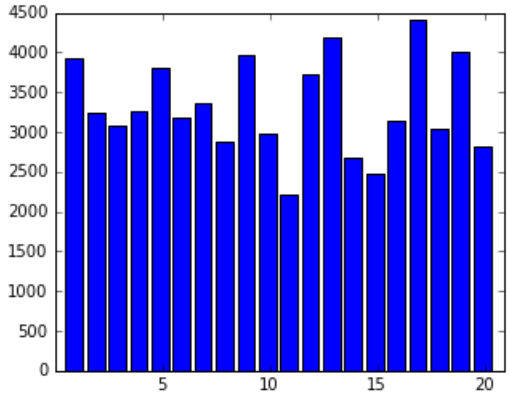


Figure 4.7: Distribution of number of trajectories per gesture.

Fisher Vector encoding is applied to each video per each descriptor. Then, multi-class SVM shall be used for classification, first, single descriptors, then, combinations of them by aggregating confidence values. Due to the higher amount of samples of this dataset, it is possible to operate on higher dimensionality, losing less information. Then, PCA applied over descriptors keeps 64 dimensions and GMM with 32 components is fitted to this data. In this case, as the number of samples prevents overfitting, an RBF kernel SVM is used with $C = 100$.

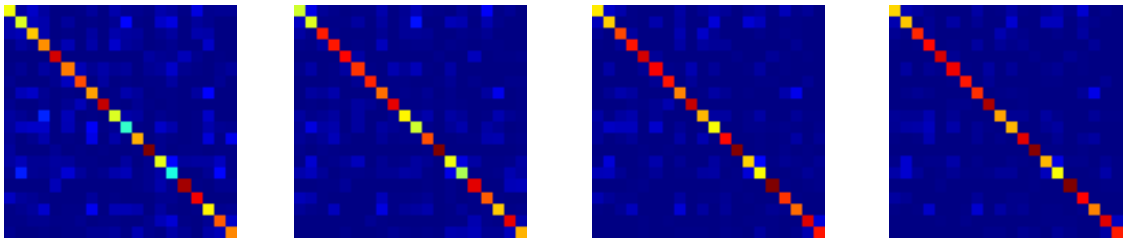
Table 4.2: Results obtained with Dense Trajectories algorithm per descriptor and combinations of them.

HOG	67.3%	HOG + HOF	79.4%
HOF	74.4%	HOG + MBH	81.6%
MBH	79.9%	HOF + MBH	82.0%
HOG + HOF + MBH		**83.5%**	

In Table 4.2 are shown the results obtained for each of the descriptors and for the combinations of them. Once again, MBH descriptor proposed by the authors of Dense Trajectories clearly outperforms the others. Furthermore, the combination of all of the descriptor achieves a great accuracy. It can be seen how, despite having a higher number of classes, the performance over this dataset is quite better than MSRDailyAct3D results. The more relevant difference in this case is the larger number of samples the dataset has (about 40 times more videos), other factors that contributed are the lower intra-class variation, as there is no 'example 1' and 'example 2' for each gesture, which generated an unbalanced trajectory distribution for MSRDailyAct3D, and also the inter-class motion balance (similar number of trajectories per gesture), which allowed the model to properly learn the distribution of each gesture.

There is an additional interesting difference between these results and the ones obtained for the previous dataset. It can be seen that in this case spatial information (HOG descriptor) is quite good by itself, and the difference in performance regarding the other descriptors is not as large as for the MSRDailyAct3D dataset. This means that spatial appearance is a more powerful discriminative feature for gesture recognition compared to action recognition. Furthermore, for the previous dataset, combining motion descriptors with HOG resulted in a worse performance, while for Montalbano II, combining spatial features with motion significantly increases accuracy on every case.

In Fig. 4.8 are shown the confusion matrices for each of the descriptors and lastly for the combination of all of them. As it can be seen, all the descriptors can correctly classify most of gestures, and all of them seem to have trouble identifying the gesture 'noncenepiu'. Although some descriptors fail in



(a) HOG (b) HOF (c) MBH (d) All

Figure 4.8: Visual representation of confusion matrices for each descriptor separately and for the combination of all of them.

identifying other gestures, looks like this particular gesture is a common fail for all of them, and therefore, for their combination. The reason behind it is that 'noncenepiu' gesture consists of a very concrete hand posture with a fast wrist twist that completely changes the 2D appearance of the hand very quickly, which possibly hinders the spatial recognition, while, on the other hand, twisting movements have motion in every direction, therefore, its histogram will not have enough discriminative power to encode such motion.

4.3 Multi-modal Dense Trajectories

In this section shall be presented the results of the modification of the Dense Trajectories algorithm. For logical reasons, such results shall be compared with the original algorithm along with a discussion on their main differences. The development of this section follows the same structure as the previous one.

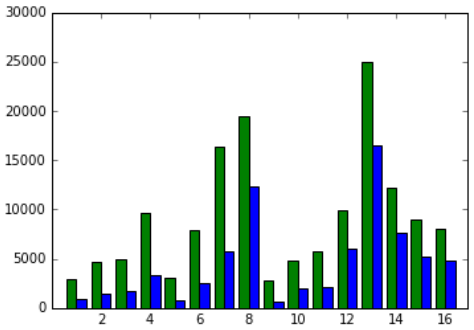
4.3.1 Experiments

In order to be able to properly compare the effects of the modifications on the performance of the algorithm, the experiments to be run shall be the same as for the original code. The main changes in the method are the trajectory validation and the addition of new descriptors. Then, it is possible to compare the effects of each change separately. First, using the original descriptors over the new set of trajectories, and later, adding the new descriptors, it will be possible to know if they are providing with a more discriminant encoding.

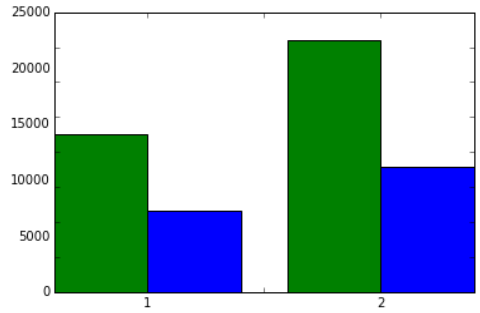
4.3.2 MSRDailyAct3D

Like the original code of Dense Trajectories, first it is analyzed the trajectories found by the algorithm. Recalling the modifications, trajectories are now invariant to camera distance, therefore, it is expected less trajectories for those actions in which the subject was too close to the camera, also, the noise filter implemented and the background removal effect explained in the methodologies section will suppress many invalid trajectories. In Fig. 4.9 is shown the distribution of trajectories for each action and example type for both, the original algorithm and the modification proposed. Green bars represent the original code output and blue bars the results of the modified algorithm. As expected, the number of trajectories is lower for every action, although not always the same proportion, therefore, the data distribution has changed. This can be seen more clearly at the histogram (b), which mainly separates actions performed sitting down (far from camera) and those performed standing up (closer to the camera). As it can be seen, the number of trajectories for 'examples 2' is still bigger than the others, as includes whole body motion,

but the number of trajectories for standing up actions has decreased more than for sitting down actions. As stated before, this changes the data distribution and is expected to provide better video representations by focusing on the most relevant parts.



(a) Mean number of trajectories per action.



(b) Mean number of trajectories per example.

Figure 4.9: Histograms showing the trajectory distribution according to both action and example for the original algorithm (green) and the modification (blue).

In order to be able to properly test the quality of the trajectories obtained with the modified algorithm, the same configuration for the experiments shall be used. In Table 4.3 are shown the results of the experiments performed with the same descriptors like the original code in order to compare whether the distribution change on trajectories improves their quality. As it can be seen, the accuracy has significantly increased in each of the experiments, which implies a higher quality trajectories. In order to confirm this, the experiments performed on the original version of the algorithm were repeated with a random filtering to achieve a similar amount of trajectories, this slightly hindered the performance, although in most of the cases it did not change. Therefore, the new trajectory filter based on scene flow has proven to outperform optical flow based filtering. Furthermore, the removal of invalid trajectories greatly reduced the size of the data to process, which supposes an advantage from the point of view of computational cost, a major limitation nowadays.

Table 4.3: Results obtained with the Multi-modal Dense Trajectory for the original descriptors.

HOG	45.625%	HOG + HOF	62.5%
HOF	60.625%	HOG + MBH	66.25%
MBH	69.375%	**HOF + MBH**	**70%**
HOG + HOF + MBH		65.625%	

Additionally, the Multi-modal dense trajectories included two new descriptors, Histogram of Oriented Normals (depth based feature) and Histogram of Oriented Scene Flow (multi-modal feature). On Table 4.4 are shown the results of the experiments performed with the new two descriptors. First of all, each descriptor is tested separately, then, combined with themselves and with the previous descriptor that showed better performance (MBH). After this, as HON encodes spatial information and HOSF captures motion information of first order, they both can be understood as an extension of HOG and HOF respectively, therefore, it is tested as well the performance for the substitution of these descriptors for their extended version on the combination of all of them.

Additionally, it is also tested their performance when combining with their 2D homologue instead of substituting. As it can be seen, HOSF performs the worst, furthermore, every combination or substitution

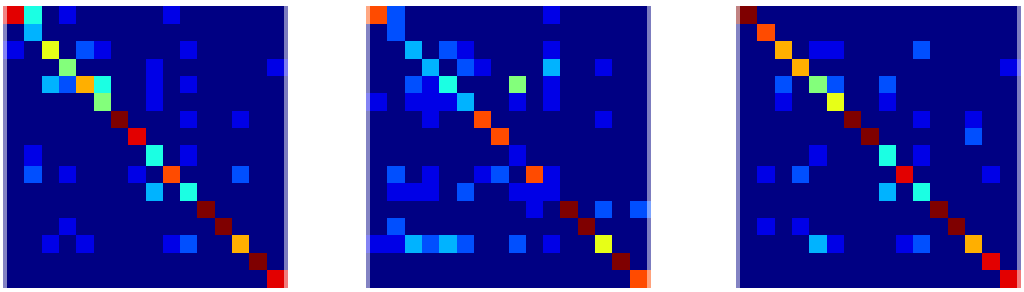
with HOSF shows an accuracy decrease. This might be due to two factors, first, as it was shown on previous sections, scene flow is very noisy around edges, therefore, a descriptor based on it shall have lower discriminative power, the other possible reason for this is the way in which histogram is computed, recalling the difference between a 2D-histogram and a histogram based on the tessellation of a sphere. For simplicity, 2D-histogram was implemented, nevertheless, due to the wide range of possible orientations, the descriptor has a large dimensionality and suffers from sparsity, also, this kind of histograms show distortion for near vertical orientations, which is increased even more by the noise. The combination of both conditions is probably the reason why HOSF shows a bad performance. On the other hand, HON seems to show the best performance of all the descriptors tested.

The advantage of HON respect HOG is that is not based on pixel intensity level (colorful patterns produce a high response on image gradient computation), so it becomes invariant to this, moreover, HON is able to capture information on the 3D structure of objects, being therefore a more discriminative feature. Although this descriptor is also constructed with a 2D histogram, its possible orientations are very limited (half sphere), also, although this kind of histogram present a bad encoding on vectors oriented in a vertical direction, those might appear on huge volumes on scene flow, but not on normal vectors, as most of them will be facing the camera.

Table 4.4: Results of the experiments performed with the new descriptors

HON	72.5%	HON + HOF + MBH	75%
HOSF	58.125%	HOG + HOSF + MBH	66.875%
HON + HOSF	69.375%	HON + HOSF + MBH	75%
HON + MBH	**78.125%**	HOG + HON + HOF + MBH	74.375%
HOSF + MBH	68.125%	HOG + HOF + HOSF + MBH	66.875%
HOG + HON + HOF + HOSF + MBH		73.75%	

Regarding combination or substitution, looks like substitution obtains better results even in spite of the bad performance of HOSF (actually worse than its 2D analogue by itself). After all the tests performed, the descriptor more capable of encoding spatial information is HON, while the best descriptor for motion is MBH, and its combination performs the best, 15 point above the best result obtained with the original Dense Trajectories algorithm.

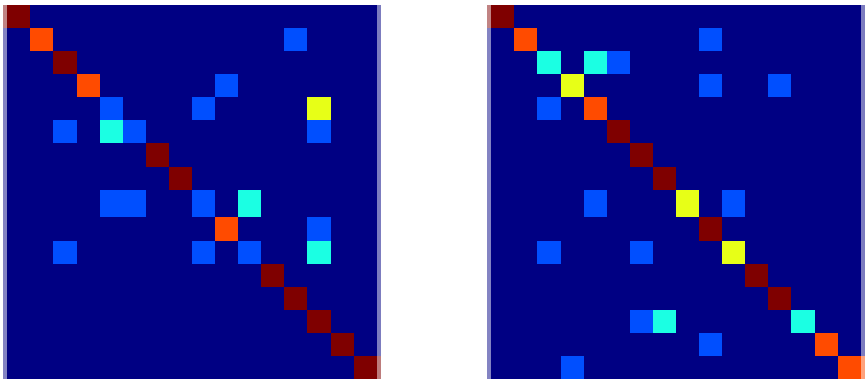


(a) HON (b) HOSF (c) HON + MBH

Figure 4.10: Visual representation of confusion matrices per descriptor and for the best combination.

Fig. 4.10 shows the confusion matrices obtained for each of the new descriptors and for the best combination found in the experiments. Comparing HON to its 2D analogue, HOG, it can be seen a great increase on performance in actions 'drink', 'read a book', 'writing on paper', 'toss paper', 'playing guitar',

'standing up' and 'sitting down'. Except the last two actions, all are object dependent, and its increase on performance is likely due to the capability of HON to better encode 3D structure of these objects. For the two last actions, there is a huge change on body posture which seems easy to recognize using normal vectors, as its distribution along a standing body and a sitting body is very different. Regarding the differences between HOF and its extension, HOSF, the actions in which they perform better are mostly the same, the only significant difference is that HOSF performs better in actions 'laying down on sofa', 'standing up' and 'sitting down', while it performs worse on the rest. These actions in which HOSF outperforms HOF is precisely large motion actions, where the effects of the noise are less relevant, therefore, it might be possible that more accurate scene flow would increase performance on the rest of the actions as well. Finally, for the best configuration found, HON + MBH, looks like most of the classes are accurately classified except actions 'sit still' and 'play game'. Both of these actions have very low motion (specially the first one), and the latter one depends on a small object seen from a point of view which is not the optimal for its recognition (to properly recognize a game controller, the best view is frontal, where the buttons and joysticks can be seen). Furthermore, these two actions were not properly classified by any of the descriptors tested.



(a) Example 1 (b) Example 2

Figure 4.11: Visual representation of confusion matrices per example of the action.

Finally, just as for the original algorithm was shown the difference between actions performed sitting down (examples 1) and those performed standing up (examples 2), in Fig.4.11 are shown the confusion matrices per each of this cases. For the original Dense Trajectories, it was clear that actions performed standing up were more difficult to correctly classify, nevertheless, after the modifications, the situations has reversed, being the examples performed standing up better classified, as for 'examples 1' there is a 76.25% of accuracy while for 'examples 2' the accuracy goes up to 80%. This confirms once again that the scene flow based trajectory filtering provides a more representative data distribution, allowing the models to learn more discriminative features.

4.3.3 Montalbano II

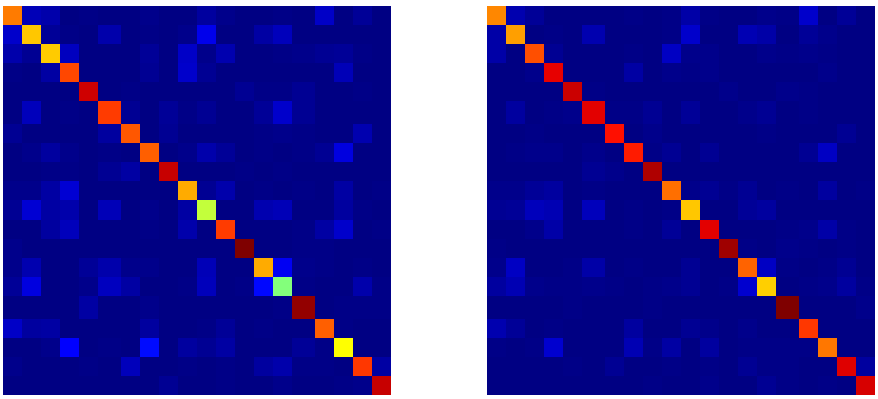
MSRDailyAct3D is a small dataset, therefore, it is possible to perform more experiments and run more algorithms, on the other hand, though, Montalbano II dataset is way larger. That is why, due to the computational cost and time limitations, scene flow could not be included for the experiment on this datasets. This multi-modal approach is reduced then to the addition of the HON descriptor, being a future task the computation and application of scene flow to the algorithm (as for MSRDailyAct3D). Despite not

being able to test the complete modification of Dense Trajectories, this experiments shall prove that a multi-modal approach for automatic action and gesture recognition can be beneficial for any dataset in general. Trajectories detected and filtered by the algorithm are then the same as for the original, therefore, an analysis on this is not necessary in this case.

Table 4.5: Results of the HON descriptor by itself and combined with the rest.

HON	77.67%	HON + HOG	79.57%
HON + HOF	82.81%	**HON + HOF + MBH**	**85.66%**
HON + MBH	85.10%	HON + HOG + HOF + MBH	85.66%

On Table 4.5 appear the results obtained with the new descriptor by itself and combined with the rest of them. As it can be seen, once again, the HON descriptor shows a very good performance by itself, and also a better spatial encoding than its 2D homologue, HOG. Also, any combination of this descriptor has an accuracy enhancement respect the same combination using HOG instead of HON. Finally, the best combination found achieves an accuracy of 85.66%, which is an increase of 2.16% with respect the best configuration for the original algorithm. Notice that as the accuracy was already high, such increase on performance is more significant in this level. Although two configurations achieve a precision of 85.66%, it is chosen as best the combination that has the less descriptors, that is, HON + HOF + MBH.



(a) HON (b) Best

Figure 4.12: Visual representation of confusion matrices for HON descriptor and for the best configuration.

In Fig. 4.12 are depicted the confusion matrices for the HON descriptor and for the best combination found. As it can be seen, the HON descriptor performs very well in most of the classes, with exception of the gesture that previous descriptors were not able to capture neither. This gesture is 'noncenepiu', which, as aforementioned, depends on a very concrete hand position. Histograms of oriented vectors are not able to encode such complex forms because they lose relative spatial information among gradients, meaning that for this shape, it is not only relevant where gradients point to, but also how are they distributed across the sub-cell. Also, there is another gesture for which the performance, although not as low as for 'noncenepiu', is still low, that is the 'OK' gesture, which also relies on a complex hand position. On the other hand though, the combination of different descriptors provide of a more discriminative transformation for these gestures, which implies that, although not being spatially recognizable, they have a characteristic combination of appearance and motion that allows a better classification.

4.4 Two-Stream Convolutional Neural Network

In this section the results obtained with the two-stream ConvNet are presented. Deep learning models have a huge amount of parameters to optimize, due to the sparsity of the MSRDailyAct3D, with 320 samples and only half of them for training, and also because of the high temporal complexity and intra-class variance of actions, it is expected that the results for this dataset are far from the state-of-the-art. On the other hand, Montalbano II does not suffer from this sparsity, so it will be able to better generalize to the test set. Furthermore, as gestures can be understood as constrained actions, there is less intra-class variation and a lower time scale (performing an action takes longer), therefore, less temporal complexity to be modeled by the network. That is why it is expected that the two-stream ConvNet performs better on this dataset.

4.4.1 Experiments

For the experiments performed with the two-streams ConvNet, first, both streams are pre-trained on one of the largest RGB dataset for action recognition (UCF101), and later on fine-tuned for the datasets to be tested on. While the spatial-stream is fed with still images, on the other hand, the temporal-stream is fed with stacked optical flow with mean subtraction. Since the original network's input size is 224×224 and the datasets used in this work have a size of 640×480, cropping the input to network's input size would result in a great loss of information, therefore, several scales of RGB and optical flow are used for fine-tuning the model.

4.4.2 MSRDailyAct3D

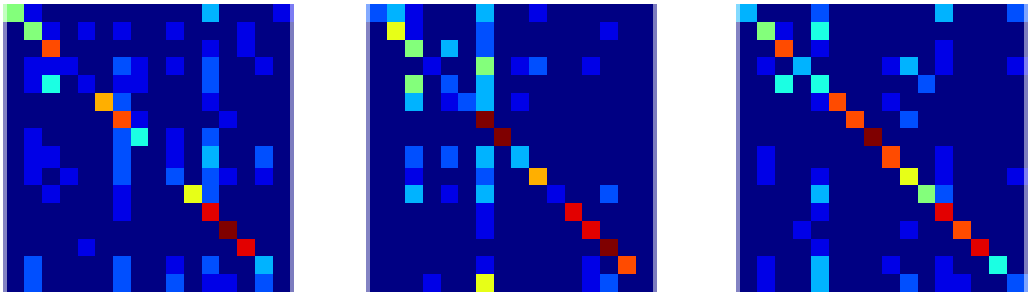
As aforementioned, MSRDailyAct3D has a very low number of samples for a deep learning model, this issue can be solved by pre-training the network on UCF101 dataset, one of the largest datasets for action recognition, to later on fine-tune its parameters on MSRDailyAct3D dataset. Spatial-stream and temporal-stream are trained separately and later on combined to be fine-tuned again by taking single RGB frames of each action along a stack of optical flow from frames around it. In Table 4.6 are shown the results for both of the streams separately and for their combination. In this case, performance of the model is worse than for hand-crafted features, due to the fact that, despite being pre-trained on a large dataset, the number of samples of the MSRDailyAct3D is far from enough for such a complex model.

Table 4.6: Results obtained for each of the streams of the network and for their combination.

RGB	48.75%
Optical Flow	47.50%
RGB + Optical Flow	**62.50%**

Despite the difficulties that both streams experimented for this sparse dataset, it is noticeable the great increase in performance (over 13% more) achieved by the late fusion of both ConvNets, confirming then the assumption that combining spatial appearance and temporal motion features improves the automatic recognition of action. In Fig. 4.13 are depicted the confusion matrices for each of the streams separately and for their combination. From RGB, the model can easily discriminate actions 'read book', 'use laptop', 'use vacuum cleaner', 'play games', 'lay down on sofa', 'walk' and 'playing guitar', while it shows a bad performance for the rest. Most of these actions are object dependent, and as RGB ConvNets are the

state-of-the-art on object recognition, it is an expected behavior, on the other hand, actions 'lay down' and 'walk' are easily recognizable by human posture, which in these cases is very different from the rest of the actions, so spatial appearance is enough. For optical flow, the actions more accurately classified by the model are 'use vacuum cleaner', 'cheer up', 'toss paper', 'lay down on sofa', 'walk', 'playing guitar' and 'stand up'. These actions have a great amount of motion except 'toss paper', but in this case the motion is very characteristic and not repeated in any other action (a flying object motion), so it is reasonable that classification is accurate. Nevertheless, it can be seen that action 'sit down' is mostly misclassified as 'use vacuum cleaner', this might be due to the fact that both actions start with a human standing and then both get lower towards the couch, in the case of 'use vacuum cleaner', subject leans to vacuum the sofa, while in 'sit down', the body experiments a downwards motion due to the nature of the movement, this global motion, similar in both actions, most likely is the reason of the error. Finally, the combination of both streams keep the strengths of the two ConvNets, classifying correctly all the actions that were accurately discriminated by the models separately.



(a) RGB (b) Optical Flow (c) Two-stream

Figure 4.13: Visual representation of confusion matrices per ConvNet and for their combination.

For the hand-crafted algorithms were analyzed the differences among 'example 1' and 'example 2' for this dataset because it directly affected the number of the output trajectories to process. Nevertheless, for this kind of model, the amount of samples and information to work with is the same independently of the example, so it is not necessary to perform such analysis for this part.

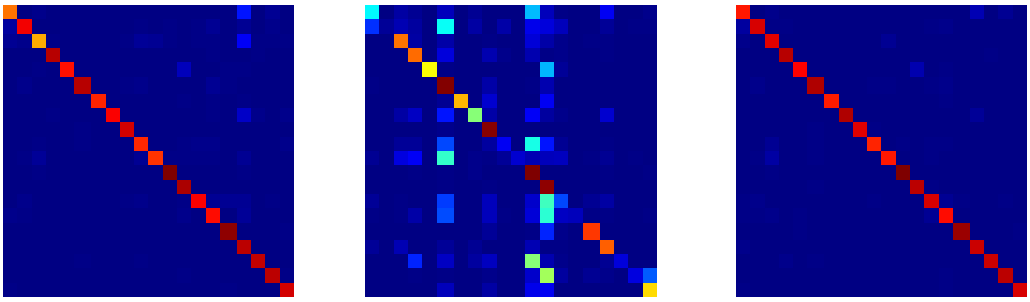
4.4.3 Montalbano II

For Montalbano II, we take advantage of both having a large dataset and also a pre-trained network on UCF101 dataset. Table 4.7 shows the results obtained for this dataset with both spatial and temporal stream plus their combination. As expected, opposite to MSRDailyAct3D, the larger size of Montalbano II allowed the model to generalize much better. Performance for RGB data only is way higher because every gesture is defined by a particular position of the head, body, arms and hands, which makes them easily recognizable from still images, while actions are developed on a larger temporal scale that has to be properly modeled for a correct classification. Optical flow on the other hand might not be discriminative enough for those gestures in which human motion is similar but differ on appearance.

Table 4.7: Results obtained for each of the streams of the network and for their combination.

RGB	94.34%
Optical Flow	53.82%
RGB + Optical Flow	**97.08%**

Just as for MSRDailyAct3D, the late fusion of the spatial and temporal stream increases the performance. In this case, accuracy for spatial stream is very high by itself, therefore, this classification enhancement is even more significant. On Fig. 4.14 are shown the confusion matrices for each stream separately and for their combination. As it can be seen, for RGB and for the combination, classification is almost perfect and there is no much to say. On the other hand, optical flow presents a very heterogeneous behavior, as few gestures are perfectly classified while some others show a very low performance. These gestures that present a good accuracy are 'chevuoi', 'combinato', 'cosatifarei' and 'basta', and its due mainly because their motion is very characteristic, while the rest of gestures have a much less discriminative motion, that hinders their classification. The final increase on performance produced by the late fusion implies that some gestures can be more easily recognizable by their concrete combination of appearance and motion, this gestures are 'vattene' and 'perfetto'.



(a) RGB (b) Optical Flow (c) Two-stream

Figure 4.14: Visual representation of confusion matrices per ConvNet and for their combination.

4.5 Fusion

In this section shall be presented the results obtained by late fusion of the outputs of both, hand-crafted and deep learning, models. Just as the late fusion of RGB and Optical Flow ConvNets supposes an increase on performance by combining strengths of both models, it is also expected that combining these results with the output of the Multi-modal Dense Trajectories algorithm enhances the accuracy as well.

4.5.1 Experiments

As aforementioned, for each sample it is extracted an array of N values, where N is the number of classes, per each model, therefore, a final feature vector size of $2 \times N$. For classification, an SVM is used, tested with different parameters in order to find the optimal configuration. The input confidences are extracted from the best performing configuration of both models.

4.5.2 MSRDailyAct3D

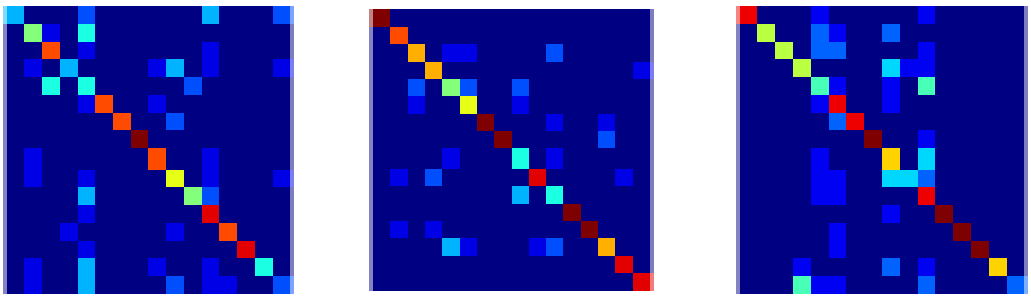
For the MSRDailyAct3D dataset, on the hand-crafted side, the best performing configuration has been chosen, that is, the results of the Multi-modal Dense Trajectories algorithm for the combination of the descriptors HON and MBH. For the deep learning model, the output of the two-stream ConvNet is used. For the fusion of both models, their scores per class and model are concatenated to be used as features for a new classifier. In this case, an SVM with an RBF kernel and $C = 100$. In Table 4.8 are shown the

results of the selected experiments and their final fusion. As it can be seen, in this case, the fusion strategy is worsening the classification accuracy the Multi-modal Dense Trajectories achieved.

Table 4.8: Results of the experiments selected separately and their final fusion.

ConvNet	RGB + Optical Flow	62.5%
Multi-modal Dense Trajectories	**HON + MBH**	**78.125%**
Fusion	ConvNet + MMDT	65%

In Fig. 4.15 are shown the confusion matrices for each model separately and for their late fusion. The comparison between this matrices shows that the combination is hindering the classification of classes in which both models 'disagree', that is, classes in which one of the models performance is high while the other is not, while keeping a good performance on those classes that already are accurately classified by both approaches. As most of the well performing classes are coincident, the accuracy obtained with this experiments is very similar to the obtained with the two-stream ConvNet.



(a) TwoStream ConvNet (b) MMDT (c) Fusion

Figure 4.15: Visual representation of confusion matrices for ConvNet, Multi-modal Dense Trajectories and for their combination.

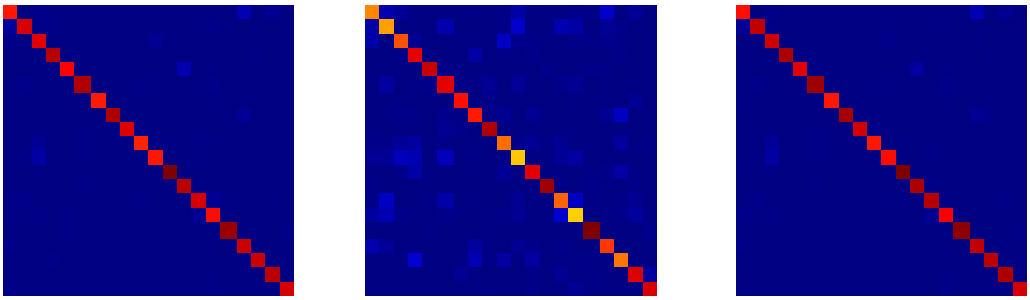
4.5.3 Montalbano II

Analogously to MSRDailyAct3D, for the late fusion of hand-crafted algorithm and deep based model, the best configuration for each algorithm is used. That is, for hand-crafted, Multi-modal Dense Trajectories using descriptors HON, HOF and MBH. And for deep based model, the output of the fusion of RGB and optical flow stream. Table 4.9 shows the accuracy obtained with the best configuration of Multi-modal Dense Trajectories, for two-stream ConvNet and for the final fusion of both models. Differently from MSRDailyAct3D, in this case, both models seem to help themselves to provide an slight boost on performance, which taking into account the very high accuracy the model already had, can be considered quite significant.

Table 4.9: Results of the experiments selected separately and their final fusion.

ConvNet	RGB + Optical Flow	97.08%
Multi-modal Dense Trajectories	HON + HOF + MBH	85.66%
Fusion	**ConvNet + MMDT**	**97.37%**

On Fig. 4.16 appear the confusion matrices for two-stream ConvNet, Multi-modal Dense Trajectories and their final combination. In this case, both models performed very well, and their fusion is even better, as can be seen in the pictures, so there is not much to say about them.



(a) TwoStream ConvNet (b) MMDT (c) Fusion

Figure 4.16: Visual representation of confusion matrices for ConvNet, Multi-modal Dense Trajectories and for their combination.

4.6 Comparison

In this section shall be compared the results of the algorithms tested for MSRDailyAct3D and Montalbano II with previous work performed over these datsets.

4.6.1 MSRDailyAct3D

To be able to properly evaluate the performance of the methodologies selected it is necessary to have a reference of previous works accuracy on the dataset. On Table 4.10 are shown the results of some previous work (upper box) tested on MSRDailyAct3D and the results of the algorithms selected on this project (lower box). Some of this methods require a full skeleton detection, therefore, cannot be directly compared, as the input of the methodologies presented on this work is RGB-D data. Nevertheless, comparing these results will provide a more general perspective on which is the best approach to deal with automatic action and gesture recognition. The last method of the previous work box is a deep based model.

Table 4.10: Results of previous works and the algorithms tested in this work. Methods marked with an asterisk require full skeleton detection.

Methods	**Accuracy**
Dynamic Temporal Warping* [88]	54.0%
Actionlet Ensemble* [89]	85.8%
HON4D* [15]	85.0%
3D Trajectories [90]	72.0%
Long-term Motion Dynamics [91]	**86.9%**

Dense Trajectories	63.125%
Multi-modal Dense Trajectories	**78.125%**
2D CNN RGB	48.75%
2D CNN Optical Flow	47.50%
2D CNN RGB + Optical Flow	62.50%
MMDT + Two-Stream CNN	65%

As it can be seen, our Multi-modal Dense Trajectories outperforms '3D Trajectories', and it achieves an accuracy similar to those hand-crafted methods that require full skeleton detection. On the other hand, our deep based approach for this dataset has a much lower accuracy then the deep learning model found on literature (Long-term Motion Dynamics), although this technique utilizes a neural network

pre-trained on many large datasets to estimate 3D motion, which is then used for classification, while our network was fed directly with raw RGB and optical flow, and pre-trained only with UCF101 dataset. It is likely then, that a more exhaustive pre-training of the network, and maybe applying data augmentation to MSRDailyAct3D, would increase the performance of the model.

4.6.2 Montalbano II

As explained before, the Montalbano II dataset has been designed for automatic gesture detection and recognition, therefore, the work found on the literature do not give classification accuracy, but another metric proposed for this dataset by the ChaLearn LAP 2014 Challenge, that is, the Jaccard index, defined as follows:

$$J_{s,n} = \frac{|A_{s,n} \cap B_{s,n}|}{|A_{s,n} \cup B_{s,n}|} \tag{4.1}$$

Where s is the sequence, n is the gesture category, $A_{s,n}$ represent the ground truth and $B_{s,n}$ the prediction for a given sequence and gesture. The final results is the averaging of this quotient for each sample and categories. As in our case detection is performed as a pre-processing step with the ground truth, the results cannot be directly compared. Nevertheless, if these algorithms had a perfect detection, Jaccard index would be equivalent to accuracy, but as perfect detection is unlikely, its classification accuracy would always be higher than this metric. On Table 4.11 are shown the results of previous works on this dataset (upper box), and as stated before, by its Jaccard index, and also the results obtained during the experiments of this project (lower box) in the form of accuracy, as only classification is assessed.

Table 4.11: Comparison among results of previous works and the results obtained in our experiments. Methods marked with an asterisk require full skeleton detection. Previous work results are given as Jaccard Index as it includes detection.

Methods	**Jaccard Index / Accuracy**
MRF, KNN, PCA, HoG* [92]	0.827
AdaBoost, HoG* [93]	0.834
Multi-Scale DNN [94]	0.836
Multi-Scale DNN* [94]	0.870
Temp Conv + LSTM [95]	**0.906**

Dense Trajectories	83.5%
Multi-modal Dense Trajectories	85.66%
2D CNN RGB	94.34%
2D CNN Optical Flow	53.82%
2D CNN RGB + Optical Flow	97.08%
MMDT + Two-Stream CNN	**97.37%**

As it can be seen, the accuracy of the hand-crafted methodologies is similar to the Jaccard index of the previous works, but as explained before, as just classification task is performed by our experiments, the accuracy should be higher than this index. Taking this into account, it can be stated that performance is quite similar, being the best approach the deep learning models. In the case of a large dataset as this one, it is expected that such models outperform hand-crafted algorithms. Nevertheless, the accuracy of the deep model based on motion features (2D CNN Optical Flow) is quite low, this is likely due to the

similar motion along gestures (mainly arm motion), this shows that for this dataset, appearance is more discriminative than motion. It is, though, the combination of the hand-crafted approach and deep learning model that achieve an almost perfect accuracy.

4.7 Conclusion and Future Work

Automatic action and gesture recognition has growth on interest over the few years, its main characteristic is what makes it a very challenging research topic. This key aspect is the temporal evolution of actions and gestures.

Reviewing the literature around the topic shows an ongoing competition among the authors over finding the best approach to the new dimensionality. Differently from other domains, where deep learning based models have outperformed every hand-crafted feature, in action and gesture recognition there are methods from both sides (hand-crafted and deep) at the state-of-the-art performance level. Each proposed approach deals with time in its own way, for hand-crafted features, authors developed mainly extensions of 2D descriptors to space time descriptors by whether treating time as an additional spatial dimension or aggregating local features through video frames, another approach is the computation of motion features through video frames that are able to encode motion information. For deep learning models, the approaches proposed by the community can be classified into four non-mutually exclusive categories according to how to deal with time dimension: 2D CNNs that perform recognition on single frames and averages results per video, CNNs fed with motion features, 3D CNNs that compute convolutions with 3D filters, where the third dimension is time and temporal deep networks that model temporal evolution.

In this thesis, one hand-crafted feature and one deep based model are selected to test. Dense trajectories, as hand-crafted technique that operates over RGB data, which is later on modified to combine two sources of information, RGB and depth maps. This multi-modal approach showed a boost on performance, confirming the hypothesis that models can benefit from a combination of multiples data sources. And two-stream ConvNet as deep learning model, which performs classification over still images and stacks of optical flow. Results showed that for sparse datasets, hand-crafted features perform better than deep models, furthermore, the combination of RGB and depth significantly improved the approach. On the other hand, for larger datasets, deep models are able to better exploit their capabilities, achieving almost human-like accuracies. Finally, a simple late fusion strategy is applied to combine both approaches, hand-crafted and deep learning, which resulted on an increase of performance on the Montalbano II dataset. It can be concluded that due to the lack of data for the domain, and due to the necessary increase in complexity of the deep based models, hand-crafted features are able to compete with them, it is likely then, that in the future, with an increase of datasets and computational resources, neural networks, once again, show dominance over hand-crafted approaches.

During the experiments, and after the analysis of the results, it has been seen that the quality of the scene flow (3D motion feature) is still far from perfect, and, if with this quality it was possible to increase the performance, it is expected that a more fine computation of this motion feature would provide even better results. Due to computational time and quality requirements, scene flow could not be applied to Montalbano II dataset. Therefore, a possible path for the development of methodologies would be a more precise computation of scene flow. Another possibility for future research is the definition of new descriptors based on depth or combinations of both, RGB and depth. Furthermore, an interesting line of research is the combination of trajectory based approaches, such as the hand-crafted algorithms tested

in this project, with deep based filters, achieving more discriminative encodings. Additionally, just as a multi-modal hand-crafted approach performed better than its RGB homologue method, it is likely than combining RGB and depth maps within deep learning based models would also show an enhancement on the performance. To do so, it is possible to extend the two-stream ConvNet to a four-stream ConvNet by adding two additional streams, one for normal vectors, extracted form depth maps, and an additional one for scene flow, extracted from both RGB and depth.

Summarizing, automatic action and gesture recognition is a challenging task that requires a proper modeling of the temporal dimension. A domain where both hand-crafted and deep based approaches compete for the best performance. Finally, it has been demonstrated how the combination of RGB data and depth maps can significantly increase the performance of these models, pointing as a future line of research the design of different new ways of combining both data sources.

Bibliography

[1] Paul A Wilson and Barbara Lewandowska-Tomaszczyk. Affective robotics: modelling and testing cultural prototypes. *Cognitive computation*, 6(4):814–840, 2014.

[2] Riyad Al-Shaqi, Monjur Mourshed, and Yacine Rezgui. Progress in ambient assisted systems for independent living by the elderly. *SpringerPlus*, 5(1):624, 2016.

[3] Venet Osmani, Sasitharan Balasubramaniam, and Dmitri Botvich. Human activity recognition in pervasive health-care: Supporting efficient remote collaboration. *Journal of network and computer applications*, 31(4):628–655, 2008.

[4] Somayeh Danafar and Niloofar Gheissari. Action recognition for surveillance applications using optic flow and svm. In *Asian Conference on Computer Vision*, pages 457–466. Springer, 2007.

[5] Zahid Halim and Ghulam Abbas. A kinect-based sign language hand gesture recognition system for hearing-and speech-impaired: a pilot study of pakistani sign language. *Assistive Technology*, 27(1):34–43, 2015.

[6] Antonio Tejero-de Pablos, Yuta Nakashima, Tomokazu Sato, and Naokazu Yokoya. Human action recognition-based video summarization for rgb-d personal sports video. In *Multimedia and Expo (ICME), 2016 IEEE International Conference on*, pages 1–6. IEEE, 2016.

[7] Nijun Li, Xu Cheng, Suofei Zhang, and Zhenyang Wu. Realistic human action recognition by fast hog3d and self-organization feature map. *Machine vision and applications*, 25(7):1793–1812, 2014.

[8] Sultan Almotairi and Eraldo Ribeiro. Human action recognition using temporal sequence alignment. In *Computational Science and Computational Intelligence (CSCI), 2014 International Conference on*, volume 1, pages 125–130. IEEE, 2014.

[9] Yasuo Kuniyoshi, Hirochika Inoue, and Masayuki Inaba. Design and implementation of a system that generates assembly programs from visual recognition of human action sequences. In *Intelligent Robots and Systems' 90.'Towards a New Frontier of Applications', Proceedings. IROS'90. IEEE International Workshop on*, pages 567–574. IEEE, 1990.

[10] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[11] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz. Hand gesture recognition with 3d convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–7, 2015.

[12] Jim Giles. Inside the race to hack the kinect, 2010.

[13] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[14] Shuai Tang, Xiaoyu Wang, Xutao Lv, Tony X Han, James Keller, Zhihai He, Marjorie Skubic, and Shihong Lao. Histogram of oriented normal vectors for object recognition with a depth sensor. In *Asian conference on computer vision*, pages 525–538. Springer, 2012.

[15] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723, 2013.

[16] Mariano Jaimez, Mohamed Souiai, Javier Gonzalez-Jimenez, and Daniel Cremers. A primal-dual framework for real-time dense rgb-d scene flow. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 98–104. IEEE, 2015.

[17] Pichao Wang, Wanqing Li, Zhimin Gao, Yuyao Zhang, Chang Tang, and Philip Ogunbona. Scene flow to action map: A new representation for rgb-d based action recognition with convolutional neural networks. *arXiv preprint arXiv:1702.08652*, 2017.

[18] Andreas Eitel, Jost Tobias Springenberg, Luciano Spinello, Martin Riedmiller, and Wolfram Burgard. Multimodal deep learning for robust rgb-d object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 681–687. IEEE, 2015.

[19] Zhi Liu, Chenyang Zhang, and Yingli Tian. 3d-based deep convolutional neural network for action recognition with depth sequences. *Image and Vision Computing*, 55:93–100, 2016.

[20] Di Wu, Lionel Pigou, Pieter-Jan Kindermans, Nam Do-Hoang Le, Ling Shao, Joni Dambre, and Jean-Marc Odobez. Deep dynamic neural networks for multimodal gesture segmentation and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 38(8):1583–1597, 2016.

[21] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360. ACM, 2007.

[22] Rizwan Chaudhry, Avinash Ravichandran, Gregory Hager, and René Vidal. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *computer vision and pattern recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1932–1939. IEEE, 2009.

[23] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International journal of computer vision*, 103(1):60–79, 2013.

[24] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169–3176. IEEE, 2011.

[25] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3551–3558, 2013.

[26] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[27] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[28] Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1961–1970, 2016.

[29] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European Conference on Computer Vision*, pages 816–833. Springer, 2016.

[30] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[31] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497, 2015.

[32] Christoph Feichtenhofer, Axel Pinz, and Richard Wildes. Spatiotemporal residual networks for video action recognition. In *Advances in Neural Information Processing Systems*, pages 3468–3476, 2016.

[33] Suriya Singh, Chetan Arora, and CV Jawahar. First person action recognition using deep learned descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2620–2628, 2016.

[34] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4305–4314, 2015.

[35] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC 2008-19th British Machine Vision Conference*, pages 275–1. British Machine Vision Association, 2008.

[36] Kai Guo, Prakash Ishwar, and Janusz Konrad. Action recognition using sparse representation on covariance manifolds of optical flow. In *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*, pages 188–195. IEEE, 2010.

[37] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 722–729. IEEE, 1999.

[38] Arjun Jain, Jonathan Tompson, Mykhaylo Andriluka, Graham W Taylor, and Christoph Bregler. Learning human pose estimation features with convolutional networks. *arXiv preprint arXiv:1312.7302*, 2013.

[39] Chaoyu Liang, Yonghong Song, and Yuanlin Zhang. Hand gesture recognition using view projection from point cloud. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 4413–4417. IEEE, 2016.

[40] Pichao Wang, Wanqing Li, Song Liu, Zhimin Gao, Chang Tang, and Philip Ogunbona. Large-scale isolated gesture recognition using convolutional neural networks. *arXiv preprint arXiv:1701.01814*, 2017.

[41] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[42] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016.

[43] Shengxin Zha, Florian Luisier, Walter Andrews, Nitish Srivastava, and Ruslan Salakhutdinov. Exploiting image-trained cnn architectures for unconstrained video classification. *arXiv preprint arXiv:1503.04144*, 2015.

[44] Zuxuan Wu, Yanwei Fu, Yu-Gang Jiang, and Leonid Sigal. Harnessing object and scene semantics for large-scale video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3112–3121, 2016.

[45] Hakan Bilen, Basura Fernando, Efstratios Gavves, Andrea Vedaldi, and Stephen Gould. Dynamic image networks for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3034–3042, 2016.

[46] Basura Fernando, Efstratios Gavves, José Oramas, Amir Ghodrati, and Tinne Tuytelaars. Rank pooling for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):773–787, 2017.

[47] Hossein Rahmani and Ajmal Mian. 3d action recognition from novel viewpoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1506–1515, 2016.

[48] Bingbing Ni, Xiaokang Yang, and Shenghua Gao. Progressively parsing interactional objects for fine grained action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1020–1028, 2016.

[49] Denis Fortun, Patrick Bouthemy, and Charles Kervrann. Optical flow modeling and computation: a survey. *Computer Vision and Image Understanding*, 134:1–21, 2015.

[50] Eunbyung Park, Xufeng Han, Tamara L Berg, and Alexander C Berg. Combining multiple sources of knowledge in deep cnns for action recognition. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–8. IEEE, 2016.

[51] Bowen Zhang, Limin Wang, Zhe Wang, Yu Qiao, and Hanli Wang. Real-time action recognition with enhanced motion vector cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2718–2726, 2016.

[52] Georgia Gkioxari and Jitendra Malik. Finding action tubes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 759–768, 2015.

[53] Yang Wang and Minh Hoai. Improving human action recognition by non-action classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2698–2707, 2016.

[54] Wangjiang Zhu, Jie Hu, Gang Sun, Xudong Cao, and Yu Qiao. A key volume mining deep framework for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1991–1999, 2016.

[55] Guilhem Chéron, Ivan Laptev, and Cordelia Schmid. P-cnn: Pose-based cnn features for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3218–3226, 2015.

[56] Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Learning to track for spatio-temporal action localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3164–3172, 2015.

[57] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[58] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer, 2014.

[59] Xiaojiang Peng and Cordelia Schmid. Multi-region two-stream r-cnn for action detection. In *European Conference on Computer Vision*, pages 744–759. Springer, 2016.

[60] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[61] Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *arXiv preprint arXiv:1604.04494*, 2016.

[62] Pichao Wang, Wanqing Li, Zhimin Gao, Jing Zhang, Chang Tang, and Philip O Ogunbona. Action recognition from depth maps using deep convolutional neural networks. *IEEE Transactions on Human-Machine Systems*, 46(4):498–509, 2016.

[63] Xiaojiang Peng and Cordelia Schmid. Encoding feature maps of cnns for action recognition. 2015.

[64] Hossein Rahmani, Ajmal Mian, and Mubarak Shah. Learning a deep model for human action recognition from novel viewpoints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[65] Joe Yue-Hei Ng, Jonghyun Choi, Jan Neumann, and Larry S Davis. Actionflownet: Learning motion representation for action recognition. *arXiv preprint arXiv:1612.03052*, 2016.

[66] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[67] Elman Mansimov, Nitish Srivastava, and Ruslan Salakhutdinov. Initialization strategies of spatio-temporal convolutional neural networks. *arXiv preprint arXiv:1503.07274*, 2015.

[68] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4597–4605, 2015.

[69] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1049–1058, 2016.

[70] Yair Poleg, Ariel Ephrat, Shmuel Peleg, and Chetan Arora. Compact cnn for indexing egocentric videos. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–9. IEEE, 2016.

[71] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *European Conference on Computer Vision*, pages 768–784. Springer, 2016.

[72] Moez Baccouche, Franck Mamalet, Christian Wolf, Christophe Garcia, and Atilla Baskurt. Sequential deep learning for human action recognition. In *International Workshop on Human Behavior Understanding*, pages 29–39. Springer, 2011.

[73] Yuancheng Ye and Yingli Tian. Embedding sequential information into spatiotemporal features for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 37–45, 2016.

[74] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1933–1941, 2016.

[75] Jialin Wu, Gu Wang, Wukui Yang, and Xiangyang Ji. Action recognition with joint attention on multi-level deep features. *arXiv preprint arXiv:1607.02556*, 2016.

[76] Yingwei Li, Weixin Li, Vijay Mahadevan, and Nuno Vasconcelos. Vlad3: Encoding dynamics of deep features for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1951–1960, 2016.

[77] Moez Baccouche, Franck Mamalet, Christian Wolf, Christophe Garcia, and Atilla Baskurt. Action classification in soccer videos with long short-term memory recurrent neural networks. *Artificial Neural Networks–ICANN 2010*, pages 154–159, 2010.

[78] Alexander Grushin, Derek D Monner, James A Reggia, and Ajay Mishra. Robust human action recognition via long short-term memory. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013.

[79] Vivek Veeriah, Naifan Zhuang, and Guo-Jun Qi. Differential recurrent neural networks for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4041–4049, 2015.

[80] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2678–2687, 2016.

[81] Behrooz Mahasseni and Sinisa Todorovic. Regularizing long short term memory with 3d human-skeleton sequences for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3054–3062, 2016.

[82] Jinzhuo Wang, Wenmin Wang, Ronggang Wang, Wen Gao, et al. Deep alternative neural network: Exploring contexts as early as possible for action recognition. In *Advances in Neural Information Processing Systems*, pages 811–819, 2016.

[83] Guy Lev, Gil Sadeh, Benjamin Klein, and Lior Wolf. Rnn fisher vectors for action recognition and image annotation. In *European Conference on Computer Vision*, pages 833–850. Springer, 2016.

[84] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1010–1019, 2016.

[85] Melvyn A Goodale and A David Milner. Separate visual pathways for perception and action. *Trends in neurosciences*, 15(1):20–25, 1992.

[86] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[87] Srikanth Muralidharan, Mehrsan Javan, and Greg Mori. Learning features from improved dense trajectories using deep convolutional networks for human activity recognition.

[88] Meinard Müller and Tido Röder. Motion templates for automatic classification and retrieval of motion capture data. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 137–146. Eurographics Association, 2006.

[89] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1290–1297. IEEE, 2012.

[90] Michal Koperski, Piotr Bilinski, and Francois Bremond. 3d trajectories for action recognition. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 4176–4180. IEEE, 2014.

[91] Zelun Luo, Boya Peng, De-An Huang, Alexandre Alahi, and Li Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. *arXiv preprint arXiv:1701.01821*, 2017.

[92] Ju Yong Chang. Nonparametric gesture labeling from multi-modal data. In *Workshop at the European Conference on Computer Vision*, pages 503–517. Springer, 2014.

[93] Camille Monnier, Stan German, and Andrey Ost. A multi-scale boosted detector for efficient and robust gesture recognition. In *Workshop at the European Conference on Computer Vision*, pages 491–502. Springer, 2014.

[94] Natalia Neverova, Christian Wolf, Graham Taylor, and Florian Nebout. Moddrop: adaptive multi-modal gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(8):1692–1706, 2016.

[95] Lionel Pigou, Aäron Van Den Oord, Sander Dieleman, Mieke Van Herreweghe, and Joni Dambre. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision*, pages 1–10, 2015.