Contributions to Human Action Recognition using Deep Learning: Applications in Egocentric Vision

PhD dissertation by Adrián Núñez Marcos

within the doctoral program Engineering for the Information Society and Sustainable Development

Candidate	Advisors	
Adrián Núñez Marcos	Dr. Gorka Azkune Galparsoro	Dr. Diego López de Ipiña González de Artaza

A mi madre y mi padre por darme la oportunidad de avanzar
hasta aquí
A Sergio y Mikel por estar siempre ahí.

Abstract

Thanks to the exponential increase in the use of Information and Communication Technologies, smart homes that provide services to their inhabitants have become a reality. Among these services, those that secure the health of people are contained within the Ambient Assisted Living paradigm, in which this dissertation is included. In fact, monitoring people in their daily activities may help predicting mental health issues such as dementia, so these kind of applications have become increasingly important. For that, action recognition systems are pivotal, i.e., systems capable of analysing the performance of actions and activities to be able to detect beforehand outliers that could be given by a cognitive decline.

In that sense, ensuring that action recognition systems are able to yield the best possible information is essential. That is why there have been many efforts in the Computer Vision community to tackle the human action recognition task. In fact, it is noticeable the recent contribution of the egocentric action recognition towards predicting actions using wearable devices. In this research line, this dissertation proposes to contribute a method to introduce external knowledge to bias the predictions of action recognition systems towards real-world frequencies in zero-shot settings. Action recognition systems usually output action predictions with some confidence, given only by the input they have seen. Nevertheless, this ignores the frequency with which those actions occur in our daily lives, and that is why introducing a prior probability

distribution with this knowledge is important. More frequently performed actions should have more importance than those that are rare, balancing the distribution. This is implemented in a zero-shot setting, in which actions are decoupled into the motion and the object that is being manipulated. With this approach, motion and objects can be separately learnt by an action recognition system and any combination of both sets can be predicted as a result, offering a much wider range of possibilities requiring fewer annotations. The experiments carried out for action classification with unseen classes have shown promising results when prior probability distributions fit better the users' action distributions.

Resumen

Gracias al crecimiento exponencial en el uso de las Tecnologías de la Información y la Comunicación, las casas inteligentes que proveen de servicios a sus ocupantes se han vuelto una realidad. Entre esos servicios, aquellos que buscan asegurar la salud de las personas se encuentran dentro de los entornos asistenciales inteligentes, en los cuales se incluye esta tesis doctoral. De hecho, la monitorización de las personas en sus actividades del día a día puede ayudar a predecir problemas de salud mental tales como la demencia, así que este tipo de aplicaciones se ha vuelto cada vez más importante. Para ello, los sistemas de reconocimiento de acciones, es decir, sistemas capaces de analizar la ejecución de acciones y actividades, son cruciales para poder detectar de antemano casos atípicos que pueden ser dados por un declive cognitivo.

En ese sentido, es esencial asegurar que los sistemas de reconocimiento de acciones son capaces de ofrecer la información de más calidad. Por esta razón, la comunidad de Visión por Computador ha realizado un gran esfuerzo por mejorar en la tarea del reconocimento de acciones humanas. De hecho, es notable la reciente contribución del reconocimiento de acciones egocéntricas hacia la predicción de acciones usando dispositivos vestibles. En esta línea de investigación, esta tesis doctoral propone contribuir con un método para introducir conocimiento externo para sesgar las predicciones de los sistemas de reconocimiento de acciones hacia su frecuencia del mundo real en el marco del aprendizaje basado en

cero muestras. Los sistemas de reconocimiento de acciones normalmente emiten predicciones de acciones con una cierta confianza, dada únicamente por el dato de entrada que ven. Sin embargo, esto ignora la frecuencia con la cual esas ocurren en nuestras vidas cotidianas, y es por ello que introducir una distribución de probabilidades *a priori* con este conocimiento es importante. Las acciones realizadas con mayor frecuencia deberían tener más importancia que aquellas que son raras, equilibrando la distribución. Con este enfoque, el movimiento y los objetos pueden ser aprendidos de forma separada por un sistema de reconocimiento de acciones y cualquier combinación de ambos conjuntos puede ser predicha como resultado, ofreciendo un abanico de posibilidades más amplio y necesitando menos anotaciones. Los experimentos llevados a cabo para el reconocimiento de acciones con clases nunca antes vistas muestran resultados prometedores cuando las distribuciones de probabilidades *a priori* se ajustan mejor a las distribuciones de acciones reales.

Acknowledgements

Tengo que empezar agradeciendo de todo corazón la ayuda y el apoyo de Nacho, que ha estado desde el principio de esta tesis doctoral. Siento que sin ti esta tesis no hubiese sido posible, así que quiero dedicarte estas primeras lineas.

A mi madre, mi padre, Sergio y Mikel por apoyarme siempre incondicionalmente y por ayudarme a llegar hasta aquí. A pesar de que me han visto en las buenas y en las malas, nunca han dudado de mí y siempre han estado orgullosos de lo que hago.

A mis directores de tesis, especialmente a Gorka, por ayudarme desde el principio y enseñarme, por escucharme aun en mis bajos momentos y siempre siendo muy considerados. Por tener paciencia conmigo y no desistir, porque tenerles ha sido una fortuna para mí.

A todo el grupo de MORElab (Aitor, Gorka, Unai U., Mikel, Diego C., Diego I., Pablo, David, Koldo, Josu, Enrique, Iván, Rubén M., Unai L., etc.) con el que he compartido todos estos años o parte de ellos al menos, por acogerme y hacerme sentir uno más. A todas las compañeras y compañeros con los que empecé o compartí este viaje tan largo y con las cuales he compartido muchas experiencias, llantos y alegrías: Aritz, Oihane, Ane, Héctor, Javier, Jesús, Irene, Rubén S., Unai Z, Ornela y tantas otras personas de la universidad. Así como al apoyo de otras muchas amistades que siempre se han preocupado por mí y me han alentado: Xabier, Naiara, Asier M., Mikel G., Cris y un largo etcétera. También estoy agradecido a toda la gente que ha compartido el gimnasio todos

estos años conmigo, en la que siempre he encontrado el cariño y el apoyo.

I would also like to thank the people at the Centre for Machine Perception, with whom I shared three months of the journey. They greatly inspired me to work on my PhD with passion.

Finalmente, al Gobierno Vasco por la concesión de la beca doctoral con la que he podido trabajar en mi tesis doctoral todos estos años y a DEUSTEK por el apoyo financiero para poder publicar y realizar mi estancia doctoral.

Eskerrik asko
Adrián Núñez Marcos

Table of Contents

List of Figures

List of Tables

Acronyms

AAL Ambient Assisted Living. 2, 3, 45, 49, 121, 122, 154

ADL Activity of Daily Living. 2, 4, 46, 58, 59, 122, 125, 135, 140, 143, 144

AP Average Precision. 93, 94

AR Average Recall. 93, 94

BB bounding box. vii, 39, 47, 48, 88, 89, 90, 93, 94, 95, 97, 98, 102, 111, 112, 113, 115, 145

BOM Bag of Objects Matrix. vii, 73, 99, 101, 100, 102, 105

CNN Convolutional Neural Network. vii, 18, 19, 20, 21, 24, 27, 29, 31, 33, 34, 36, 39, 40, 53, 54, 79, 80, 82, 84, 87, 88, 101, 102, 106, 114, 113, 114, 118, 124, 131, 132, 145

ConvLSTM Convolutional Long-Short Term Memory. viii, 27, 31, 114, 118, 119, 124, 126

DCNN Deconvolutional Neural Network. 35

DL deep learning. 4, 6, 7, 8, 25, 53, 54, 55, 70, 88, 113, 145, 147, 149

DT Dense Trajectories. 84

DTW Dynamic Time Warping. 20

EAR egocentric action recognition. v, 2, 4, 6, 7, 8, 9, 11, 13, 19, 23, 35, 36, 38, 39, 40, 41, 42, 43, 51, 73, 84, 88, 118, 120, 121, 122, 129, 144, 149, 151, 154

FC Fully-Connected. v, vii, 18, 20, 25, 27, 31, 42, 54, 56, 86, 95, 114, 124

FCN Fully Convolutional Network. 19

FN False Negative. 61, 77, 78, 81, 93, 94, 102, 105, 115, 116

FP False Positive. 61, 77, 78, 81, 93, 94, 102, 115, 116

FPS Frames Per Second. 66, 74

GCN Graph Convolutional Network. 42, 152

GMM Gaussian Mixture Model. 32, 47

HAR human action recognition. 1, 3, 8, 51, 70, 136, 143, 154

HMM Hidden Markov Model. 22, 32, 47

HOF Histogram of Optical Flow. 23, 24, 84

I3D Inflated 3D Convolutional Neural Network. 30, 106, 118

ICT Information and Communication Technologies. 2

IDT Improved Dense Trajectories. 31, 33, 35, 40, 84

IMU Inertial Measurement Unit. 27, 32, 34, 46

IoU Intersection over Union. vii, 91, 93, 94

k-NN k-Nearest Neighbors. 24, 32, 47

LSTA Long Short-Term Attention. 27, 36, 118

LSTM Long-Short Term Memory. 18, 19, 27, 33, 86

MKBoost multi-kernel boosting. 36

MKL multi-kernel learning. 36

MLP Multi-Layer Perceptron. 20, 40, 46, 47, 70, 102

MS Motion Segmentation. 31

MSCNN Multi-Scale Convolutional Neural Network. x, xi, 102, 105, 106, 118

NLP Natural Language Processing. 101

NMS Non Maximum Supression. 91

NN Neural Network. 18, 27, 31, 42, 47

OF Optical Flow. v, vi, 22, 23, 24, 25, 27, 29, 30, 31, 32, 33, 35, 36, 37, 51, 52, 51, 52, 53, 52, 54, 55, 56, 66, 67, 70, 82, 84, 86, 113, 132, 133, 145

PCA Principal Component Analysis. vii, 23, 24, 33, 95

PoT Pooled Time Series. 24, 33

R-CNN Region-based Convolutional Network. vi, vii, 29, 42, 73, 88, 89, 90, 89, 90, 91, 94, 95, 97, 98, 99, 102, 106, 111, 115, 118, 145, 153

ReLU Rectified Linear Unit. 31, 56, 68

RNN Recurrent Neural Network. vii, 19, 27, 31, 33, 114, 113, 114, 124

RoI Region of Interest. 16, 20, 27, 33, 42, 116

RPN Region Proposal Network. 20, 42, 89, 91, 106

SVM Support Vector Machine. 23, 24, 25, 32, 33, 34, 35, 36, 46, 47, 88

TN True Negative. 61, 77, 81, 93, 102, 105, 115, 116

TP True Positive. 61, 77, 78, 81, 93, 94, 102, 115, 116

TSN Temporal Segment Network. 29, 36, 84, 106, 118

ZS-EAR zero-shot egocentric action recognition. 5, 6, 7, 8, 9, 40, 43, 121, 123, 131, 132, 142, 149, 150

ZSL zero-shot learning. 11, 122, 124, 126, 129, 131, 132, 153

The journey of a thousand miles begins with one step.

Lao Tzu

CHAPTER

1

Introduction

UNDERSTANDING what humans are doing from video data is a pivotal challenge within the Computer Vision community. In contrast to some rigid and controlled environments, realistic human-object interactions are complex in their dynamics and rich in details. So far, the majority of the research has been focused in third-person view videos, in which the visual evidence is taken from a fixed reference point. This is the case for surveillance cameras, for instance.

An alternative to this conventional approach is the use of first-person cameras, mounted in the head or simply worn by people, attached to the clothing. There has been significant research interest in this novel viewpoint of the human action recognition (HAR) field for the challenge it provides: the constant motion generated by the body and head movement, the lack of context as in third-person videos, and so on. Nonetheless, rich features such as the occlusion-free interactions with objects, the focus on the manipulation of objects, the gaze movement, and so forth have been identified in the literature (Li et al., 2015). These cues make the first-person or egocentric action recognition a research field on its own, apart from the third-person research, from which several solutions have been exported. However, exploiting the

intrinsic features of this type of vision seems to be key to correctly recognise the content of a video (Nguyen et al., 2016).

Nowadays, the research line of the egocentric vision has been adopted by various research groups and several solutions have been proposed. Even new features such as the use of sound are being leveraged in recent works (Arabacı et al., 2018) (Cartas et al., 2019a), as some actions cannot be distinguished using visual cues. The field is advancing, although it has yet to get as large as the third-person one. In addition, the results are still far from being acceptable. In fact, the use of egocentric vision techniques for real world settings is still a challenge and more research is required in order to steer new solutions in the correct direction.

In that sense, the zero-shot paradigm (Wang et al., 2019b) is a promising approach to further scale egocentric action recognition (EAR) systems to realistic problems. Apart from the physical scalability of the wearable cameras from this field, the ability to scale to several action classes is also pivotal to determine uncommon events in the daily life. That is, removing some of the constraints imposed by the current datasets in terms of the possible predicted action classes and being able to cover a wider range of options.

The remainder of this chapter is structured as follows: Section 1.1 explains the context and the motivation behind this research; Section 1.2 formulates the hypothesis, the goals to achieve, and the scope of the work; Section 1.3 describes the methodology to achieve the goals; Section 1.4 enumerates the contributions of this dissertation; and Section 1.5 presents a scheme of the dissertation.

1.1 Context and Motivation

One of the current main challenges for the public administration is to promote active and healthy ageing for as long as possible. Achieving it would pose positive consequences for the society and the socio-sanitary services, such as reducing the costs from medicines and other treatments. The latter expenses are becoming more and more worrying with the ageing of society. For example,

Spain dedicated the 9.8% of its GDP to elderly care in 2014 [1]. Given that reports estimate that the world's older population is going to duplicate by 2050 [2], the magnitude of the problem may become unmanageable. Due to this, public administrations are investing in research projects which may help alleviating or avoiding this problem in the future, creating an active and healthy older population. For example, some projects aim at mitigating serious illnesses such as mild cognitive impairment (MCI) (Akl et al., 2015) (Rawtaer et al., 2020) or frailty (Goonawardene et al., 2018) of elderly citizens using unobtrusive technologies. The core idea is to use sensor infrastructures to monitor people's activity and behavior and correlate their evolution with MCI and frailty. This way, suitable Information and Communication Technologies (ICT) enabled interventions can be planned to minimise the consequences of those risks. Such approaches can be developed within smart homes, ICT-augmented houses that provide novel services, and can be grouped in the Ambient Assisted Living (AAL) paradigm, which promotes the use of modern ICT technologies to assist elderlies in their Activities of Daily Living (ADLs). The main objective of the AAL is to avoid the dependence of elderlies on other people in their daily living activities.

In particular, automatic HAR becomes a key enabler for AAL approaches. The activity recognition field has two main approaches: the sensor-based (Chen et al., 2012) (Wang et al., 2019a) and the vision-based (Bux et al., 2017) (Zhang et al., 2019) approaches. The first one makes use of sensors spread around the environment where the activities are going to be held so that each element which we can interact with has a sensor that activates itself whenever someone makes use of that element. Then, we have multiple activations, each one associated with a specific timestamp, as our data to recognise activities. This approach is somehow limited due to the poor scalability of the sensors. The other approach, the vision-based recognition, is a more popular way of recognising activities, specially important in its third-person vision version (surveillance cameras, for example). However, some pivotal problems

[1] https://www.imserso.es/InterPresent2/groups/imserso/documents/binario/112017001_informe-2016-persona.pdf

[2] https://www.nih.gov/news-events/news-releases/worlds-older-population-grows-dramatically

arise from the latter. Namely, the occlusions created due to the environment, the difficulty to keep all the body parts visible, and the limited field of view of third-person cameras (Nguyen et al., 2016). These problems can be addressed using first-person or egocentric vision-based approaches, which use cameras that are attached to the users' bodies and only record the surroundings or, more specifically, what the user is seeing. As the egocentric vision can be very relevant to recognise self-performed activities (Nguyen et al., 2016), this thesis proposes further research on the topic.

The state-of-the-art approaches for the vision-based egocentric action recognition (EAR) are currently based on deep learning (DL) techniques. The effectiveness of DL for tasks related to video learning (taking into account both the feature learning and the classification) has been demonstrated through several papers. For a detailed review of the subject, we recommend reading the work of Asadi-Aghbolaghi et al. (2017b). With DL, there is no need to hand-engineer features, as DL networks automatically learn them from input videos, i.e., they are able to learn characteristics of videos automatically. Furthermore, they are able to output probability distributions over the possible pre-defined classes (for instance, all the possible ADLs) and, thus, the class with the highest probability is the one predicted. However, DL networks are data hungry models, meaning that thousands of samples are needed to learn to discriminate between different classes, as mentioned in the works of Munappy et al. (2019), Tan et al. (2018), and Yu et al. (2015). In fact, ADLs can be generally decomposed into the type of motion (such as cutting, grabbing, and so on) and the objects involved (if any, such as a knife or a door), and all the combinations must be learnt separately, requiring huge datasets with many combinations. Unfortunately, creating them is a laborious task and, usually, video datasets have few samples compared to image or text datasets. Moreover, rare combinations may not have many samples, making it difficult to learn them.

Therefore, to solve the aforementioned issue, this thesis proposes to go beyond the classic supervised training and prediction schemes and apply the zero-shot approach to infer ADLs that may be out of the training dataset's bounds. In the zero-shot setup, in the evaluation phase, a system observes

samples from classes that were not observed during the learning phase. In our scenario, that means predicting actions without seeing any sample at training time. For that, instead of learning action labels, actions are decomposed into the motion (also called the verb) and the active object, having to learn all the possible verbs and objects in the learning phase from samples of action classes that are not used in the evaluation phase. For instance, learning the verb "cut" (from actions not contained in the test set) and many objects separately would allow for inference of many actions related to cut, such as cutting a cucumber (common action) or cutting a sponge (which may be rarer). The motion related to cut is learnt so that, regardless of the object it is accompanied with, it can be discriminated from any other motion pattern. By combining separate predictions, new action predictions can be done without requiring any sample at training time. Hence, this zero-shot setting allows for a broad range of possible predictions, opening new possibilities.

In this thesis, we propose to augment the zero-shot egocentric action recognition (ZS-EAR) approach with external knowledge. The information extracted from the latter would be used to weigh the action probability distribution that the action recognition system would output. For example, that would mean that, for the case of action predictions that do not exist or are not common, such as cutting a fridge (which, in fact, could be possible but it is not likely to happen), their probabilities would shrink. A rich source to do that is text, having large corpora (text datasets) with many actions appearing in them is useful to estimate how probable an action may be in the real world. Hence, the objective of this thesis is to propose a ZS-EAR system that employs estimations from external corpora.

1.2 Hypothesis, Objectives and Scope

Based on the current state of ZS-EAR, the hypothesis of this dissertation is:

Hypothesis. Using external knowledge extracted from text corpora, it is possible to improve the standard classification metrics of the zero-shot egocentric action recognition.

In order to validate this hypothesis, this dissertation sets the following goal:

Goal. To design and implement a zero-shot egocentric action recognition system that leverages knowledge from text corpora for the improvement of the standard classification metrics.

This general goal can be achieved by addressing the following more specific and measurable objectives:

1. To study the current state of the art in EAR and its zero-shot counterpart.

2. To design a suitable DL architecture for the zero-shot problem.

3. To implement a set of supervised classification models that take videos as input and are able to output probability distributions of motion and object labels.

4. To identify the evaluation methodology for the ZS-EAR which better addresses the requirements of the system.

5. To identify and extract the appropriate knowledge sources that can be employed: within the domain of the dataset and those of general knowledge.

6. To quantitatively validate the results obtained with and without the use of external knowledge sources, and with general and domain specific knowledge sources.

The resulting system should also fulfil the requirement of being able to exploit any knowledge source.

In addition, the work presented in this dissertation does not deal with the following conditions:

1. We do not aim to build and leverage the best possible DL architecture.

2. The evaluation of all the possible classes is out of the scope of this work, only those labeled classes from the state-of-the-art datasets will be employed.

3. We assume the conditional independence between verb and object learning for actions.

1.3 Research Methodology

In order to achieve the statement and derived goals presented in Section 1.2, the following strategy has been defined:

1. Explore the literature in the area of DL, Computer Vision applied to DL, action recognition and EAR (and its zero-shot counterpart) to build a solid theoretical framework from which to begin the research. Throughout the thesis, this basis will be updated periodically to be up to date with the state of the art.

2. Evaluate current solutions for the ZS-EAR, analysing their scope and limitations, and identifying their shortcomings in order to propose contributions to the state of the art.

3. Design and develop a prototype of a motion or verb detector and test its performance through several experiments.

4. Design and develop a prototype of an active object detector and test its performance through several experiments.

5. Implement the action inference system leveraging previous verb and active object detectors.

6. Networking by visiting other research labs: the author was a visiting researcher in the Czech Technical University for 3 months in 2018.

7. Redesign the EAR system with feedback from the research stay and the literature.

8. Choose an appropriate evaluation methodology and evaluate the system.

9. Disseminate the results obtained to the scientific community.

1.4 Main contributions

The following scientific contributions can be found in this dissertation:

- An end-to-end DL approach for classifying falls in videos. The fall classification task is presented in Chapter 3, as well as the system structure, the data management, and the evaluation of the proposed approach.

- A methodology to leverage external knowledge to improve the results of ZS-EAR systems that have a branched verb and object learning, i.e., both elements of an action are separately inferred. The system, the data management, and the evaluation are shown in Chapter 5.

1.5 Thesis Outline

The thesis is structured in six chapters:

Chapter 1, the current one, presents the context and motivation, the hypothesis, goals and scope of the research, and the research methodology followed to achieve those goals.

Chapter 2 reviews the literature on the field of EAR, providing a structured taxonomy, and the zero-shot approaches of the field.

Chapter 3 presents the motion or verb detector prototype developed with the aim of being employed for the EAR later. The evaluation is carried out using the fall detection task, providing the necessary background to understand its goals and the evaluation process. Furthermore, an extensive evaluation with three datasets and various experiments to prove the generalisation of the system are performed.

Chapter 4 starts introducing the two main EAR datasets employed through the rest of the dissertation and various active object detector proposals, followed by their corresponding evaluation and discussion sections. An applic-

ation of those methods to the classification of actions is also included for an extensive comparison with HAR proposals of the literature.

Chapter 5 includes the EAR system developed taking into account the reviewed literature and the results from past chapters. It goes deeper into the ZS-EAR paradigm employed for the system and its evaluation, leading to the discussion that aims at validating our proposed hypothesis.

Chapter 6 summarises the final conclusions and insights of the dissertation, as well as the future research lines related to it.

If I have seen further it is by standing on the shoulders of giants.

Isaac Newton

CHAPTER

2

Related work

Since the introduction of the first wearable camera in 1998 (Mann, 1998), the research on first-person videos has gone steadily increasing and becoming popular among the community. In fact, several names have been used for this type of vision: first-person vision, egocentric vision, or even ego-vision. During its brief history, encompassing roughly two decades, there have been several milestones that have promoted the eagerness to explore the characteristics of this type of data. From those that have brought solutions from its third-person or exocentric counterpart to the novel approaches that are tailored to the rich features that egocentric videos present. This chapter will present the research history and advances of this field, as well as the literature that went one step further and took the challenge of the ZSL on egocentric videos.

First, Section 2.1 reviews the works in which egocentric videos are used to infer actions and, then, Section 2.2 is aimed at presenting research on ZSL for EAR. Finally, we make a summary of the chapter in Section 2.3.

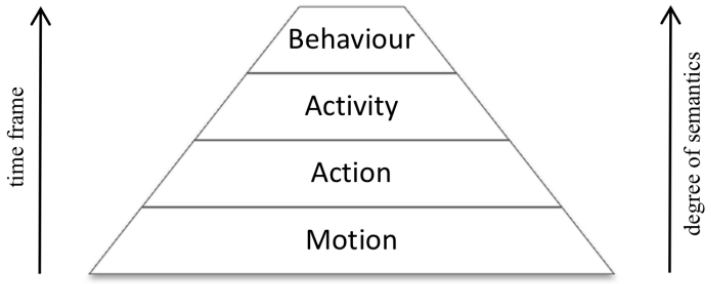


Figure 2.1: Human behavior classification. From the work of Chaaraoui et al. (2012).

2.1 Egocentric Action Recognition

The idea of using egocentric videos has only started to be exploited in the last decade thanks to novel, lightweight, and affordable devices such as Google Glasses or GoPro. In fact, lifelogging has gone mainstream. Indeed, the number of datasets in the state of the art of the EAR field has gone increasing during this decade, with releases such as the large EPIC Kitchens dataset (Damen et al., 2018a). This has also motivated the research on the topic (Kanade and Hebert, 2012) (Bambach, 2015) (Betancourt et al., 2015) (Del Molino et al., 2016) (Nguyen et al., 2016) (Asadi-Aghbolaghi et al., 2017a) (Asnaoui et al., 2017), being mainly divided into three areas: (i) activity recognition/classification, (ii) video summarisation, and (iii) object detection. In this section, we aim to provide an extensive review on the action recognition subfield, referred to as EAR throughout the document.

First of all, it should be noted that the literature presents two conflicting terms: actions and activities. Nguyen et al. (2016) discussed that both things are semantically different: an action is a short event such as ”opening a jar” while an activity is a semantically higher event where various actions are combined, lasting from several minutes to hours. Nonetheless, part of the literature does not take this difference into account and uses the word ”activity” instead of action. Moreover, some works even denote the motion using the word ”action”, i.e., the movement generated when something is being cut would be called an action, regardless of the objects present. In this disserta-

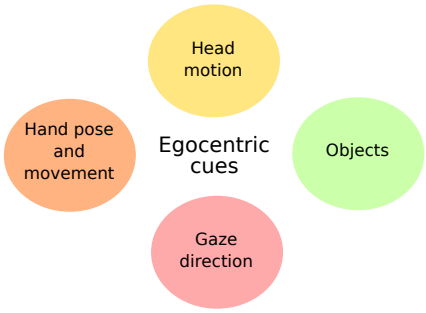


Figure 2.2: Intrinsic egocentric cues.

tion, we will differentiate between actions and activities and between actions and motion, being the motion for us the movement generated from an action independently of the object. A hierarchical classification is depicted in Figure 2.1.

Reviewing the literature on EAR, it is noticeable that there are various special cues intrinsic to egocentric videos that drive the type of approach researchers use to tackle the EAR challenge. For example, Li et al. (2015) stated and used (i) the hand pose and its movement, (ii) the head motion, and (iii) the gaze direction as egocentric cues in their work. In addition, they also stressed the importance of objects in the egocentric setting. In general, from the literature, we can extract the main egocentric features or cues used, summarised in Figure 2.2. Hence, we can split these characteristics into two groups: those related to the appearance or objects and to the movement or motion.

Therefore, in this chapter, we split the literature into four sections depending on the type of modality driving the approaches: (i) object- or appearance-based approaches, (ii) motion-based approaches, (iii) hybrid approaches (combining appearance and motion), and (iv) other approaches that consider not as often seen modalities such as the sound or that are making a contribution that is not related to these modalities. The proposed taxonomy used for this section is illustrated in Figure 2.3. All the references following this taxonomy can be found in Table 2.1.

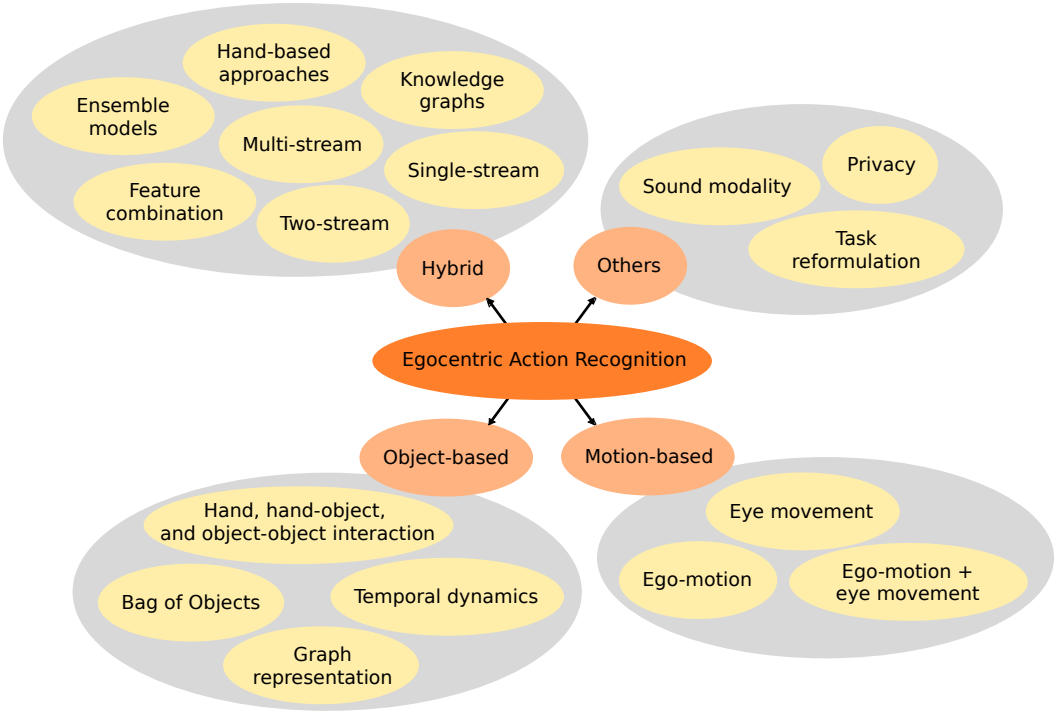


Figure 2.3: The proposed taxonomy used to summarise the literature on egocentric action recognition.

Object – based approaches	
Category	*References*
Bag of Objects approaches	Surie et al. (2007), Pirsiavash and Ramanan (2012), McCandless and Grauman (2013), Fathi and Rehg (2013), Matsuo et al. (2014), Nakatani et al. (2018), Aboubakr et al. (2019), Kapidis et al. (2020)
Hand-Object and-Hand relations	Behera et al. (2012), Behera et al. (2014), Cartas et al., Gkioxari et al. (2015), Bambach et al. (2015), Ma et al. (2018), Tekin et al. (2019), Nebel et al. (2018)
Graph representations	Fathi et al. (2011a), Nagarajan et al. (2020)
Temporal dynamics	Zaki et al. (2017)
Motion – based approaches	
Category	*References*
Eye movement	Yu and Ballard (2002a), Yu and Ballard (2002b)
Ego-motion	Sundaram and Cuevas (2009), Kitani et al. (2011), Ryoo and Matthies (2013), Song et al. (2014), Narayan et al. (2014), Poleg et al. (2014), Singh et al. (2015), Poleg et al. (2016), Kumar (2017)
Eye movement and ego-motion	Ogaki et al. (2012), Yan et al. (2014)
Hybrid approaches	
Category	*References*
Two-stream architectures	Ma et al. (2016), Kwon et al. (2018), Wray et al. (2018), Sudhakaran and Lanz (2018), Li et al. (2018), Verma et al. (2018), Zhong et al. (2019), Wray and Damen (2019), Lu et al. (2019a), Sudhakaran et al. (2019c), Lu et al. (2019b), Yu et al. (2019b)
Multi-stream architectures	Tang et al. (2018), Furnari and Farinella (2019), Huang et al. (2019), Wang et al. (2020), Jiang et al. (2020)
Single-stream, multiple tasks	Singh et al. (2016), Kapidis et al. (2019a), Planamente et al. (2020), Perez-Rua et al. (2020b)
Combination of multiple features	Spriggs et al. (2009), Shiga et al. (2014), Yan et al. (2015), Singh et al. (2016), Zhang et al. (2017), Nakamura et al. (2017), Javidani and Mahmoudi-Aznaveh (2018), Possas et al. (2018), Diete et al. (2018), Zuo et al. (2018), Zuo et al. (2019), Kapidis et al. (2019b), Yu et al. (2019a)
Knowledge graphs	Wray et al. (2016), Sahu et al. (2020)
Hand-based recognition	Zhou et al. (2016b), Garcia-Hernando et al. (2018), Cai et al. (2018)
Other approaches	
Category	*References*
Sound modality	Arabacı et al. (2018), Cartas et al. (2019a), Cartas et al. (2019b), Kazakos et al. (2019)
Task reformulation	Moltisanti et al. (2017), Wray et al. (2017)
Privacy	Dimiccoli et al. (2018)

Table 2.1: Summary of the literature following the taxonomy proposed in Figure 2.3.

2.1.1 Object- or appearance-driven approaches

The current literature is highly dominated by works that believe that objects present in the scene and, specially, those objects related to tasks are the main cues in the recognition of actions. That is, analysing objects in videos can become a critical hint towards recognising an action. In fact, Fathi et al. (2011a) argued that the egocentric paradigm is specially beneficial to analyse actions that involve objects due to three reasons: (i) object occlusions are minimised, as the space where these are manipulated is always present; (ii) objects are often seen at consistent viewing directions with respect to the egocentric camera, as poses and displacement of the manipulated objects are also consistent in workspace coordinates; and (iii) the camera is usually focusing on objects and actions, which are usually in the centre of the image or video, thus obtaining high quality image measurements.

Regarding the classification of objects, there are various ways in the literature to categorise them. Surie et al. (2007), for example, opted for defining objects by the type of space they are in. That is, the space observed by the actor (the one wearing the camera) is known as the *observable space*. Then, any object that is graspable or can be reached using the hands is contained within the *manipulation space*. Lastly, an object that is grabbed by the actor is said to be a manipulated object.

In a complementary way, Nguyen et al. (2016) stated that four types of objects can be observed:

- Active and passive objects: active objects are those who are relevant to actions and passive objects are background or non-important items.
- Salient and non-salient objects: those fixated by the gaze or those in which the focus is put are considered salient objects.
- Manipulated objects: objects that are in hands are said to be manipulated.
- Multi-state objects: those that have changes in terms of color or shape.

It is specially important to stress that active objects are considered the key to estimate the action (Ren and Gu, 2010), but also a challenging task due to hand occlusion or background clutter. To diminish the effect of the background clutter, Fathi et al. (2012) and Fathi et al. (2011a) proposed to first detect a Region of Interest (RoI) before localising objects. In fact, there are authors that aim to detect active objects in an unsupervised way (without categorising them). Namely, Kang et al. (2011) generated a pool of segmentations, individually searching for instances of specific objects (one at a time) by enforcing constraints such as geometric consistency. Damen et al. (2014) used a gaze tracker to infer the most important objects and analysed the interactions with them. Mishra et al. (2011) made a segmentation process in two steps: first, they generated a probabilistic boundary map of the scene and, second, they made use of the fixation point to get the closed contour that includes that point. Sun et al. (2009) presented *EYEWATCHME*, an integrated vision and state estimation system that, at the same time, tracks, among others, the position of hands and active objects. The approaches using the gaze are specially interesting, as Land et al. (1999) and Hayhoe (2000) showed that the eyes always look directly at the objects being manipulated (active objects). In fact, these approaches could be integrated in an action recognition system that aims to use active objects' information.

Bag of Objects approaches. There are several studies in which the bag of objects approach is used. Works such as those of Pirsiavash and Ramanan (2012) and Matsuo et al. (2014) made use of bags of active and passive objects to infer actions, being the objects first detected by an object detector and, then, classified into active or passive. Surie et al. (2007) used two complementary sets: one of observable objects and another one of manipulable objects.

McCandless and Grauman (2013) argued that, as an extension of the classic bag of objects, spatio-temporal binning approaches can capture space-time relations and, to solve the issue of the inflexible predefined schemes, they first proposed to learn the spatio-temporal partitions that are most discriminative. For that, they generated a pool of randomly generated candidates and used a boosting approach to select the best ones. Second, to further improve the

first contribution, they aimed to create object-centric partitions, i.e., regions of videos where active objects are supposed to appear, by creating a histogram of active objects for a video. For the classification, they computed features from each proposal in the pool and applied the boosting operation to get the best proposals that are used to train the final classifier.

One aspect related to this bag of objects are the object fluents, i.e., a time-varying attribute of an object or a group of objects, and its values are the specific states of the attribute (Liu et al., 2017) (Fire and Zhu, 2015) (Mueller, 2014). For example, for a mug, the states or fluents can be *empty* and *full* (binary fluents). Specifically, Liu et al. (2017) proposed to represent an action as concurrent and sequential object fluents. Given an egocentric video, beam search is used to recognise the fluents per frame and then infer actions. The bag of objects used in the work of Fathi and Rehg (2013) was composed of sequences of visual patches of objects (a sequence represents the change of an object during a video). Aboubakr et al. (2019) also modeled object state transitions as a means of inferring actions. In their model, a Convolutional Neural Network (CNN) extracted visual features from a set of frames selected from K segments (uniformly sampled across each video), one per segment. The network was later divided into two branches by means of a point-wise convolution: one in charge of learning nouns and another of learning states. A global average pooling was applied to obtain a feature vector from each of the branches, one per frame. For the noun vectors, a point-wise convolution led to a single feature vector, while, for the states, two channels were left after the same operation. The two channels of the state branch represented the verb (the type of change applied from the pre-state to the post-state), learnt using a Fully-Connected (FC) layer. For the action classification task, another FC layer was used. Kapidis et al. (2020) analysed the use of object detections from YOLO (Redmon et al., 2016) as a tool to detect indoor actions and to experiment with various detection parameters. They observed that object presence is highly correlated with the action and that the lack in the detection of those relations hampered the detection of actions. Thus, they compensated this using the temporal information of objects, i.e., they gathered detections of various frames to get a more complete picture of the scene. More specifically,

they trained a Neural Network (NN) with a per-frame bag of objects to infer the location (physical place), they also did the same using a Long-Short Term Memory (LSTM) network to infer the location using the whole video. Finally, for action recognition, another LSTM was used, including in the input the location and shape of the bounding boxes of the detected objects apart from the presence vectors.

New methodologies to represent the Bag of Objects approach are also arising, such as that of Nakatani et al. (2018). They presented a preliminary work on object-based action recognition in which they detected objects using a pre-trained CNN and they recognised the action without training any other model. Specifically, to estimate the action, they exploited web data to compute semantic similarity between the detected object names and the names action classes.

Hands, Hand-Objects and Object-Object interactions. The interactions between humans (using hands mainly) and objects and between objects themselves are also a quite analysed topic in the EAR field. Behera et al. (2012) presented their bag of relations, which extends the idea of the bag of objects including, not only the object itself, but also the part of the body that interacted with the object (object-body) and the object-object relations. With the same idea of the "bag of interactions", Behera et al. (2014) proposed a Histogram of Oriented Pairwise Relations, in which the spatial relations (distances, orientations and alignments) between visual-words were represented. Similarly, Cartas et al. also aimed to capture hands and the objects that were being manipulated. For that, they leveraged the R*CNN presented by Gkioxari et al. (2015) to detect the primary region (hands) and the secondary regions (objects). The output of that module was given to an LSTM to process the evolution of the video. Going one step further, Tekin et al. (2019) presented a unified model which, given a single RGB image, in a single feed-forward pass, estimated the 3D hand and object poses, their interactions and the object and action classes. They extracted features using a Fully Convolutional Network (FCN), in which each output cell predicted 3D hand poses and object bounding box coordinates. Then, these cells were associated with a vector that contained target values for hand and object pose,

object and action classes, and the overall confidence value. Those predictions with the highest confidence were passed to their *interaction Recurrent Neural Networks (RNNs)*.

In contrast, without the need to include interactions, there is research about the sole use of the shape and pose of hands to determine actions. Bambach et al. (2015) argued that they could infer the actions in their dataset with only that information. To test their hypothesis, they masked out the region where there were no hands and used a CNN to infer the action. Even though the results were not perfect, they showed that there is a high correlation between hands and actions. Taking into account the temporal domain by applying a simple majority voting, they concluded that their results improved as a consequence of the importance that certain hand poses may have, being more distinctive than others.

While the interactions between hand and objects are important, the relation between different objects is also a central element of an action and only a subset of objects may be relevant to the task. That is why Ma et al. (2018) proposed a way to model arbitrary relations between arbitrary subgroups of objects. Their method was first divided into two parts: (i) in the coarse-grained part, a CNN extracted features from each frame, these were passed through a Multi-Layer Perceptron (MLP) and, to join all the features, the Scale Dot-Product Attention (SDP-Attention) of Vaswani et al. (2017) was applied to them; and (ii) in the fine-grained part, the Region Proposal Network (RPN) proposed by Ren et al. (2015) was used to extract object RoIs, which were fed to the Recurrent Higher-Order Interaction (Recurrent HOI) module they contributed: this module employed a learnable attention mechanism to decide the set of candidate objects that are relevant for the action. Finally, the output of both streams were concatenated and a FC layer with a softmax activation was used.

Nebel et al. (2018) investigated the acquisition of additional features that modeled the interaction between hands and objects. For that, they followed the bag-of-visual-words (BoVW) approach to model actions. To infer the class of new samples, Dynamic Time Warping (DTW) was applied to compare the features from a new sample and the ones of the rest of samples. Next, they

trained an object detector to recognise left and right hands. With these detections, the distance to any object could be determined. As active objects should be in contact with hands, those objects that were being manipulated (very close to hands' positions) were considered actives and the distance between both hands and each hand and the active object were computed. The addition of these features to the presence of objects boosted the performance on action recognition.

Graph representations. Graphs are also used to represent actions, as in the case of the work of Fathi et al. (2011a), in which they built a hierarchical graph (a tree-shaped graph) for activity recognition where an activity was composed of action nodes. The latter had some leaf nodes: object and hand nodes. Their goal in inference time was to be able to predict hands, objects, actions and the activities. To train the system, they employed an algorithm similar to the Expectation-Conditional Maximization of Meng and Rubin (1993). Recently, Nagarajan et al. (2020) presented a work where they built a topological map (represented by a graph) of the scene (of the physical space) from egocentric videos. In order to cluster zones, they employed a Siamese network that took pairs of images and found pairs that corresponded to the same zone. Then, the graph they constructed had collections of clips within nodes (representing zones and the clips in which those zones were visited) and edges represented weak spatial connectivity between zones based on how people traversed them. From it, they could infer the primary places of interactions and the actions related to those spaces. Moreover, they showed how to link zones across multiple related environments (such as kitchens from different datasets).

Temporal dynamics. The appearance in a frame, the local features, can be extended to model the whole appearance of the video or, better said, its dynamics and how it evolves. Zaki et al. (2017) proposed to model the high level dynamics of the sub-events within an action by dynamically pooling features of sub-intervals of time series using a temporal feature pooling function. Specifically, each frame was encoded using a CNN, in which each activation neuron was considered a point in the time series, and features were pooled in determined intervals (sub-events) to model the short-term changes.

Then, these sub-event dynamics were temporally aligned and a group of Fourier coefficients were extracted in a temporal pyramid to encode the overall video representation.

2.1.2 Motion-Driven approaches

Apart from the objects cues, which have shown to be relevant in egocentric contexts, there are also cues related to motion: eye movement, hand motion, and head motion. There is also a feature called ego-motion, usually referring to the global motion generated from the movement of the body and the head.

Eye movement. Land and Tatler (2009) stated that a person's eye movement is a valuable source of information to recognise actions. In addition, as mentioned by Bulling et al. (2010), the eye movement can be classified into three types of movements: saccades, fixations and blinks. Saccades are the constant and simultaneous movements of both eyes that are aimed at building a mental "map" of the interesting parts of the scene; fixations are stationary states in which the gaze is fixed on a specific place; and blinks are the regular opening and closing movements of the eyelids. Yu and Ballard (2002a) limited themselves to actions performed on a table and took hand positions, the locations of eye and head, and the recorded ego-videos. Their aim was to be able to segment actions. For that, and based on the fact that eye and head movements are related to the attention as mentioned in the work of Hayhoe (2000), they developed a method to detect attention switches. The tracking was done using a head-mounted ISCAN infra-red video based eye tracker. With this, they divided each video into action segments and used multisensory data to recognise actions. In another work, Yu and Ballard (2002b) explored the movement dynamics of some body parts, namely, the eye (gaze), head, and hand movements. They integrated and modeled the action using Parallel Hidden Markov Models (HMMs): body parts were processed in parallel streams and integrated by the end. The benefit was that it allowed for different sampling rates and different learnt topologies in each stream and that the noise of a stream was isolated without corrupting the others.

Ego-motion. A large part of the literature aims at capturing the ego-motion or the general motion generated from the head movement and employ it to recognise actions. Kitani et al. (2011) stated that there are two types of motion: instantaneous motion (directional component) and periodic motion (frequency component). In the first case, actions such as turning one's head have strong directional component while repetitive actions such as walk have strong periodic components. Sundaram and Cuevas (2009) aimed to recognise interactions (each one composed of the manipulation, the object, and the location) using low resolution images and temporal templates or motion history images. These templates captured any motion detected in a video, using weights inversely proportional to the temporal distance from the frame in which the motion was detected to the current one. For each class, they computed a mean template and experimented with simple image matching, leading to finding out that normalized cross-correlation performed the best. To infer the location, objects, interactions, events, and activities, they proposed a Dynamic Bayesian Network. Kitani et al. (2011) used sparse Optical Flow (OF) vectors as their motion features to encode the ego-motion. They found out that this feature itself is a strong descriptor for actions related to sports. With a different task, Ryoo and Matthies (2013) studied interaction-related actions, i.e., actions that are performed to the observer such as "a person hugging the observer" or "throwing objects to the observer". They went one step beyond the work of Kitani et al. (2011) and explored multi-channel kernels to integrate global and local motion information. They also introduced a methodology that took into account the temporal structure of egocentric videos. Specifically, their global descriptors are histograms extracted from OF data and the local descriptors were composed of 3-D XYT data, i.e., computing salient motion in the video and summarising the gradient values of the detected motion patches. Moreover, they clusterised the motion descriptors and used the visual words approach to represent the video.

Similar to the previous one, Narayan et al. (2014) made use of first-person dense trajectories in their motion pyramidal structure. The relative strengths of motion along the trajectories were then used to create various bag of words descriptors that were later combined into a single descriptor of the action. A

non-linear Support Vector Machine (SVM) was fed with these descriptors to classify actions. Poleg et al. (2014) presented their Cumulative Displacement Curves, a method based on the assumption that, over a long period of time, the average displacement caused by head rotation is practically zero. Therefore, they divided the frames with a fixed grid and accumulated the displacement up to a certain point within each cell (Cumulative Displacement or CD). Analysing trends in these displacements allowed them to focus on long term actions and to avoid small perturbations due to head motion. Moreover, for long-term trends, they convolved the CDs with a gaussian kernel to smooth them. For classification, they obtained various features and statistics computed from these motion vectors and applied an SVM. Song et al. (2014) contributed a new dataset called LENA and provided several experiments on it with various feature descriptors for trajectories (namely, HOG, HOF, and MBH), Fisher Vector encoding, Principal Component Analysis (PCA) for dimensionality reduction, and a linear SVM for the classification step. Singh et al. (2015) argued that there did not exist a method for both short-term (take, put, and so forth) and long-term actions (walking, driving, and so on) and proposed a way to solve the task. Their solution was based on OF, in which they aimed to identify the dominant motion, i.e., motion generated by objects and hands. They compensated the camera motion using a RANSAC-based homography (Fischler and Bolles, 1981) and applied an extension of an Histogram of Optical Flow (HOF). Their classification goal was solely to infer if a video shows a short-term or a long-term action, but this could be applied in an EAR system.

Poleg et al. (2016) aimed to recognise long-term activities (helpful to segment long and unstructured videos) with a CNN architecture. They sampled segments of 4 overlapping seconds from videos, spatially divide each frame into a non-overlapping grid of size 32×32, and computed OF features from two corresponding grid cells in consecutive frames. This led to a cube of size $32 \times 32 \times 2$ (due to the x and y components of flow), which was used to create an stack of shape $32 \times 32 \times 120$ from the whole video, finally employed as input to a 3D CNN. Kumar (2017) extracted features such as Histograms of Oriented Gradients, Motion Boundary Histograms, and trajectories, combined

all of them, and applied PCA to reduce the dimensionality before applying the classifiers: SVMs, k-Nearest Neighbors (k-NN), and the combination of both (SVMkNN). There are some works that provide new ways to arrange motion information, such as that of Ryoo et al. (2015), that presented a new feature representation, called Pooled Time Series (PoT), based on time series pooling of feature descriptors, particularly designed for motion information in egocentric videos. However, it could be applied to any feature descriptor such as HOFs or CNN features. PoT summarised the short- and long-term changes in the descriptors over time, it applied various temporal filters (set of time intervals) that were pooled with various operators and concatenated to obtain a single feature vector.

Combining eye movement and ego-motion. Others, such as the work of Ogaki et al. (2012), combined both approaches and exploited the eye movement and the ego-motion; specifically, Ogaki et al. (2012) analysed the combination of eye movement taken using an *inside* looking camera and ego-motion taken using an *outside* looking camera. For the first case, they presented their own encoding method while for the second one they used global OF values. Yan et al. (2014) aimed to recognise actions in an unsupervised way in an office and a home environment: they employed encoding saccade information (from an inside camera) and OF encoding obtained from the video frames of an outside camera. They introduced two variants of Multi-Task Clustering, including data from different users in their clusters.

2.1.3 Hybrid approaches

So far, the most promising approaches have been the object-driven ones. However, motion-driven methods may add more robustness and, thus, hybrid models are also proposed in the literature. Specially, the Deep Learning approaches dominate the literature on this matter due to their advantage in automatically extracting features from different information sources.

Two-stream architectures. A highly popularised approach in the DL community is the two-stream network presented in the work of Simonyan and Zisserman (2014a) (see Figure 2.4), which employs both RGB and OF

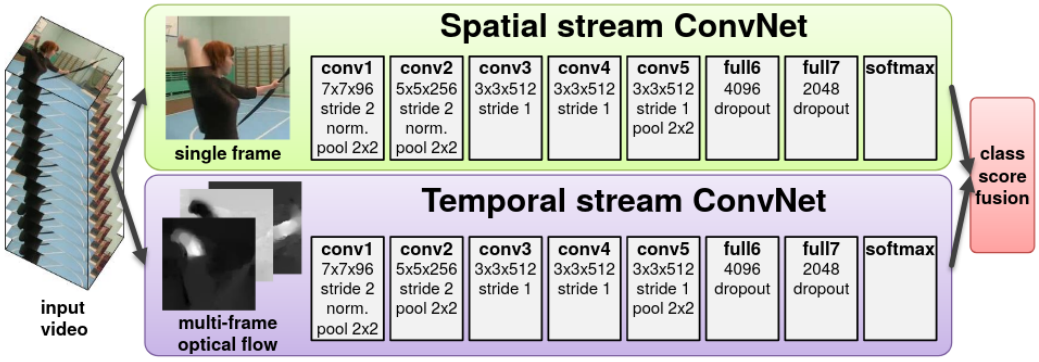


Figure 2.4: Original two-stream network. From the work of Simonyan and Zisserman (2014a).

information as input. This was first used for exocentric vision but was then adapted to egocentric vision (Kwon et al., 2018) (Sudhakaran and Lanz, 2018) (Lu et al., 2019a) (Sudhakaran et al., 2019c). In addition, Simonyan and Zisserman (2014a) observed that networks perform better when they do not need to learn to estimate the motion implicitly. Ma et al. (2016) proposed an improvement for the appearance stream, dividing it into two modules: one for hand segmentation and the other, that took the output of the first one, for object classification. The hand segmentation part segmented and localised hands, creating a gaussian bump in the region where hands are located (or the space between hands). That part was cropped and was fed to the object classification part, which was trained for object recognition. Both this network and the motion stream had their own loss. At the end, both network outputs were concatenated and a FC layer with a softmax activation was used to classify actions, thus, having three different losses for training. The fusion of both branches was done at first with a concatenation operation; however, this was later revisited in the work of Kwon et al. (2018), in which they contributed a long-term fusion pooling to aggregate the features coming from the two branches and they also analysed the effect of various pooling methods, namely, sum pooling, max pooling, and gradient pooling. A combination of all them seemed to provide the best accuracy. An SVM was used as a classifier on top. Instead of employing the standard hard assignment to a single label, Wray et al. (2018) used a soft assignment to various motion labels, e.g., *{open,*

hold, turn, rotate} can denote the kind of motion used to open a jar or a bottle instead of *open*. This representation can generalise to unseen actions in which the motion pattern vary in some way, depending on the active object. Later, Wray and Damen (2019) presented a multi-label verb-only representation for action recognition and action retrieval. Their method allowed for an overlap of labels, removing the ambiguity of previous single label methods. They observed that a multi-verb approach with hard assignment is best suited for recognition tasks and with soft-assignment is best suited for retrieval tasks.

As the two-stream approaches required an aggregation operation for each clip of the video, Sudhakaran and Lanz (2018) proposed to extend the architecture in a CNN-RNN fashion using the Convolutional Long-Short Term Memory (ConvLSTM) network of (Xingjian et al., 2015) as the RNN. Moreover, one of their contributions was a spatial attention layer between the the CNN and the ConvLSTM in the spatial branch: they used Class Activation Maps (CAM) (Zhou et al., 2016a) from a pre-trained CNN to encode the video. Following the idea of Sudhakaran and Lanz (2018) of adding attention mechanisms, Li et al. (2018) developed a NN that jointly classified actions and learnt an attention map distribution using gaze information as supervision during training. An attention map was sampled from this distribution an applied spatially and temporally to the frames in order to guide the action recognition as an attention mechanism would do. At test time, using the received input video, the network could infer both gaze and action. The idea of employing the gaze for an attention mechanism was also exploited in the work of Lu et al. (2019a), who implemented a two-stream network whose spatial branch had an attention mechanism on top. This was composed of a linear transformation supervised with a gaussian bump created from the gaze fixation point, i.e., a 2D gaussian centred in the point the subject of the action was staring at. After that, both branches have a Bidirectional LSTM and, following it, they are fused.

Verma et al. (2018) aimed at demonstrating that a two-stream approach with an LSTM is suitable for egocentric actions without any egocentric feature. Moreover, they also showed that resizing images to adjust the size of

objects to those of Imagenet's ones can potentially improve the results. Sudhakaran et al. (2019c) hypothesised how a CNN-RNN structure could focus on RoIs to better discriminate actions and, for that, they analysed the shortcomings of the LSTM and proposed their alternative Long Short-Term Attention (LSTA). This new RNN introduces a built-in spatial attention and a revised output gating. They deployed their LSTA in a two-stream architecture and also proposed, for the cross-modality fusion of RGB and OF, a novel control of the bias parameter of one modality by using the other. Lu et al. (2019b) aimed to learn spatiotemporal attention features using human gaze as supervision. For that, they proposed a two-stream network, in which each of the streams includes the spatiotemporal attention module (STAM) they contributed. This module includes a 3D inception module and a 3D convolutional layer to predict an attention map. This map is combined with the original feature of the stream to create more informative features. Yu et al. (2019b) advocated for the use of Inertial Measurement Unit (IMU) for the motion classification instead of the OF, arguing that the OF computation is rather demanding. Instead, they created a layered-like approach. The classification of the motion was performed first by an LSTM. Depending on the predicted label, samples were categorised in different motion groups (for example, "standing", "walking", and so on). Within each group, various possible actions could be inferred, but the actions not associated to the motion of the group were discarded (e.g., actions in which it is impossible to be "standing" are discarded if the sample is categorised as "standing"). If the sample was contained within a group with only one action, then this action was predicted. In case there were various possibilities, the motion group was used as a prior for the other branch (the appearance branch), whose objective was to classify the sample among the possible actions of the group using visual features. To adapt the method for low and high frame-rate photo streams, two branches were contained within the appearance stream. For a low frame-rate, a CNN was used and, for a high frame-rate, a CNN with an LSTM. Similarly, Lu and Velipasalar (2018) implemented a two-stream network in which one of the branches passes IMU data through an LSTM. The other branch employs a Recurrent Capsule Network (RecCapsNet) and a ConvLSTM to extract

spatiotemporal features. Then, both branches' features are fed to FC layers (separately), then combined by concatenation, and, once again, the result is fed to a single FC layer. A softmax activation is finally used to provide an action probability distribution.

The application of the two-stream started becoming mainstream, as the architecture was being employed as a baseline. For example, Zhong et al. (2019) focused on hand-hygiene egocentric actions and proposed a method for first locating the action within an untrimmed video using low-cost hand mask and motion histogram features. And, in fact, once the action had been found, the classification was done using a two-stream network.

Multi-stream architectures. While two-stream architectures became popular, a natural extension arose: including more branches containing different multimodal inputs, supposed to be complementary to each other and, thus, helpful to improve the classification of actions. Furnari and Farinella (2019), for the action anticipation task, used three complimentary modalities of data: RGB (for appearance, using a Batch Normalised Inception), OF (for motion, using a Temporal Segment Network or TSN) and object features (confidence scores obtained from an object detector). They introduced their Modality ATTention (MATT) mechanism to fuse them, weighting each of them in an adaptive way, and to predict the action. The use of object detector information was again explored in the work of Wang et al. (2020), who detected a shortcoming in the two-branched architecture (modeling appearance and motion): both failed to exploit local information as there was no position-aware information. In fact, just looking at the motion change or the collection of objects in the scene may not be enough for an annotator to understand the action, that is when position-aware features (referred to as privileged information) could help to drive the learning to action-relevant motion and objects. In addition, they contributed a Symbiotic Attention mechanism for Privileged information (SAP) that allowed for the communication of the three sources of information. A 3D CNN was used to process appearance and motion (outputting a single feature vector) and a Faster Region-based Convolutional Network (R-CNN) for the object features (extracted with RoIAlign). The motion and appearance features were individually fused with the detector's

features and some learnt gate weights (from the opposite branch) were applied to them. One further attention step was applied using the opposite branch's features before obtaining the last feature vector for a branch. Both the verb and the noun were inferred separately and the predictions were combined and re-weighed by the training set's distribution to get the action prediction.

Tang et al. (2018) leveraged depth information in their multi-stream deep neural network (MDNN), having two more branches fed with RGB and OF data. The contribution of this approach is that they aimed to preserve the distinctive characteristics of each stream and to explore the shareable information. That is, as features extracted from each stream are neither fully independent nor correlated, the fusion of these features lacks any meaning. Hence, they proposed a non-linear fusion strategy where they mixed the shareable components and the distinctive components (both obtained with a non-linear mapping of the original features) with a weighted addition. In the loss function, apart from the categorical cross-entropy loss, they included two more terms: (i) a term to measure the correlation between the shareable terms (modeled with a Cauchy estimator) and (ii) a term to enforce the orthogonality constraint on both the shareable components and the distinctive ones. Moreover, they also included a hand module that is fed with the RGB frames. Within this module, a binary mask was generated to black out parts of the original RGB images that were later used for classification. In fact, the softmax output of this module was combined through a weighted fusion with the softmax of the original network.

In fact, multiple streams can arise in an intermediate step of the system, not only at the beginning, as in the case of the work of Huang et al. (2019). They presented a novel Mutual Context Network (MCN) that jointly learnt an action-dependent gaze prediction and a gaze-guided action prediction. RGB and OF frames were processed by an Inflated 3D Convolutional Neural Network (I3D) and fused by addition at the beginning to create the feature F. Then, three parallel modules or branches could be found: (i) the saliency-based gaze prediction module, which outputted a set of saliency map predictions (bottom-up attention); (ii) the action-based gaze prediction module, which took action predictions and created kernels using them, then de-

convolved F with these kernels to create another set of gaze prediction maps; and (iii) the gaze-guided action recognition module, which took the generated combination of saliency and gaze maps, used them to pool the gaze and non-gaze regions of the input features, convolved them, and obtained a probability distribution over the set of possible actions.

Jiang et al. (2020) proposed a branched method (for global and local characteristics) where each branch had tree streams: for RGB, OF, and warped OF, each one having as their backbone network a C3D (Tran et al., 2015). The local branch, in contrast to the global one, was fed with the crop of the salient region of the input frames, which were first aligned. To evaluate each stream's performance, they analysed early and late fusion strategies. To combine both branches, they came up with a cross-fusion strategy, in which pairs of different modalities of data (RGB and OF) and the same modality of data were mixed, and a NN decided which features should be attended. To generate the video-level prediction, they opted for the maximum-weighted-score voting, choosing the label with the highest weighted confidence score.

Single-stream, multiple tasks. Some authors criticise the use of various streams and propose a single stream approach that is trained for various tasks in order to improve the generalisation of the method. Singh et al. (2016) presented an initial single-stream Ego Convnet, which consisted of two convolutional layers, each followed by a max pooling operation, a Rectified Linear Unit (ReLU) non-linearity, and local response normalization (LRN). As the classifier, two FC layers were used. The inputs of the network were encoded egocentric cues: hands masks (as binary images, automatically segmented from the input images), camera motion (horizontal and vertical components separately and corrected using 2D homographies and RANSAC, as grayscale images), and saliency maps (also as grayscale images). These features were taken from a set of L adjacent frames, creating a set of input features with a channel depth of $4 \times L$ (four features per frame). To handle the class imbalance, they proposed using the infogain multinomial logistic loss. Kapidis et al. (2019a) proposed the multi-task learning approach to learn verbs, objects, coordinates for hand locations, and the gaze-based visual saliency at the same time. This allowed for an improved generalisation due to the network

having to follow various objectives and also forced the network to exploit the secondary cues (hand locations and visual saliency) that may otherwise be missed. Planamente et al. (2020) argued that appearance and motion information should be jointly learned, as opposed to two-stream approaches with late fusion. They proposed their self-supervised first-person action recognition network (SpartNet), a single-stream network that coupled both appearance and motion through a Motion Segmentation (MS) self-supervised task, i.e., they included this training objective to force the network to learn the movement of objects. Their architecture was composed of a CNN-RNN structure (the RNN in this case was a ConvLSTM), with a global average pooling, and a FC classifier for action recognition. For the MS, the output of the CNN was fed to another convolutional layer, then to a FC layer. This output was compared with the ground truth using a per-pixel cross entropy loss. For the ground truth, Improved Dense Trajectories (IDT) (Wang and Schmid, 2013) were computed, then each pixel was labeled as moving or static depending on whether movement had been detected for at least 10 frames in these features. The network was trained using both losses with equal relevance. The video attention mechanism of Perez-Rua et al. (2020a) was used in the work of Perez-Rua et al. (2020b) in a single-stream architecture that branches at the end to predict the verb and object that compose an action. The backbone network used in the approach was the Temporal Shift Module (TSM) (Lin et al., 2019).

Combination of multiple features. Part of the literature advocate for the use of different features outside of the end-to-end deep learning systems. Spriggs et al. (2009) employed ego-video and IMU data to tackle the action segmentation and recognition in cooking-related tasks. They explored in supervised and unsupervised settings the performance of Gaussian Mixture Models (GMMs), Hidden Markov Models, and k-NN. For high dimensional data, they observed that k-NN worked better than the other two approaches. Shiga et al. (2014) combined gaze motion (using statistical features) and visual features (in a bag of features approach, more suitable for object and scene recognition). Both branches were independently processed and trained using an SVM. Results were combined by the end to get an action prediction. Singh

et al. (2016) added two more streams to their single-stream approach with deep-learnt features: one that captured the appearance (being fed with RGB images) and the other motion (taking a stack of OF images). Fisher Vectors were used to encode these features and an SVM for classification. To fuse the three streams, weighted classifier scores were taken.

In a similar fashion to the two-stream network, Zhang et al. (2017) presented a two independent branched architecture, being the motion features represented with a Fisher Vector encoding of various features, IDT among them, and being the appearance features encoded with CNNs. They considered the use of early, early+late, and late fusion approaches to train an SVM for action recognition. In the proposal of Javidani and Mahmoudi-Aznaveh (2018), the short-term motion was modeled with OF, while the long-term one was extracted using the PoT representation proposed by Ryoo et al. (2015) with different pooling operators. The appearance of a video was represented with its middle frame. With the concatenated features, an SVM was applied to classify actions. With a different objective compared to the previous authors, Possas et al. (2018) contributed a novel Reinforcement Learning algorithm to save the energy of wearable devices by trading off vision-based action recognition with low power motion-based sensor. For the vision part, a CNN-RNN approach was used, while an LSTM was employed to process the motion. The policy function approximator was implemented using an LSTM and trained using the Asynchronous Advantage Actor Critic (A3C) algorithm. Yan et al. (2015) proposed a multi-task clustering using two methods: the earth movers distance multi-task clustering and the convex multi-task clustering. In addition, they had two test cases: home and office environments, in which different features were used as descriptors, as they argued that the feature selection was not important for their method. Therefore, for the home environment, they employed the object-centric features of the work of Pirsiavash and Ramanan (2012), while, for the office, both the eye motion and the head and body motion were considered, and also OF images.

Nakamura et al. (2017) employed image and accelerator features and aimed at predicting both the action and the energy expenditure (in terms of kilocalories). They concatenated both input features and used an LSTM to model

their evolution in videos. Diete et al. (2018) supplemented the inertial data from motion sensors (from a smart watch) with vision-based features and studied whether this addition can be helpful in settings in which only sensor data is not enough to recognise the action. Zuo et al. (2018) introduced the gaze-informed RoI (GRoI), the region where the gaze is fixated (and, supposedly, an area relevant to the task). Feature extraction was done in that area, then features were encoded, and fed to a classifier. Specifically, the feature encoding consisted of (i) a normalisation step with RootSIFT, see the work of Arandjelović and Zisserman (2012); (ii) a dimensionality reduction with PCA; and (iii) the encoding was done with Vectors of Locally Aggregated Descriptors (Jegou et al., 2011) and Fisher Vectors (Perronnin and Dance, 2007). Later, Zuo et al. (2019) proposed to enhance the local spatial and temporal feature extraction using saliency maps. Kapidis et al. (2019b) aimed to recognise hand-related actions based on the presence and position of detected RoIs in the scene explicitly, without using visual features. For that, (i) they detected hands and tracked them across time and (ii) they also looked for objects that were relevant for the action. The problem was modeled as the learning of the sequence of the detected spatial positions.

Yu et al. (2019a) had three different data streams: one extracting information from images (appearance stream), another one from IMU and GPS data (motion stream), and the last one adding external knowledge included by the user (external knowledge stream). In the appearance stream, a CNN processed images so that it obtained probability distributions, from which several object probabilities could be obtained. The set of all the objects, as strings, are used to compute the basic belief assignment (BBA) for this stream. In the motion stream, several features such as the mean, standard deviation, correlation, and so forth are extracted and an SVM is applied with those features as input. The third one creates a BBA using knowledge acquired from users: for some ranges of the time of the day, the possibility of actions occurring is provided by the user with some confidence values. Given an input timestamp, it is possible to acquire this time-based prior. To fuse all the three streams, Dezert–Smarandache theory (DSmT) (Dezert and Smarandache, 2004) based multi-source fusion is used.

Knowledge graphs. Wray et al. (2016) presented SEMBED (SEMantic emBEDding), an approach to embed egocentric object interactions within a semantic-visual graph (SVG). With this, their aim was to be able to estimate the probability distribution over its potential semantic label. The interactions' verb annotations were unbounded (many verbs could describe the same interactions), thus, embrancing ambiguity in order to capture the semantic relationships and the visual similarities of motion and appearance features. The SVG was built from the training set, in which (i) videos that were semantically linked (they had the exact same verb label) were also linked in the SVG (first type of edge), (ii) nodes that were visually similar, yet semantically different, were linked (second type of edge), and (iii) edge weights corresponded to the normalised visual similarity with neighboring nodes. With the SVG created, they employed the Markov Walk (MW) of Fang and Torresani (2012) and, taking the z nearest neighbors and t steps, they found the probability distribution over the possible labels. Similar to the work of Wray et al. (2016), Sahu et al. (2020) leveraged graphs for the representation. First, they extracted features from individual frames (with the whole frame and also dividing it in four bins) employing a Pyramidal Histogram of Oriented Gradients (PHOG) (Bosch et al., 2007). They also used the center-surround model proposed by Sahu and Chowdhury (2018) to capture the ego-motion. With those features, they built a weighted Video Similarity Graph (VSG), in which nodes represented frames of a video and edges were measurements of the similarity between frames. To infer actions, they used a weakly supervised approach in which only 5% of the frames were required to be labeled and the remaining ones had their label predicted using a Random Walk execution.

Hand-based recognition. Unlike previous approaches in which only the shape and pose of hands or their interaction with objects were analysed, the following approaches exploit both their appearance and motion or leverage hand information combined with other features. Starting from Zhou et al. (2016b), who trained a Deconvolutional Neural Network (DCNN) to infer a pixel-to-pixel segmentation of hands from weakly and strongly supervised data, i.e., massive bounding box annotations and fully annotated segmentation masks, respectively. The network was trained using a novel Expectation-

Maximization (EM) like learning framework. For further improvement, the hand segmentation masks were paired with motion maps (OF) and object feature maps (the top-5 strongest object feature maps) with another DCNN to detect active object regions or the interactional foregrounds. As the objective, both a softmax for object class probabilities and a bounding box regressor were used. They trained two such detectors, one for active objects and another one for passive objects, and histograms were extracted from these representations to model the appearance of actions. For the motion, IDT were extracted from the global image (global motion) and from the active object region (local motion). Combining all these features and applying an SVM classifier, they inferred the action. Garcia-Hernando et al. (2018) focused on hand-related actions and introduced a dataset of 3D hand poses. They presented an extensive experimental evaluation of RGB-D and pose-based action recognition covering 18 baselines (state-of-the-art approaches). They concluded that the hand pose cue is of major help in the EAR field. Cai et al. (2018) presented their work on desktop action recognition, i.e., actions performed while humans are sitting at a desk, and contributed a new dataset for the task. Specifically, they focused on actions involving the manipulation of objects. By extracting various features from hands (e.g., hand shape, position, and motion, inner hand OF, and so on), they analysed their discriminative potential to classify actions. The conclusion they extracted was that hand shape and motion were determining to recognise desktop actions.

Finally, there are some approaches used for challenges that are commonly based on ensemble models, i.e., models such as the ones presented by Sudhakaran et al. (2019a). In their case, they employed an ensemble of CNN-LSTA (Sudhakaran et al., 2019c) and also of Hierarchical Feature Aggregation (HF)-TSN (Sudhakaran et al., 2019b).

2.1.4 **Other approaches**

The remaining of the literature opt for going in other research directions within the EAR field. For example, including other modalities of data or changing some standards of the field.

Sound modality. Sound has not been extensively researched in the literature of the EAR field, having few datasets with both RGB frames and sound. However, it is a promising solution for some actions in which the visual appearance and motion are not enough (see Figure 2.5). For example, Arabacı et al. (2018) suggested combining audio-visual features with multi-kernel learning (MKL) and multi-kernel boosting (MKBoost). Specifically, MKL was used to learn the weights of different features, kernels, and their parameters using the training set. Concerning the features, the authors thought about employing complementary features from different modalities: from videos, they extracted global features using the Grid OF-based Features (GOFF), Vision-based inertial features (VIF) (both from the work of Abebe et al. (2016)), and Log-Covariance (Log-C) features (Guo et al., 2013); local features leveraging Cuboids (Laptev et al., 2008); and, for audio features, they used Mel-frequency cepstral coefficients (MFCCs) (Davis and Mermelstein, 1980). For the classification part, apart from the MKL and the MKBoost, they employed an SVM.

With the launch of the EPIC Kitchens dataset (Damen et al., 2018b), we can expect more works exploring these areas. That is the case of Cartas et al. (2019a), who explored the usefulness of sound. Their system processed an audio spectrogram of the first four seconds of the video (fully covering more than the 80% of the videos with that length) using a VGG-11 network. For the spectrogram, they employed a short-time Fourier transform to filter human voices and focus on noises. In another work, Cartas et al. (2019b) created a three-stream network, taking spatial, temporal, and audio features and combining them using a late-fusion approach. Data was sparsely sampled from the video by dividing it into K uniformly distributed segments. Audio was again represented by its spectrogram. Kazakos et al. (2019) proposed a new architecture for multi-modal temporal-binding (the combination of modalities within a range of temporal offsets) of RGB, OF, and audio. In contrast to previous work, the modality fusion was done before the temporal aggregation and per Temporal Binding Window (TBW), defined in the paper as "a range of temporal offsets within which an individual is able to perceptually bind inputs across sensory modalities". The width of the TBW was dependent on

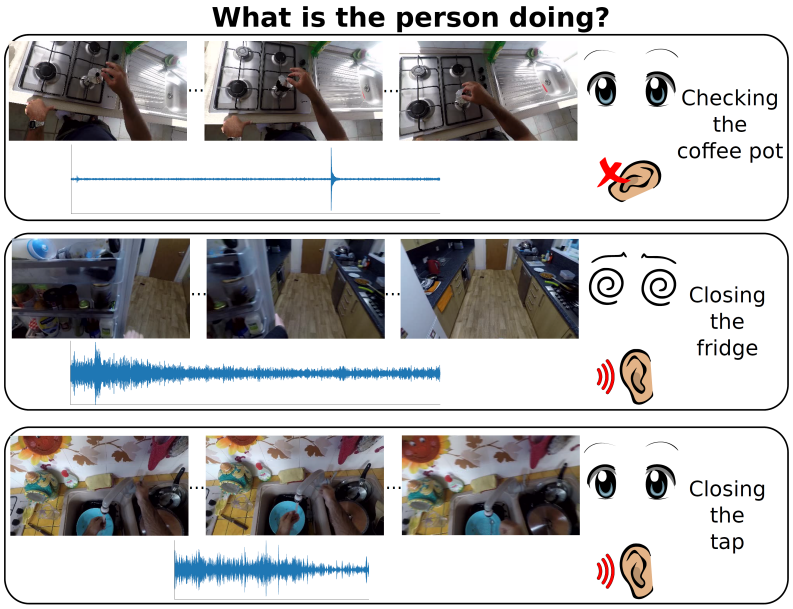


Figure 2.5: Example of the importance of sound for the recognition of some egocentric actions. From the work of Cartas et al. (2019b).

the length of the video (not to bias the network towards short or long actions) and K such windows were sampled from videos. For each window, a three-stream network (with three Batch Normalised Inceptions as backbone networks) processed each modality, then all modalities were fused at a mid-level and a prediction was given for the window. For the video level prediction, all windows' predictions were averaged. As a conclusion, with their experiments they demonstrated that the audio is complementary to the appearance and motion representation of the RGB and OF inputs.

Task reformulation. Some researchers provide novel viewpoints, aiming at changing EAR conventions. For instance, Moltisanti et al. (2017) explored the inconsistencies in the annotation of the temporal boundaries of object interactions within and across annotators and datasets. They argued that this phenomenon is given mainly due to the limited understanding of the different phases of an action and proposed annotating based on Rubicon Boundaries from the Rubicon Model of Action Phases presented by Gollwitzer (1990). Wray et al. (2017) claimed that there are three incorrect assumptions driving the recognition of object interaction as a standard one-vs-all classification

problem: (i) classes are self-contained and have strict boundaries, yet when the number of classes increase these tend to overlap; (ii) sequences can be split into segments containing only one object interaction, but multiple interactions can be given at the same time; and (iii) it is a binary decision whether a verb can be used to label an action, whereas different annotators can label the same action in different ways. Therefore, in their work, they proposed to reformulate the recognition of object interactions as a multi-label classification, obtaining better results compared with the single-label approach.

Privacy. Dimiccoli et al. (2018) aimed at classifying actions while preserving the privacy of bystanders. In contrast to other approaches that selectively filtered regions of images, they proposed to blur the whole image without a significant drop in the performance. In fact, they carried out a quantitative analysis with 640 users to assess the trade-off between the privacy and the performance, reaching the conclusion that degrading egocentric images leads to a more positive perception of privacy, increasing the willingness of users to be captured. They employed CNNs for the EAR and analysed the effect of different levels of blurring and the obtained performance.

2.1.5 Egocentric action recognition datasets

Since 2009, several datasets for EAR have been proposed in the literature, being the main resource for researchers to evaluate their proposals. Table 2.2 summarises the most relevant datasets of the literature and their characteristics. Specifically, we show whether they contain bounding box (BB) annotations, their publication year, the number of action clips (instances used for training and evaluation of machine learning models), and the number of action, verb, and object classes. In the case of Charades-Ego, the dataset is partially egocentric, having part of its content filled with third-person videos. The annotation of actions in all the presented datasets consists of a verb and a set of nouns, creating an action when combined. That may be one of the reasons why popular methods such as the two-stream network approach have adapted well to the egocentric vision, i.e., as the motion and the object fea-

Dataset	Object BBs?	Year	Action clips	Action classes	Verb classes	Object classes
CMU (De la Torre et al., 2009)	✗	2009	516	31	16	33
ADL (Pirsiavash and Ramanan, 2012)	✓	2012	436	32	24	42
GTEA Gaze (Fathi et al., 2012)	✗	2012	511	94	10	33
GTEA Gaze+ (Fathi et al., 2012)	✗	2012	3,371	44	9	29
BEOID (Damen et al., 2014)	✗	2014	742	34	15	20
EGTEA Gaze+ (Li et al., 2018)	✗	2018	10,325	106	19	53
Charades – Ego (Sigurdsson et al., 2018)	✗	2018	30,516	157	33	36
EPIC – Kitchens (Damen et al., 2018b)	✓	2018	50,547	2,747	93	272
EPIC – Tent (Jang et al., 2019)	✗	2019	921	11	6	9
EPIC – Kitchens – 100 (Damen et al., 2020)	✓	2020	89,979	4,025	97	300

Table 2.2: Summary of the most relevant egocentric action recognition datasets ordered by their publication year. Adapted from the work of Damen et al. (2018b).

tures can be decomposed, there are labels to train two separated classifiers and/or to jointly train two branches.

2.2 Zero-Shot Egocentric Action Recognition

The ZS-EAR task aims to infer actions that have never been seen before by a system. That is, the zero-shot system has not used data labeled with the actions it want to infer, so it must exploit characteristics of the input data or acquire knowledge in another way. Following the literature in EAR of Section 2.1, a popular strategy to classify actions, the two-stream (or multi-stream, in general), is inherently built to decompose the two main drivers or components of an action, the verb and the noun, acquire knowledge separately about verbs and objects and leverage it for action recognition. In fact, the idea of fusing verbs and objects to infer new combinations was already introduced in 2017 by Zhang et al. (2017). They employed the Fisher Vector encoding of IDT and Histogram of Oriented Gradients for the verb and visual CNN features for the noun. They argued that the specialised features they considered for each decomposed concept are the key to success in zero-shot tasks. In addition, they analysed various fusion methods among *early*, *late*, and *early+late* stage

fusion. Al-Naser et al. (2018) used a Myo armband sensor data and a MLP for verbs and a GoogleNet (Szegedy et al., 2015) that took crops of video frames, considering that these crops were extracted from the gaze region. Any new action composed from the combination of the learnt verbs and objects could be inferred using their system. The last two approaches are similar to the one contributed in this dissertation except for the fact that we only employ images as input and we leverage external knowledge to improve the final inference result.

Wray et al. (2019) had as an objective the creation of a representation suitable for cross-modal search, i.e., video-to-text or text-to-video queries, in which the query and the target had different modalities. They proposed two Multi-Modal Embedding Networks (MMEN) that embedded video features and text features into the same space, one for verbs and the other for nouns. Specifically, the same video features were sent to the two MMEN while, in the case of the text, the verb and the set of nouns are extracted first and, then, sent to their corresponding module. The representations obtained from the MMENs were further encoded to get a feature for verbs and another one for objects. The network leveraged intra- and cross-modality losses to preserve the neighborhood structure within verb and noun spaces and to ensure that the representation of a query and a relevant item for that query from a different modality were closer than the representation of the query and a non-relevant item. With their approach, they aimed to create verb and noun spaces that are suitable for actions, i.e., for example, for a given verb, independently of the objects accompanying it, the representation should be able to capture its essence.

Apart from the strategy of dividing the learning of the verb and the noun, other approaches could be employed for the zero-shot task. The approaches followed by Wray et al. (2016) and Wray et al. (2018) (discussed in Section 2.1), as the authors mentioned, could be employed for zero-shot EAR, although they were not aimed at that.

Other approaches. There are other works out of the egocentric domain that are relevant for this dissertation due to the methodology or procedure

they use, as they have similarities with the kinds of approaches used for egocentric videos.

The use of text as external knowledge was exploited in the work of Guadarrama et al. (2013), where they aimed to infer descriptions of actions in videos. They leveraged free annotations to construct a semantic hierarchy. More specifically, they searched for triplets of subject, verb, and object. This information was used to describe actions never seen, providing a less specific and more general answer that fitted those actions, as opposed to the case of labeled videos, in which a more specific answer could be given. In contrast, the contributions proposed in this dissertation make use of co-occurrences of words in text to generate a probability distribution. And, instead of using features extracted from text to provide an answer, we leveraged them to support the prediction of the NNs that infer verbs and nouns. Kato et al. (2018) extracted subject, verb, and object triplets just like Guadarrama et al. (2013) from external bases, which contained information about a large range of actions, but they built a knowledge graph instead of a semantic hierarchy. The graph was composed of verb and noun nodes, and also action nodes that were connected to their corresponding verb and noun nodes. The latter ones had their word embedding value (Mikolov et al., 2013) as their node feature, while features of action nodes were initialised to zero at the beginning. Thus, to learn the action representation, information must be propagated through the graph at training time. This was achieved using a multi-layer Graph Convolutional Network (GCN) (Kipf and Welling, 2016) training methodology. Then, the zero-shot action recognition could be done with a simple nearest neighbor search in that space.

Shen et al. (2018) aimed to classify Human Object Interactions (HOI) for images, actions composed of a noun and a verb, similar to the the formulation presented in solutions for EAR. In order to perform zero-shot classification, they disentangled the learning of verbs and nouns using two separated branches. Both have in common a base object detector model, a Faster R-CNN, that provided RoI proposals. The verb module was composed of two parts: (i) the first one took the proposals of the RPN and fed them to two FC layers to get the verb-appearance feature; (ii) the network of Cao et al. (2017)

took the input image, outputted a set of maps representing the probability distribution of human joints location in the input image (one map per joint) and, then, the heatmaps were passed through two FC layers to obtain the verb-pose feature. The object network took the RPN proposals and applied, yet again, two FC layers to get an appearance feature vector. The training of both branches was done jointly, each one having their own bounding box regressor, the verb or noun classification softmax, and their respective losses. Their approach is the most similar to ours, as they had separate branches that output verb and object probability distributions, then combine them to get an action probability distribution. However, we believe that the addition of external knowledge was important to palliate the issue of non-existing actions and the balance of common and uncommon actions for real-world applications.

Even though it did not aim to classify actions, in the system proposed by Nawhal et al. (2019) a branched architecture was employed to disentangle actions. They aimed to generate synthetic data of never seen actions exploiting the knowledge of verbs and objects acquired at training time. For that, they introduced their HOI-GAN, a network similar to the Generative Adversarial Network (GAN) (Goodfellow et al., 2014) in which the verb and object inputs were encoded using word embeddings.

In general, taking into account the review of ZS-EAR done in this chapter, we can categorise the zero-shot approaches into those that employ a branched architecture to separately learn the disentangled concepts that compose the action and those that employ a graph architecture. Moreover, we can also discern among the works in which external knowledge is used and the ones in which they do not apply further knowledge.

2.3 Summary and Conclusions

A thorough review of the egocentric action recognition field has been presented in this chapter, followed by the compilation of works that sought the zero-shot approach within the EAR paradigm. Four main distinct categories of EAR proposals have been seen: those solutions based on objects or appearance, the ones employing motion as their main driver, hybrid approaches that consider

both appearance and motion, and other approaches (still not that abundant) that consider more modalities like the sound or contribute on other topics of the field. Within this division, our work is a hybrid approach that leverages (i) the appearance of the scene to be able to infer the main objects or nouns and (ii) the short-term and long-term motion to predict the type of movement or verb. Among the zero-shot approaches, this work falls in the category of those that use a branched architecture to separately learn verbs and nouns and also in the group that exploits an external knowledge source such as text.

It's all about the journey, not the outcome.

Carl Lewis

CHAPTER

3

Prototype of a motion detector: a fall classification use case

This chapter will focus on the design and development of a basic motion detector, i.e., a system capable of learning motion patterns across time. To evaluate it, a use case related to the topic of AAL was chosen: vision-based fall classification, being falls a major cause of mortality in old adults as stated by Ambrose et al. (2013). In fact, one out of three adults falls, at least, once per year, leading to moderate to severe injuries, fear of falling, loss of independence, and death of the third individual of the elderly who suffer these accidents. Therefore, increasing the research on this task will enrich the AAL community and will serve as an starting point to classify the movement or motion of videos with the objective of being able to adapt our system for actions later.

The chapter is divided into four sections: Section 3.1 introduces the task of fall classification, a brief summary of the literature and the motivation for the task; Section 3.2 describes our motion prototype, the data management, and the experimental setup of the model; and Section 3.3 presents the leveraged

fall classification datasets, the evaluation methodology, the results, and the discussion. Finally, Section 3.4 closes the chapter briefly going through it and pointing out future uses for this prototype.

3.1 The fall classification task

The automatic fall classification task aims at classifying clips or segments of videos into two categories: fall or "no fall". A fall is an involuntary event usually started from the standing position of a person and finished when the person that has fallen, in general, lies on the floor. It can happen that the person does not completely lie on the floor, but rather in a bending down position, depending on how and where they fall, if the person was able to hold an object while falling, and so forth. Any sub-event or part of that movement can be categorised as a fall. In contrast, any ADL or the absence of movement is labeled as "no fall". However, there are some movements within this group that may get confused with falls because of their similarity, for instance, squatting or bending over. These type of events are called confounding events and the goal is to correctly classify them as a "no fall".

To recognise falls, the literature proposes two methods: the sensor-based and the vision-based approaches (Mubashir et al., 2013). The former has commonly relied on the use of accelerometers, specially in the vertical acceleration measures obtained from them. Apparently, these measures are different between falls and ADLs (also including confounding events). Therefore, Vallejo et al. (2013) and Sengto and Leauhatong (2012) proposed giving 3-axis (x-, y- and z-axis) accelerometer data to an MLP model. Kwolek and Kepski (2014) combined IMU and depth maps from a Microsoft Kinect camera as their input data. Depth maps are images that represent the distance of surfaces from the viewpoint of the camera. For the classification, they first checked if the measured acceleration was beyond a certain threshold, providing the first hint. If the event was categorised as a fall, the location of the putative fall was extracted from the depth map and classified again leveraging an SVM. Similarly, Harrou et al. (2016b) used accelerometer data to detect the presence of falls, using a Shewhart control chart for that, and employed

the visual data as a support to classify confounding events. Even thought these last two approaches relied on sensor data for their prediction, they still required to use vision-based solutions to ascertain their prediction.

Meanwhile, the vision-based approach makes use of frames of videos (their visual content) to detect falls. For that, they first extract meaningful visual features such as silhouettes or BBs using Computer Vision techniques to ease the task. Then, these features are fed to Machine Learning classifiers (e.g., GMM, SVM, and MLP) as in the sensor-based approach to predict whether a fall has occurred or is taking place at the moment. Tracking techniques are also quite popular, as in the case of Lee and Mihailidis (2005), in which they detected the person, subtracted the background information, and computed the silhouette for the human using the connective-component labelling technique (Devijver, 1984). Applying rules over the motion vectors computed from the silhouette (such as the magnitude of the vertical component), the probability of falls happening in clips could be inferred. In a similar fashion, silhouettes were also employed by Rougier et al. (2011). Alongside a matching system that tracked the deformation of the silhouette, they quantified the shape deformation using shape analysis techniques and used a GMM to detect falls.

Another popular vision-based approach is based on the features within BBs, analysing them to see if they contain a person or not and, then, classifying the content of BBs (Miaou et al., 2006) (Liu et al., 2010). More features were extracted by Vishwakarma et al. (2007) from BBs, namely, aspect ratio, horizontal and vertical gradients of objects, and fall angle. All these were used as input to a GMM to get predictions. Charfi et al. (2012) extracted 14 features (the height and width of BBs, aspect ratio, and so on), applied various transformations to them (the first and second derivatives, the Fourier transform, and the Wavelet transform), and used an SVM to do the classification step.

Many solutions follow the supervised learning paradigm, i.e., extracting a batch of features from raw images and using classifiers to learn models from labeled data. For instance, Zerrouki et al. (2016) computed occupancy areas

around the body's gravity center, extracted their angles, and studied the performance of various classifiers using that data: Naive Bayes, k-NN, NN, and SVM. The latter was found to be the most promising one. Later, the same authors extended this work adding, as an extra feature, Curvelet coefficients and, as an extra classifier, a Hidden Markov Model (HMM) (Zerrouki and Houacine, 2018). The unique strategy of Harrou et al. (2016a) was to apply Multivariate Exponentially Weighted Moving Average (MEWMA) charts. Nonetheless, their method did not allow to distinguish between falls and confounding events. In fact, a great number of false alarms can be triggered from not being able to discriminate between those two types of events.

The 3D vision is also a largely employed tool within the vision-based approaches, i.e., making use of 3D structures to model falls. It requires the use of multiple cameras (passive systems) or depth cameras (active systems) such as Microsoft Kinect or time-of-flight cameras. The Kinect camera is specially popular due to its low price and high performance. Auvinet et al. (2010) leveraged a Kinect camera to build a 3D silhouette to, then, analyse the volume distribution along the vertical axis. Using the same technology, Gasparrini et al. (2014) extracted 3D features and applied a tracking system to detect falls. Moreover, the Kinect software also provides body joints. These were used in the work of Planinc and Kampel (2013) to get the orientation of the major axis, i.e., a straight line crossing the selected joints. Besides, the 3D ground floor and the spine distance to the ground floor were estimated. To detect a fall, they checked whether the major axis of a person is parallel to the ground floor and whether the height of the spine is near the ground floor. Diraco et al. (2010) made use of a time-of-flight camera with an off-line calibration procedure that automatically estimated the external camera parameters without landmarks, calibration patterns, or user intervention. The moving regions (in which motion is found) were localised in real-time with a Bayesian segmentation applied to the whole 3D point cloud obtained from the camera. Mastorakis and Makris (2014) went beyond the classic 2D BBs and created 3D ones, whose velocity and inactivity calculations (i.e., contraction or expansion of their width, height, and depth) were the cues to detect falls.

So far, the presented methods have proposed hand-engineered features or approaches that did not have proofs of being able to generalise to other scenarios. In contrast, in this chapter we aim to present, not only a proposal of a motion detector prototype, but also a system capable of generalising to many scenarios and automatically choose and extract the appropriate features from videos without the need of an expert. Therefore, we first propose to use the fall classification task to evaluate this motion detector prototype, but the details of the system will be reused in later chapters for other tasks. For this use case, a binary label is generated from the system (representing whether a fall has occurred), although the goal is to be able to generalise to multiple classes later. This is actually not an issue, as adapting the system for more classes is straightforward, but we do not focus on how to adapt the system for actions, or multi-class problems in general, within this chapter, this will be covered in later chapters.

Moreover, in contrast to our aim of using egocentric videos, as fall classification datasets are mainly composed of exocentric videos, we use exocentric data to train and evaluate our system. Technically, this does not imply that any changes are needed in the system itself, as the same type of data structure is employed in both exocentric and egocentric cases. In fact, exocentric data is used in egocentric systems to assist the learning at the beginning. However, we chose to use this use case for its resemblance to the AAL field in which this dissertation falls, and also because of the abundance of exocentric data, providing a suitable task for the evaluation of the system. For all this, we believe this task suits well the prototype presented in this chapter.

3.2 **Proposed prototype**

In its simplest form, a motion detector is a system capable of inferring the movement in a video. This section goes deeper into that definition and aims to explain what does the system consist in and how the video is used in that system.

3.2.1 Data pipeline

2D data recorded by 2D cameras were used as input of the system. The are two reasons for this choice, the first one is that the majority of the already deployed cameras are 2D cameras, recording falls (surveillance cameras) or others events in both public and private spaces (such as smart homes). This means more data is available for systems to exploit, allowing for a wider range of research opportunities. The second reason is that, taking into account the future application of this system to the field of action recognition (both exocentric and egocentric), the majority of large action datasets are mainly composed of 2D videos, even though some of them have data of other modalities such as depth data or sound. Therefore, this type of data is suitable for both this chapter and the proposed contribution of this dissertation. In addition, the system exploits exocentric vision data. The reasons for this are the following ones:

- The knowledge acquired from the large exocentric vision datasets can be reused for the egocentric vision and, in fact, this approach leads to better results as we will explain latter. Moreover, the choice of exocentric or egocentric vision does not affect the system itself, so it does not imply changes in it.
- We have not found egocentric fall classification datasets in the literature, but we consider that the task is relevant enough for this dissertation and that a binary classification suits well a prototype.
- The type of data processing used is generally seen for exocentric videos and, therefore, it is interesting to test it first before applying it in egocentric videos.

Due to all of this, the videos considered in this section are recorded from a third-person viewpoint (such as the ones of surveillance cameras) and each video consists of a set of frames. A frame or an image is a $\mathbb{R}^{H\times W\times C}$ tensor, where H and W are the height and width of the video, respectively, and C is the amount of channels. That is, an $H\times W\times 1$ matrix represents the amount of

intensity in each point in the space with an integer value ranging from 0 to 255. These points of the space are called pixels. However, with just one of these matrices we could only represented grayscale images containing colors ranging from white to black (being gray a mid-step between those extremes). With the addition of the concept of channels, we can represent more colors having C matrices or channels, each one representing the intensity of one primary color. The dataset we employ in this dissertation follows the RGB scheme, in which C is fixed to 3, as we have matrices for the colors red (R), green (G), and blue (B). The combination of these three colors and their intensity per pixel is what is usually observed in an image, in which a wide range of colors can be seen. Whenever we refer to a video, we refer to the set of individual frames, usually ordered, and, thus, to a list of shape $N \times H \times W \times C$, where N is the number of frames of a given video.

Therefore, a task such as fall classification involves processing the whole temporal span of videos, i.e., from the first to the last (N^{th}) frame. RGB images by themselves represent the appearance of videos, which could be useful for tasks such as object detection or place classification. However, given the system we aim to build, there are better representations when the feature we seek to learn is the motion. Similar to the state-of-the-art approaches in HAR and EAR at that time, Simonyan and Zisserman (2014a) and Ma et al. (2016), respectively, we aimed to employ OF images to encode motion. The two mentioned approaches, two-stream networks as the ones reviewed in Section 2.1, used these type of images in their motion streams (the part of the system that is in charge of learning the motion patterns) and, thus, we believed OF images are appropriate given that our desired system resembled those two approaches.

More specifically, Optical Flow or optic flow is a representation of the motion generated by objects, surfaces, and edges in a scene, caused by the motion with respect to an observer. That is, how all these elements move with respect to the camera, from one frame to the next one, is represented in a single image. In fact, the magnitude, velocity, or intensity of the movement are encoded too. As with the images, an OF image is a $OF \in \mathbb{R}^{H \times W \times C}$ tensor, where C equals 2. The first matrix of this tensor (first channel) represents

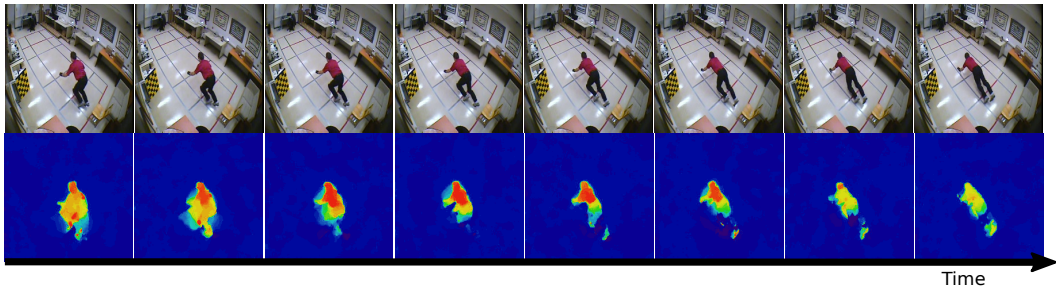


Figure 3.1: Examples of RGB (top) and OF (bottom) images. The bottom row of OF images are computed from the top row RGB images. From the work of Nunez-Marcos et al. (2017).

the motion in the x-axis direction, defined as OF_x, while the second channel does the same for the y-axis, as OF_y. When both are combined, the overall magnitude for a specific pixel (i, j) can be computed as $\sqrt{(OF_x^{i,j})^2 + (OF_y^{i,j})^2}$, where i, j are coordinates in the space of the OF image. Figure 3.1 can be used to compare the differences and similarities between RGB and OF images. The bottom row of OF images are computed from the top row RGB images.

Going beyond the idea of encoding the motion between two adjacent frames, short-term motion can be encoded using a small set of adjacent OF images (referred to as a stack). For that, Simonyan and Zisserman (2014a) proposed to stack L OF images so that the resulting tensor had a shape $H \times W \times (C * L)$, taking into account that each OF image has a C of 2, for OF_x and OF_y. The same idea is employed in future works such as those of Ma et al. (2016) and Wang et al. (2015). Thus, the stacking would be ordered as $\{OF_x^t, OF_y^t, OF_x^{t+1}, OF_y^{t+1}, ...OF_x^{t+L}, OF_y^{t+L}\}$, alternating between the horizontal and vertical components, where the superscript represents the index of the OF images, starting from the t^{th} OF image and covering the next L images. Moreover, there are various algorithms to compute the OF, each one with its own implementation. In our case, we used the TVL-1 algorithm (Zach et al., 2007) as in the work of Simonyan and Zisserman (2014a), as it has a better performance with changing lighting conditions compared to other OF algorithms.

To sample these stacks of L OF images from the video we employed a sliding window approach with a stride of 1. That is, we got the first stack by

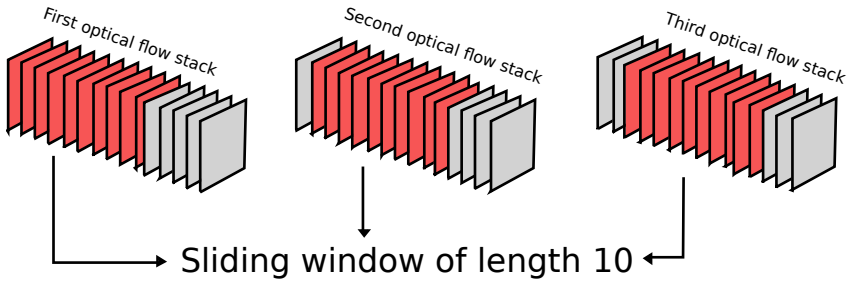


Figure 3.2: The sliding window strategy with a stride of 1 applied to extract all the possible chunks of L OF images from a video. The sliding window is highlighted in red. From the work of Nunez-Marcos et al. (2017).

extracting the first L frames (from the 1^{st} OF image to the 10^{th}), the second (from the 2^{nd} OF image to the 11^{th}), and so on. More formally, given a set of N OF images and a sliding window of size L, we got the t^{th} stack taking the images $\{OF^t, OF^{t+1}, ...OF^{t+L}\}$, where $t = \{1, 2, ...N - L + 1\}$. In total, we got $N-L+1$ stacks per video instead of the N/L stacks we would got if the sliding window had a stride of L (non overlapping windows). See Figure 3.2 for a graphical description of this process.

Each stack has its own label, i.e., "no fall" or fall. This labeling is produced from the annotations of datasets that have a label per frame. As it is not obvious how to label a stack, we divided videos containing falls in three parts: the part previous to the fall, the fall, and the part after the fall. The sliding window approach is applied to each part separately, without overlapping. All the stacks within the fall class were labeled as such, and the same reasoning applies to the stacks of the "no fall" class (before and after the fall). Stacks of videos without falls were just annotated with the "no fall" label. We will talk about the specific datasets that were employed for this process in Section 3.3.1.

3.2.2 Video processing system

In the recent decade, the DL approaches (LeCun et al., 2015) have become the standard for video related tasks because of their state-of-the-art results, allowing to skip the step of the hand-engineering of features, a quite tedious task for video processing algorithms. Instead, DL algorithms automatically

learn the set of features that better suits the problem given the provided data, as the internal variables are adjusted with an iterative optimisation process using that data. More specifically, the DL algorithms that can process 2D or even 3D inputs such as images or videos are CNNs.

In this work, we instantiate a type of CNN called VGG-16 (Simonyan and Zisserman, 2014b), a standard network in the DL community that takes 2D inputs. See Figure 3.3 for a scheme of its architecture in a high abstraction level. That is, when talking about CNNs, researchers usually describe architectures in a high level talking about layers. These type of networks are basically composed of a group of layers, each one connected to the next, where each layer has its own functionality depending on the type of layer instantiated. In addition, these networks output predictions, i.e., probability distributions over sets of pre-defined classes. In this case, as we have the binary problem of recognising falls, this is implemented by a single number p_{fall}, the probability of a clip containing a fall. Taking this into account, the system outputs the following:

$$y = \begin{cases} 0 & if \;\; p_{fall} <= t_{fall} \\ 1 & if \;\; p_{fall} > t_{fall} \end{cases} \tag{3.1}$$

The system predicts a 0 if the system considers the clip to contain a fall. In contrast, if the video is considered to be a "no fall" event, then the output is a 1. t_{fall} is a threshold value between $[0, 1]$ that usually takes the value 0.5. See LeCun et al. (2015) for more details about DL networks and how these are optimised.

Nonetheless, standard CNNs are usually adapted for images of 3 channels. Therefore, the inputs described in Section 3.2.1, with shape $H \times W \times (C * L)$, cannot be directly fed to the system. For that and to pre-train the network so that the learning of fall patterns is easier, Wang et al. (2016b) and Wang et al. (2015) proposed a cross modality pre-training used nowadays in a standard way. First, a training with the large Imagenet dataset (Deng et al., 2009) is performed, using RGB images (the standard VGG-16 is not modified so far). This step allows for a better initialisation of the variables of the network

Figure 3.3: Architecture of the VGG-16 network. Green blocks are convolutional layers, orange blocks are max pooling layers, purple block are Fully-Connected layers, and the blue block is a softmax activation. From the work of Nunez-Marcos et al. (2017).

compared to a random initialisation, allowing for a faster convergence in the next steps. The second step aims at changing the type of data used, i.e., going from RGB to OF. A large dataset such as UCF101 (Soomro et al., 2012) is used as in the case of Imagenet. To adapt the domain of classes (and their number), the last layer is dropped and another one, randomly initialised, is added. In these two steps a technique called transfer learning (Tan et al., 2018) is applied: it allows to reuse the knowledge acquired in the first training to benefit the second training, usually obtaining better results and faster convergence. However, in this last step the shape of the data is changed, so the fix proposed by Wang et al. (2016b) and Wang et al. (2015) has to be applied. The first layer is a convolutional layer with K filters, each one with M channels, hence, having $K \times M$ values. The paper proposes to do a per-pixel mean across the M channels (i.e., mean of M values), obtaining a single channel of shape $K \times 1$. Each kernel's matrix is repeated $C * L$ times, i.e., the amount of channels of the new input. The new kernel shape ends up being $K \times (C * L)$, being compatible with OF stacks. After the UCF101 training, we wanted to proceed to do the training with our target dataset, so the last layer was dropped once again and a new layer was added, having a single neuron that is the one in charge of outputting p_{fall}. Moreover, this last training had the convolutional layers' variables frozen, i.e., these variables did not change their values, only the last layers, the FC layers, were tuned. This kept the knowledge acquired from previous steps untouched and focused the learning on the classifier.

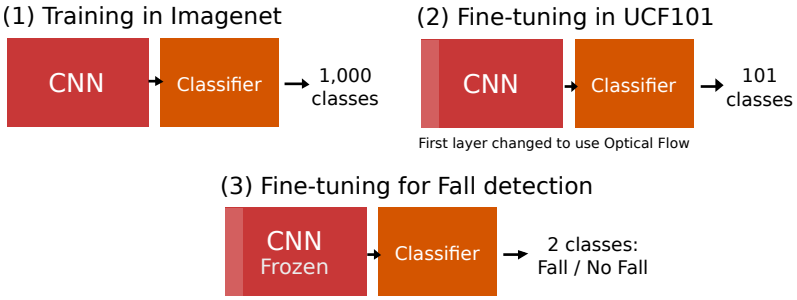


Figure 3.4: The cross modality pre-training of Wang et al. (2016b) and Wang et al. (2015). An Imagenet RGB training is used first, then the last layer is dropped and a UCF101 OF training is done. Finally, the last layer is dropped again and a training with the target OF dataset is done, in this case, for falls. From the work of Nunez-Marcos et al. (2017).

Training details. The step of optimising the network's variables using data is called the training of the network. To do so, some hyperparameters must be set beforehand, as DL architectures are not completely free of hyperparameters.

The input requires a normalisation step before being fed to the system. There are various ways to normalise in the literature, for the case of OF images, subtracting the value 128 is enough to get zero-centered values, ranging from -128 to 127, including both.

Regarding other architectural decisions, Batch Normalisation (Ioffe and Szegedy, 2015) was used before applying the ReLU non-linearity after each FC layer. This removes the internal covariate shift issue, also acting as a regulariser. In addition, Dropout (Srivastava et al., 2014) is used between each FC layer (after the ReLU activation) to help reducing the overfitting, i.e., to prevent the network from learning by heart the training set, the details, as the network is supposed to be able to generalise to unseen cases after the training phase ends. The dropout values were fixed to 0.9 and 0.8, specifically, as in the work of Wang et al. (2015). That is, only 10% of the neurons remained active in the first case, 20% in the second. This strategy is quite aggressive, as the common value used in dropout is 0.5.

Regarding the data augmentation techniques, we did not apply any. However, the datasets that were employed were quite imbalanced, meaning that

the "no fall" class had much more samples than the fall class. This created a problem in the training, since the "no fall" class would have been benefited. To minimise that, the data labelled as "no fall" was resampled without replacement so that both classes had the same number of samples. This technique is called undersampling.

To further mitigate the issue of the imbalanced classes, we also added class weights to the loss function in some cases to increase the importance of the fall class. The loss function expresses how far the predictions of the network are from the ground truth labels (the real ones) and is used to guide the training and adjust the values of the network. The chosen loss function, the binary cross entropy loss, has the following expression:

$$loss(p,t) = -(t * log(p) + (1-t) * log(1-p) \tag{3.2}$$

where p is the prediction of the network and t is the ground truth. With the class weights, two scaling factors, w_0 and w_1, were added, also adding two more hyper-parameters to be tuned. The resulting expression is the following one:

$$loss(p,t) = -(w_0 * t * log(p) + w_1 * (1-t) * log(1-p) \tag{3.3}$$

With this approach and without loss of generality, if we use a class weight higher than 1 for the class 0, that is, w_0, it would penalise the loss function more for mistakes done with samples of class 0 than those of class 1. As the neural network modifies its variables or weights using this loss value, the network is encouraged to prioritise the learning of the class 0 in order to minimise the loss. This may worsen the performance on class 1, though, so the values need to be carefully chosen. This example can be applied for the learning of the fall class, being prioritised over the "no fall" class. In fact, the drawback of worsening the "no fall" class may not be that important in the context of fall classification, as being able to correctly classify all the falls is of pivotal importance in comparison with creating a few false alarms.

Given that loss function, the optimisation was done using the Adam optimiser (Kingma and Ba, 2014), with all the hyper-parameters except for

the learning rate set as in the original work. The network was trained for $3000-6000$ epochs and the early stopping strategy was applied. This is a regularisation tool to reduce the overfitting in an iterative optimisation method using the training data to improve the model's ability to predict unseen cases up to a certain point, in which its performance on the training sets keeps increasing but the generalisation starts to get worse. The early stopping allows the model to stop at that point by monitorising a specific metric or criterion, the loss in the validation set in our case. If the metric does not improve for some pre-defined number of epochs (the patience), then the training stops. The patience is set to 100 epochs in our experiments.

3.3 Evaluation of the prototype

To evaluate the motion detector prototype defined in Section 3.2, various experiments were set up using fall detection or classification datasets. First, the datasets employed are presented in Section 3.3.1, then the metrics and the evaluation strategy in Section 3.3.2, and, finally, the results and discussion are presented in Section 3.3.3.

3.3.1 Fall classification datasets

The datasets that have been chosen are commonly used in the literature and, thus, they are a useful resource for benchmarking. These are the the UR Fall Dataset (URFD)[1], the Multiple Cameras Fall Dataset (Multicam)[2], and the Fall Detection Dataset (FDD)[3]. The specific details of each dataset are summarised in Table 3.1.

URFD is composed of 30 videos of falls and 40 videos of ADLs, labeled as "no falls". From there, 60 "no fall" videos can be extracted from each fall, i.e., the pre- and post-fall parts.

Multicam contains 24 videos or scenarios, the first 22 videos have at least a fall and the remaining two are only composed of confounding events and

[1] http://fenix.univ.rzeszow.pl/ mkepski/ds/uf.html
[2] http://www.iro.umontreal.ca/ labimage/Dataset/
[3] http://le2i.cnrs.fr/Fall-detection-Dataset?lang=fr

	URFD	*FDD*	*Multicam*
Fall sequences	30	99	184
"No fall" sequences	100	130	376
Total sequences	130	229	560
Fall stacks	1,470	3,667	14,376
"No fall" stacks	20,972	77,496	535,262
Total stacks	22,442	81,613	549,638

Table 3.1: Fall classification datasets' details.

ADLs. Each scenario has been recorded using 8 cameras, each one providing a different viewpoint. The scenario used to record is the same for every video and camera, with the exception of some furniture reallocation.

FDD is recorded in 4 different stages (two coffee rooms and two homes) with multiple actors.

These datasets are recorded under the following constraints:

- They have controlled environments. That is, the scenario is set specifically for the dataset, closed, with controlled lighting, and without possible interruptions such as a person walking into the room.
- There is only one actor in the scenario, although that person can be different through the dataset. Therefore, all the ADLs and falls are performed by at most one person.

There is a variation in the distance between actors and cameras. In the Multicam dataset, in which various cameras are available, this distance varies significantly among cameras, providing much more variation. In fact, the actor can be seen from the rear, back, and from various side views. In the FDD dataset some falls are also far from the camera, but that is not the general case, just like in the URFD.

The division of the videos has been first mentioned in Section 3.2.1 to clarify how the inputs of the system were labeled. Regarding the segmentation itself, videos come with annotations of two types: (i) annotations per frame (if

a frame is part of a fall or not, for example) as in the FDD and the Multicam datasets and (ii) start and end frames of events. The second case is more common but, in any case, the beginning and ending position can be inferred from the first type of annotation. Inputs are generated from the segmented parts of the video, without overlapping.

3.3.2 Evaluation methodology

The task of fall classification, in the supervised learning setting in which it is analysed, is a binary decision problem; classifiers must decide whether video sequences contain falls by outputting p_{fall}. For this task, metrics that are not biased by the number of samples are required, i.e., those that are not affected by imbalanced class distributions, in which one class has more samples than the other. This is an issue that may lead to wrong conclusions. In fall classification datasets, the number of fall samples are usually much lower than those of "no fall" events, as falls are supposed to be rare events (see Table 3.1 for some references). That is also why the prediction of the fall class is prioritised, even at the cost of some false alarms as discussed in Section 3.2.2. To provide a specific example, a biased metric such as the accuracy provides an overview of how many samples have been correctly classified. Let us assume that the number of "no fall" samples is N and the number of fall samples is M, where $N >>> M$. If the N samples are correctly classified and no sample within the fall class has been predicted, the accuracy would be very high, as the majority of samples are contained in the "no fall" class.

Therefore, taking into account this issue, the most common metrics to evaluate the performance of these type of classifiers are (i) the sensitivity, also known as the recall or the true positive rate, and (ii) the specificity, also called the true negative rate. These metrics are suitable for the fall classification tasks as they are not biased by the number of samples of each class. The sensitivity is a measure of how good our system is in predicting falls, whereas the specificity measures the performance for the "no fall" class. Nonetheless, for the sake of comparison with other existing approaches, we also computed the accuracy, the False Alarm Rate (FAR), and the Missed Detection Rate

(MDR) metrics. However, notice that the accuracy does not allow for a fair comparison between two approaches.

More formally, the sensitivity, specificity, accuracy, FAR, and MDR metrics are defined as follows:

$$Sensitivity/Recall = \frac{TP}{TP + FN} \tag{3.4}$$

$$Specificity = \frac{TN}{TN + FP} \tag{3.5}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.6}$$

$$FAR = 1 - Specificity \tag{3.7}$$

$$MDR = 1 - Sensitivity \tag{3.8}$$

where TP refers to True Positives, TN to True Negatives, FP to False Positives, and FN to False Negatives. Next, we aim at defining how the TP, TN, FP, and FN are computed. These variables summarise the outcome of the system (its predictions) with respect to the inputs and their labels. That is, for a specific sample, depending on the prediction given by the system and the ground truth label of the sample, the prediction is considered a TP, a TN, a FP, or FN. The following conditions are applied to each sample to decide its category:

- TP: an input stack labeled as fall is predicted as fall.
- TN: an input stack labeled as ”no fall” is predicted as ”no fall”.
- FP: an input stack labeled as ”no fall” is predicted as fall.
- FN: an input stack labeled as fall is predicted as ”no fall”.

These categories are given taking into account that the fall class is the positive class and the ”no fall” class the negative class. Hence, the value of TP used in Equations 3.4, 3.5, and 3.6, for instance, is computed as the number of samples that triggered a TP, i.e., that were labeled as a fall and the system predicted that were falls.

Furthermore, for a robust evaluation, we applied a 5-fold cross-validation for the URFD and the FDD datasets and a leave-one-out cross-validation for the Multicam dataset following Rougier et al. (2011). For the first case (URFD and FDD), we separately extracted the fall and ”no fall” inputs and evenly divided each of them into five parts or folds, thus, having 5 folds per class. With this, 5 models were trained: in each iteration, 4 data folds were used for training (mixing a fold of each class) and the remaining one (again, combining both classes) for evaluation. Thus, 5 evaluations (with each fold) were done and the final result is given as the average of them, with the standard deviation. For the case of the Multicam dataset, we divided the dataset into 8 folds, each of them containing inputs obtained from one of the eight cameras. The evaluation is done like in the case of the URFD and the FDD datasets but in 8 iterations. In addition, for all the datasets, in order to balance the distribution of samples between classes in the training fold, the dominating class (”no fall” class) was undersampled (by random sampling without replacement) to match the number of samples of the fall class. Before the undersampling step, the validation set is created using a stratified sampling. The test set was not modified.

3.3.3 **Fall classification experiments**

The experiments were done with the three datasets using the settings presented in Section 3.2.2 and the evaluation of Section 3.3.2. Tables 3.3, 3.4, and 3.5 show the results for the URFD, the Multicam, and the FDD datasets, respectively, and compare these results with those presented in the literature. Depending on the metrics given by other authors, different metrics are presented in the aforementioned tables. As stressed in Section 3.3.2, the sensitivity and specificity metrics are preferred for the comparison with other authors, as

Dataset	*Epochs*	*Learning rate*	*Mini−batch size*	w_0	*Activation*	*Validation used*
URFD	3000	$10e^5$	64	1	ReLU+BN	No
Multicam	6000	$10e^3$	*Full batch*	1	ReLU+BN	Yes (100 samples)
FDD	3000	$10e^4$	1024	2	ReLU+BN	No

Table 3.2: Hyperparameters for the experiments of fall classification with the URFD, the Multicam, and the FDD datasets. "Full batch" refers to do batch training instead of mini-batch training. The validation set is built with a stratified sampling of the amount of samples referred in the cell.

Proposal	*Sensitivity*	*Specificity*	*Accuracy*	*FAR*	*MDR*
Zerrouki and Houacine (2018)	-	-	96.88%	-	-
Ours	100.00% (±0.00)	99.15% (±0.30)	99.20% (±0.29)	0.85% (±0.30)	0.00% (±0.00)

Table 3.3: Comparison between the state-of-the-art approaches and ours for the vision-based fall classification task with the URFD dataset. For our experiments, the average result is presented on top and the standard deviation is shown below.

the comparison is fairer. For training, the hyperparameters of Table 3.2 were used.

Harrou et al. (2016a) and Zerrouki et al. (2016) used both URFD and FDD, but it is not clear how they were used, as they do not specify which dataset they are using to obtain their results. Thus, they are not included in the comparative table. Harrou et al. (2016a) reported their results using the False Alarm Rate (FAR, or False Positive Rate) and Missed Detection Rate (MDR, or False Negative Rate) metrics, obtaining a FAR of 7.54% and a MDR of 2.00%. We computed those metrics with the URFD obtaining a FAR of

Proposal	*Sensitivity*	*Specificity*	*Accuracy*
Wang et al. (2016c)	89.20%	90.30%	-
Wang et al. (2016a)	93.70%	92.00%	-
Ours	99.10% (±1.72)	99.55% (±0.30)	99.54% (±0.28)

Table 3.4: Comparison between the state-of-the-art approaches and ours for the vision-based fall classification task with the Multicam dataset. For our experiments, the average result and the standard deviation are shown.

Proposal	*Sensitivity*	*Specificity*	*Accuracy*
Charfi et al. (2012)	98.00%	99.60%	-
Zerrouki and Houacine (2018)	-	-	97.02%
Ours	99.50% (±0.49)	97.90% (±0.55)	97.95% (±0.52)

Table 3.5: Comparison between the state-of-the-art approaches and ours for the vision-based fall classification task with the FDD dataset. For our experiments, the average result and the standard deviation are shown.

0.85% (±0.30) and a MDR of 0.00% (±0.00). Zerrouki et al. (2016) reported the sensitivity and specificity, they got a 98.0% and a 89.40%, respectively, while we got 100.00% (±0.00) and a 99.15% (±0.30). Besides, Zerrouki and Houacine (2018) only provided the accuracy for their results, 96.88% compared with our 99.20% (±0.29). Nonetheless, due to the imbalance nature of the dataset and following the accuracy paradox, both results cannot be compared. That is, a higher accuracy does not imply a higher predictive power. For instance, a system predicting "no fall" for all the samples would obtain a 93.45% of accuracy in the URFD dataset without detecting any of the existing falls. For that reason, as explained in Section 3.3.2, the sensitivity and specificity metrics were chosen.

30 stacked frames were used by Wang et al. (2016c) and Wang et al. (2016a) to make a prediction, while only 10 were used in this work. In fact, our system achieves a sensitivity of 99.10% and a specificity of 99.55%, while Wang et al. (2016c) obtained 89.20% and 90.30%, and Wang et al. (2016a) 93.70% and 92.00%, respectively.

The works done by Auvinet et al. (2010) and Rougier et al. (2011) are not included in the comparative table of the Multicam dataset due to the different evaluation methodology they apply[1]. In their criterion, they set a variable t_{fall} with the frame number where the fall starts, dividing the video in the part previous to the fall and the one after the starting point, containing both the fall and any event after that. A fall detected after t_{fall} by their system is considered a TP, otherwise, if they miss it, they get a FN. Taking into account the part previous to the fall, if the fall is predicted before t_{fall}

[1] http://www.iro.umontreal.ca/ labimage/Dataset/technicalReport.pdf)

they classify it as a FP, otherwise, they consider it a TN. We believe this methodology is not suitable for the comparison with other authors and, thus, we did not consider its use.

Moreover, in the work done by Auvinet et al. (2010), they detected lying-on-the-floor positions rather than falls. With this approach, they got sensitivity and specificity values up to **99.70%**. Rougier et al. (2011) obtained an accuracy of **99.70%** evaluating the falls at video-level rather than at a stack of frames level, as in our case. An issue that may arise from this approach is that it requires a specific data framing, using videos as input rather than a continuous stream of images, which is not the ideal case for real-world deployments. In addition, they used a 3D vision system, providing them with more information compared to a 2D system. However, our system performs comparably even with less information.

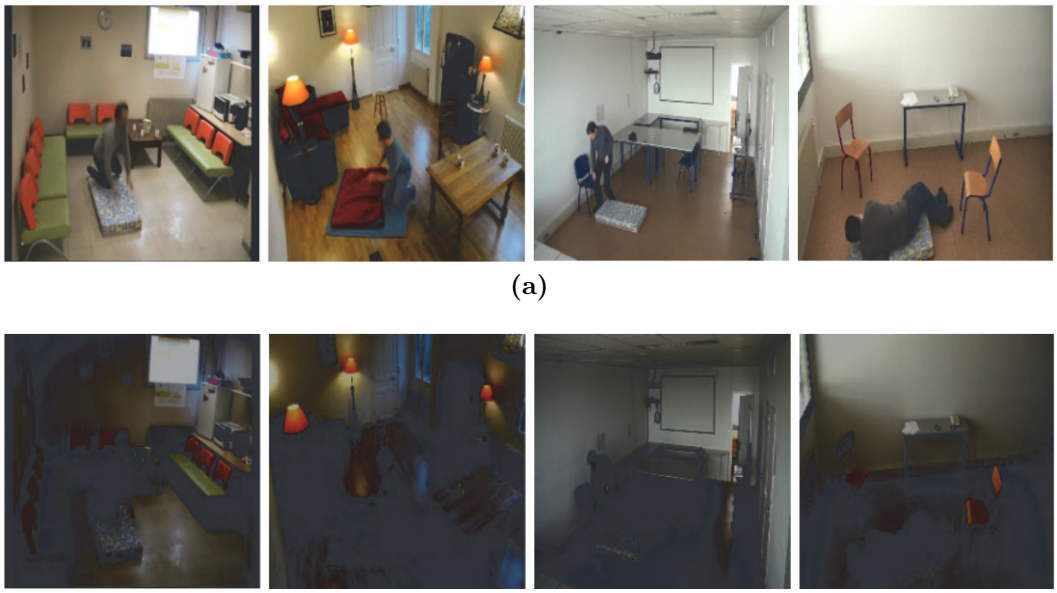
Both Harrou et al. (2016a) and Zerrouki et al. (2016), as in the case of URFD, mentioned the combined use of FDD and URFD, but only provided one result, making the comparison unclear. Harrou et al. (2016a) reported a FAR of **7.54%** and a MDR of **3.00%**, while we got **2.10%** (±0.55) and **0.50%** (±0.49) for those metrics. Zerrouki et al. (2016) presented a sensitivity of **98.00%** and a specificity of **89.40%**, meanwhile we obtained **99.50%** (±0.49) and **97.90%** (±0.55), respectively. Zerrouki and Houacine (2017) showed an accuracy of **97.02%**, and in our case, **97.95%** (±0.52). As it was mentioned in the URFD case, the evaluation using pure accuracy is misleading, so both proposals cannot be fairly compared. Charfi et al. (2012) used ad-hoc hand-tuned thresholds in their system to detect falls, obtaining very high results: **98.00%** of sensitivity and **99.60%** of specificity. However, it is not clear how this system would perform in other datasets, as the thresholds were hand-tuned for the FDD dataset.

Experiments with different lighting conditions. The three public datasets present a static and controlled lighting conditions, suitable for artificial vision tasks. Nevertheless, in real-world scenarios, the lighting is usually changing, for instance, with the sunlight coming through the window or with a lamp switching on and off the lighting changes dynamically, or at night

when the scenario is darkened there is a static change. To test the generalisation capability of our system under those circumstances, we propose the following two experiments: (i) the use of static lighting, in which the lighting conditions simulate night-like scenarios and (ii) the use of dynamic lighting, adding a dynamic lighting that smoothly increases its intensity from frame to frame until reaching a specific intensity, decreasing afterwards until reaching the original lighting condition. These experiments are carried out using the FDD dataset, which shows a variety of scenarios.

For the static lighting experiment, a value of 100 is extracted for every pixel in every channel of each image so that they get darkened as if it was night (values lower than 0 are set to 0). Figure 3.5 shows samples of the original images and their darkened versions. The dataset was divided in a $80:20$ ratio into two sets: training and test. The class distribution in the training set was balanced. At the beginning, the original images were used for training and the evaluation was done in a darkened test set. The model was trained using the same configuration as in the previous experiments with the FDD dataset. A sensitivity of 45.85% and a specificity of 98.67% were obtained from this. The result is coherent with that fact that falls are difficult to detect when the actor is close to the floor, exactly the place where the area is very dark in the image and the silhouette of the subject falling is not distinguishable. Therefore, even a lying-on-the-floor position would be difficult to detect. Then, the darkened images are used in both training and test, using the same training configuration. The sensitivity went up to 87.12%, while the specificity decreased a bit, to 94.92%. The difference with respect to the original experiment is not that large, specially taking into account that the training parameters were not adjusted for these new images.

For the dynamic lighting experiment, the aim was to simulate non-stable lighting condition, unlike a lab environment's one. For example, a lamp may be switch on/off in the background, generating displacement vectors in the OF algorithm. To reproduce this effect in the FDD dataset, a progressive change of lighting was added. This effect takes 32 frames to light up and fade. As the videos of the FDD dataset were recorded at 25 Frames Per Second (FPS), the lighting lasts for about 1.3 seconds. To create it, for each pixel in a frame, the



(a)

(b)

Figure 3.5: FDD dataset images: (a) original and (b) darkened images. From the work of Nunez-Marcos et al. (2017).

intensity of the pixel was multiplied with a sinusoidal function. From frame to frame, the intensity would follow the sinusoidal function's change pace, allowing for a smooth transition. This modification was done once per video in its first 32 frames. Figure 3.6 shows samples of original images, their OF images, and the modified versions of both the original and the OF images. The first experiment was once again performed, training with the original images and evaluating using the new images to test how the system would react to this change. The configuration for the experiment was the same as for the darkened images. The resulting sensitivity was 28.04% and the specificity 96.35%. It is natural that the network gets confused because of the amount of displacement vectors generated by the dynamic lighting, as the network has only been trained with the displacement vectors of a moving person. Next, to see if the system could adapt to these changes if it was properly trained with the new images, the following experiment proposed to both train and test using those images (using the same training configuration). The result was a sensitivity of 90.82% and specificity of 98.40%.

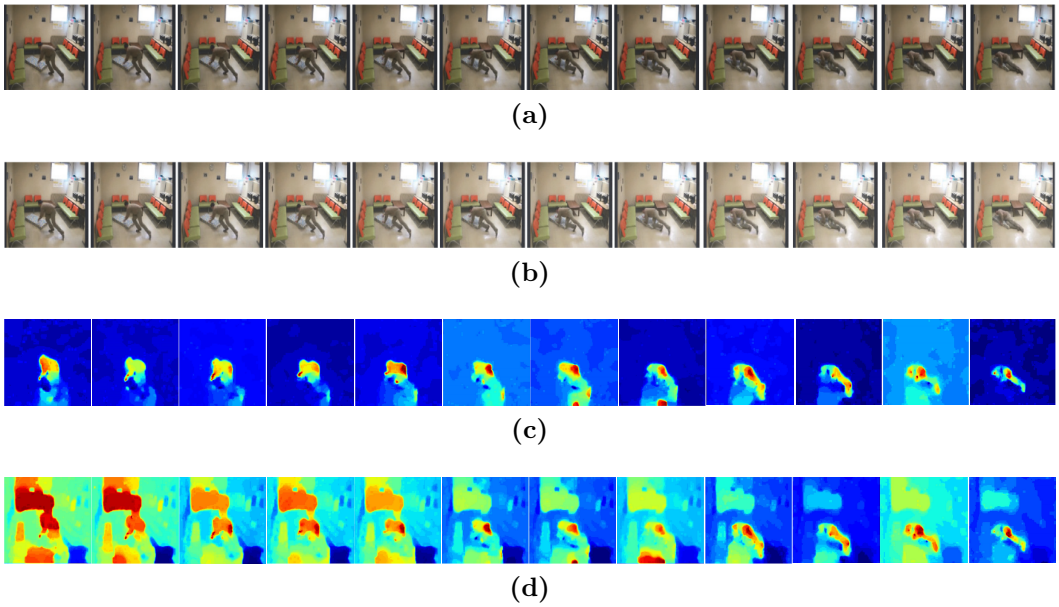


Figure 3.6: FDD dataset images: (a) original RGB images, (b) RGB images with the artificial lighting added, (c) original OF images, and (d) OF images computed from the images with the artificial lighting added. From the work of Nunez-Marcos et al. (2017).

Experiment with combined datasets. One of the characteristics of the model proposed is that it should be able to generalise to several scenarios, i.e., given unseen fall sequences, the detector should be able to correctly classify them no matter the scenario where these are given. Previous experiments considered each dataset individually, possibly taking into account special features in each of them. In contrast, in this part, the three datasets were combined into a single one to avoid any singular feature guiding the training. For that, in order to give an equal weight to each dataset, all the datasets were resampled (without replacement) to match the number of samples of the smallest dataset's minor class, URFD's fall class. With this change, the three datasets had the same relevance (amount of samples) and both classes (fall and "no fall") were balanced. More concretely, each dataset had 1200 samples, 600 fall and 600 "no fall" samples.

For the evaluation process, a 5-fold cross-validation was applied, dividing the dataset into 5 groups, each group containing the same amount of samples of both classes. A validation set was created using 200 samples (again, bal-

Proposal	*Sensitivity/Recall*	*Specificity*
URFD+Multicam+FDD	96.33% (±4.28)	97.17% (±2.82)
URFD	98.50% (±2.60)	97.17% (±2.82)
Multicam	92.33% (±8.29)	99.50% (±0.41)
FDD	98.17% (±2.13)	94.83% (±3.55)

Table 3.6: Experiment with the system trained with the three datasets combined (URFD, Multicam, and FDD). The results are provided over each individual test set and also in the combined test set.

anced in both classes and datasets) in each iteration of the cross-validation for the early stopping, which had a 100 epoch patience. The network was trained for 1000 epochs with a learning rate of $10e^{-3}$, a batch size of 2048, a class weight of 2, and using the ReLU activation with Batch Normalisation. The results are shown in Table 3.6, presented for each individual dataset's test set (created from the samples of each dataset in the evaluation fold) and also for the combined test set.

We believe that the obtained results support our claim of having a generic fall detector system, mainly based on two reasons. The first one is that our system has been evaluated on three public datasets, obtaining state-of-the-art results on all three of them. To the best of our knowledge, our system was the first one achieving such results on those three datasets, with each one having different scenarios, types of falls, and different characteristics. For instance, when annotating the beginning and ending of falls, the FDD dataset differs from the other two, as they consider that a fall starts when a person is inclined around 45 degrees with respect to the floor (while falling). In addition, this dataset considers more scenarios than the other two datasets. Another specific characteristic between these datasets is the amount of points of perspectives from which falls are recorded in the Multicam dataset. It has eight cameras recording the same event at the same time, providing novel viewpoints of each fall and, thus, a higher variance of the fall class. However, our system is still able to achieve results that equal the best state-of-the-art algorithms. This can be considered a solid proof of the generality of the fall detector and a promising cue towards a motion detector capable of generalising to different

motion patterns.

Due to having to train an MLP classifier each time per dataset, it can be argued that the detector requires to be trained to fit each dataset. Nonetheless, the experiment combining the three datasets of Table 3.6 aimed at refuting that reasoning. The combination of datasets was done both for training and evaluation, taking into account the differences between the datasets. It can be seen that during the evaluation phase, the result over the test set that combines the three datasets obtains very high detection rates (an average sensitivity of **96.33%** and an average specificity of **97.17%**). When observing the results obtained in the test sets of each individual datasets it is clear that the results remain high, except for the case of the Multicam dataset, whose performance (at least for the fall class) is slightly lower (a sensitivity of **92.33%** and a specificity of **99.50%**). The reason may well be that when generating the combined dataset, many stacks from specific perspectives are discarded to keep an equal importance weight for each dataset. In that process, many of the stacks may have been helpful for our network to learn the different perspectives that falls may have. Hence, we conclude that having more stacks of different perspectives in the training phase may have alleviated this issue. Still, the results back up our generality claim, since the system is able to generalise to unseen samples showing high detection rates. In fact, under more realistic conditions, in real-world scenarios, the system may get poor results unless trained in real-world data, obtaining a better generalisation and better motion-related features for action recognition.

The generalisation capabilities that are defended in this experiment come from two decisions: (i) the use of OF images, which are suitable for modeling the motion of consecutive frames, and (ii) the three-step training phase, in which the network learnt generic motion-related features thanks to the pre-training in large datasets such as UCF101.

3.4 **Summary and Conclusions**

In this chapter, the fall classification task has been presented, with a brief analysis of the state-of-the-art approaches at the time this prototype was

published (Nunez-Marcos et al., 2017). The aim of studying this task was to employ it as a use case to test the motion detector prototype that was designed and implemented. The latter was built using a DL framework and the motion of the sequences was encoded using OF images. The task was simple compared to the HAR or the learning of several motion classes, but useful for validating the methodology, the network, and the data processing.

As a conclusion for the chapter, we believe the motion detector designed and implemented in this chapter is a suitable prototype for the use case of fall classification. Nonetheless, with the analysis and the results obtained in this work, we believed the prototype still required some improvements for being valid for actions. Apart from the obvious extension to multi-class problems, the short-term motion modeled by this system might not be enough to model actions, which are more complex. Future chapters will introduce changes to learn long-term motion. The basis of encoding short-term motion, however, is a good starting point, as hierarchically encoding the motion seems to be a promising approach to learning actions based on recent works presented in Section 2.1.

In addition, we also saw that the results obtained from a sliding window approach can be misleading. The temporal overlap of the input stacks and the sliding window presented in Section 3.2.1 result in inputs that are very similar due to their proximity or overlapping in videos. Therefore, training and evaluating using these stacks can be seen as a kind of overfitting to the test set. We take this issue into account for future systems.

The road of life twists and turns and no two directions are ever the same. Yet our lessons come from the journey, not the destination.

Don Williams, Jr.

CHAPTER

4

Active object classification

EGOCENTRIC actions are mainly based on interactions with objects. The latter represents one of the most important aspects of these type of actions, as the sole clue of identifying which objects are being manipulated gives humans important hints towards distinguishing them. The very basis of Bag-of-Objects approaches (Pirsiavash and Ramanan, 2012) (Matsuo et al., 2014) are the presence of objects, not even requiring specific features of appearance or motion in the earliest works. Therefore, as one the the pillars of the EAR, this chapter will focus on various proposals for leveraging or modeling the objects present in an egocentric video and recognising active objects. Due to the interest in comparing the performance of the proposed methods with those of the action classification literature, each proposal of the chapter is going to be used for action classification too, even though the active object classification is still the main point of the chapter. Moreover, various conclusions can be extracted from this application, summarised at the end of the chapter.

The chapter is divided into various sections, starting from Section 4.1, which presents the GTEA family of datasets used across the rest of the dis-

sertation and Section 4.2, which shows the first experiments with appearance based features following a methodology similar to that of Chapter 3. The Faster R-CNN to explicitly learn objects is shown in Section 4.3, followed by two proposals that leverage the outcome of this part: Section 4.4, that presents the Bag of Objects Matrix (BOM) data structure, and Section 4.5, that presents another method to exploit the learnt objects' information to build an attention mechanism. Finally, the conclusions of the chapter are presented in Section 4.6.

4.1 GTEA action classification datasets

The Georgia Tech Egocentric Activity (GTEA) datasets are a family of egocentric datasets developed by the Georgia Institute of Technology, having four versions when this work was developed: the GTEA dataset (Fathi et al., 2011b) (Li et al., 2015), the GTEA Gaze dataset (Fathi et al., 2012), the GTEA Gaze+ dataset (Fathi et al., 2012) (Li et al., 2015), and the newest version at that moment, the Extended GTEA (EGTEA) Gaze+ dataset (Li et al., 2018). The last two were employed for the experiments carried out in this chapter and the following one. Therefore, this section aims at providing a presentation to these datasets and any necessary detail concerning them.

These datasets were collected at the Georgia Tech's AwareHome, a living lab within the Aware Home Research Initiative (AHRI). The latter is an interdisciplinary research facility since 1998, mainly used for health and well-being related research. The AwareHome is a 5040 square foot facility that provides a realistic home scenario. Specifically, these datasets were recorded within the kitchen of this living lab and they consist in several videos, in which some subjects were recorded preparing meals following pre-defined recipes.

The GTEA Gaze+ dataset has 6 persons preparing seven different meals: an american breakfast, a pizza, a snack, a greek salad, a pasta salad, a turkey sandwich, and a cheese burger. It is important to note that the preparation of each meal is considered an activity while all the steps or subobjectives required to prepare it are the actions: "taking a vegetable", "turning

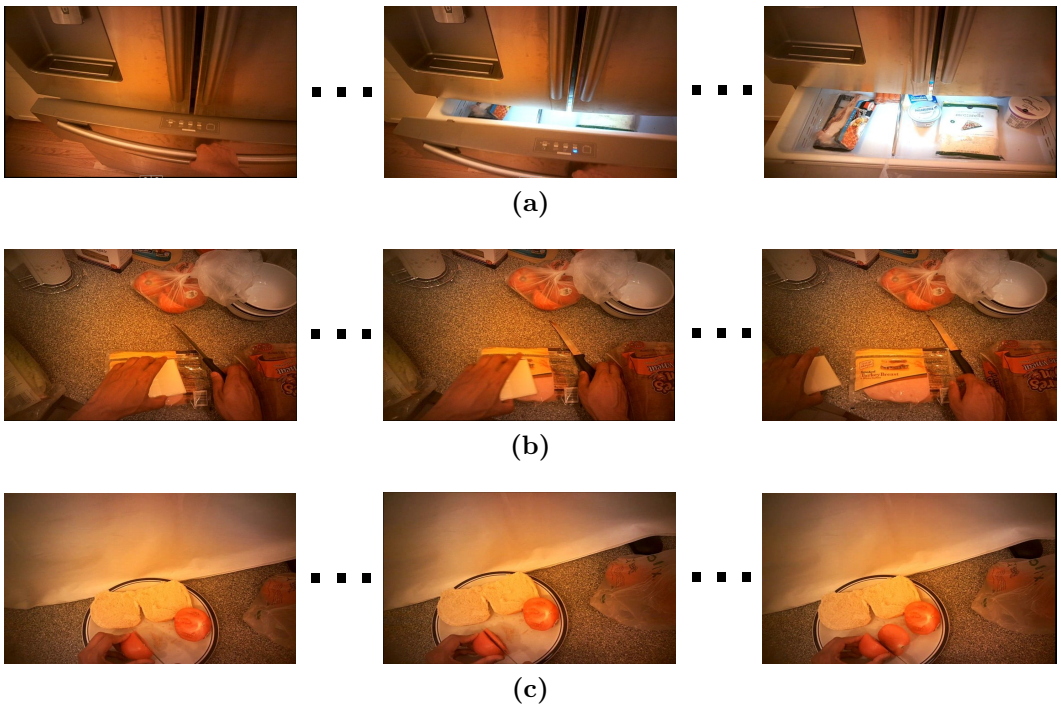


Figure 4.1: GTEA Gaze+ action examples sampled from the dataset: (a) "open freezer" action, (b) "put knife" action, and (c) "cut tomato" action.

on the heat", and so on. Concerning the recording, they employed SMI eye-tracking glasses to record HD videos with 24 FPS and used the ELAN software for the annotations.

Each video contains several action annotations. These are composed of start and end frames, as well as verb and object annotations. The combination of a verb and an object can express an action when combined. A verb is simply a name to tag the type of motion of the action (e.g., "cut", "take", "put", and so forth). The object, or more specifically, the active object, is the element that is interacted with, being the motion the type of interaction that is given with the object. In fact, in these datasets, there are not actions without objects (such as "walk" or "jump"), so it is assumed that there will always be a verb and an object. In addition, verbs and objects are presented in a stemmed way rather than in a legible or natural way. For instance, the label comes in the form "take knife" and not "take that big knife". Concerning the visual part, the clips used to learn actions are trimmed from their respective

	Subjects					
	Ahmad	*Alireza*	*Carlos*	*Rahul*	*Shaghayegh*	*Yin*
Actions	44	44	44	44	35	40
Sequences	504	298	475	361	146	240
Seq./class	11.45 (±11.88)	6.77 (±8.25)	10.79 (±10.85)	8.20 (±11.18)	4.17 (±4.02)	6.00 (±5.52)

Table 4.1: GTEA Gaze+ distribution per fold (subject). A subject's fold contains all their data, i.e., all the videos in which the subject was recorded performing actions.

videos using the given beginning and ending annotations. A few samples of these trimmed videos from the GTEA Gaze+ dataset can be seen in Figure 4.1.

The evaluation strategy defined for the GTEA Gaze+ dataset by its authors is the leave-one-subject-out validation. Taking into account that 6 subjects (Ahmad, Alireza, Carlos, Rahul, Shaghayegh, and Yin) are performing the actions and activities, each subject's clips (in which the subject performs actions) compose a fold. However, as specified by the authors, Shaghayegh's and Yin's videos (two of the subjects performing activities) are always part of the training set and, thus, are not used for evaluation. The reason for this is that they do not perform every activity and action in the dataset. Taking all this into account, four iterations or train/test splits are used within this leave-one-subject-out cross validation.

Table 4.1 summarises the distribution and amount of data per fold. Without loss of generality, when mentioning Ahmad's fold we are referring to his data; in contrast, Ahmad's split is the one in which Ahmad's data is used for evaluation and the remaining subjects' data for training. This distinction is important, as we may be referring to both terms throughout the chapter. In the table, it can be seen that Shaghayegh's and Yin's folds have less actions than the rest, as mentioned earlier in the evaluation, and that their number of sequences (action instances) is quite low compared to the rest, although Alireza has remarkably fewer number of samples despite having all the 44 actions. Concerning the average of sequences, it can be seen that both Shaghayegh's and Yin's folds have less instances of each action. In contrast, Ahmad's and Rahul's folds are significantly larger than the rest and it is possible that,

Split	*Training videos*	*Test videos*
Split 1	8299	2022
Split 2	8299	2022
Split 3	8300	2021

Table 4.2: EGTEA Gaze+ distribution per split.

whenever their folds are used for testing and not for training, the performance drops due to the lower number of training samples.

The EGTEA Gaze+ dataset is an updated version of the GTEA family that was launched in 2017. This new dataset subsumes GTEA Gaze+, i.e., its data is contained within EGTEA Gaze+. It comes with HD videos (1280×960), audios, gaze tracking data, frame-level action annotations, and pixel-level hand masks at sampled frames, having $15,176$ hand masks from $13,847$ frames. In total, it has 28 hours of video with 32 subjects and $10,325$ instances of fine-grained actions. As with the GTEA Gaze+, EGTEA Gaze+ comes with several action annotations per video in the form of start frame, end frame, verb, object, and action label. Moreover, rather than having separate subject data for the cross-validation strategy, this dataset has three official train and test splits. In each split, the test set follows a distribution similar to that of the training set. The number of clips per split are summarised in Table 4.2. Furthermore, a few samples of the dataset can be seen in Figure 4.2.

For the evaluation of both datasets, and for both active object and action classification, the accuracy and the macro-F1 metrics are used. The accuracy, as seen in Equation 4.1, takes into account the TPs, TNs, FPs, and FNs. These values are determined by the evaluation protocol established for an algorithm and, hence, they will be defined in each section. In general terms, each sample's ground truth and its prediction are used to compute whether the actual prediction is a TP, TN, FP, or FN. So, the accuracy refers to the ability of a smart system to correctly predict both positive and negative samples (TPs and TNs).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

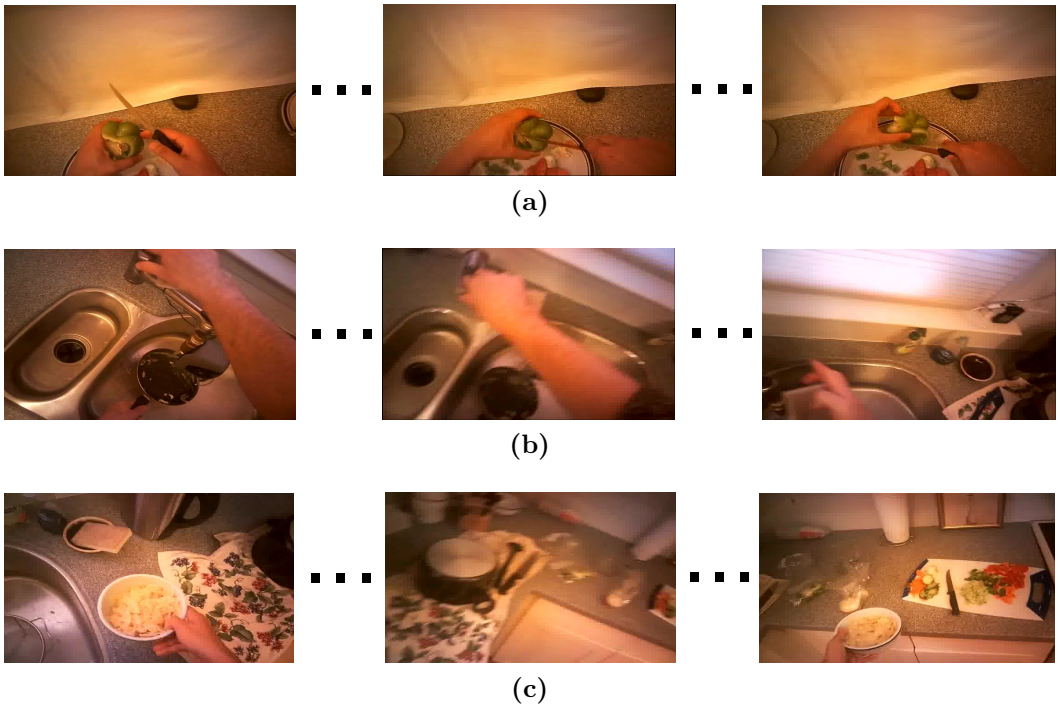


Figure 4.2: EGTEA Gaze+ action examples sampled from the dataset: (a) "cut bell pepper" action, (b) "wash pan" action, and (c) "move bowl" action.

The F1 is a metric used to summarise two commonly used metrics: the precision and the recall. The precision (see Equation 4.2) of a class measures the percentage of correct predictions done out of the total predictions and, hence, it is desirable to have fewer FPs to increase its value. Intuitively, the recall (see Equation 4.3) represents the ability of a system to predict the possible positive samples, penalised by the FNs, i.e., when the system is not able to correctly classify a positive sample. To combine both metrics, instead of the arithmetic mean, the harmonic mean is employed for the F1, illustrated in Equation 4.4. Therefore, the F1 will always be between the values of precision and recall. When the F1 per class is averaged using the arithmetic mean, the macro-F1 is obtained.

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{4.4}$$

The macro-F1 metric will be an interesting alternative to the accuracy due to the accuracy paradox. That is, when comparing two different models, it may happen that it is better to select the model with the lowest accuracy due to its higher predictive power. This is often the case with imbalanced class distributions such as those of the datasets of the GTEA family, in which the class distribution is far from following a uniform distribution. In these cases, the accuracy of dominating classes (those with the highest amount of samples) can be very high in comparison with other classes, in which the accuracy can be poor. Due to how the accuracy is computed, taking into account the total number of samples without doing an evaluation per class, the accuracy can show a high value due to the high result of the dominating classes. The minor classes do not contribute as much and, thus, their poor performance does not penalise the accuracy that much. This phenomenon does not occur with the macro-F1, in which this imbalanced performance will be reflected.

4.2 Object classification network: the SpatialNet

The *SpatialNet*, a CNN used to learn the appearance of clips of videos, was employed and popularised by the work of Simonyan and Zisserman (2014a) in their two-stream approach. Later, it was named *AppearanceNet* (Ma et al., 2016), although both works referred to the same type of network with the same objective: learning the appearance, the visual content of the videos, i.e., the objects and the scene. Because, apart from the physical place where actions occur, objects are also contained within the visual features of what it is called ”the scene”. Depending on the task, features focused on objects, the

scene, or other cues can be obtained, but in it essence, the visual features are the ones that are exploited.

Following the motion prototype presented in Chapter 3, our first approach also used a single-stream CNN structure but with a different objective. This time, we aimed at predicting objects and actions using visual features of videos. That is, the implicit learning of objects and the scene was the main driver of the learning and the classification of active objects and actions.

4.2.1 Architecture

For the backbone network used to extract RGB features, we followed the work done by Ma et al. (2016) and used a CNN-M-2048 network (Chatfield et al., 2014). In contrast to the motion detector, for the visual features, RGB images are not stacked and fed as input to the network. Instead, following the approach of Ma et al. (2016), if the appearance stream takes as input single frames of videos, then each of the frames can vote for a label. That is, the architecture takes a sequence of inputs, each one being a frame $X \in \mathbb{R}^{H \times W \times 3}$, where W and H are the width and height of the image and 3 the number of channels; instead of having a single probability distribution as output, each frame of the sequence has its own output distribution. We may see this as if each of the inputs of the sequence would have its own vote. By averaging these votes, a new probability distribution is obtained, representing the vote for the sequence. Figure 4.3 illustrates this process.

4.2.2 Experiments

Two sets of experiments are proposed: in the first one, the objective is to classify active objects, while, in the second one, actions. For both sets, the same architecture and hyper-parameters are employed. 10 timesteps are used as input, being each frame normalised by subtracting the Imagenet mean. The networks are trained for 200 epochs with early stopping with a patience of 20 epochs. A learning rate of 0.0001 is used, alongside a mini-batch size of 32. The first three convolutional layers' weights are frozen. Class weights

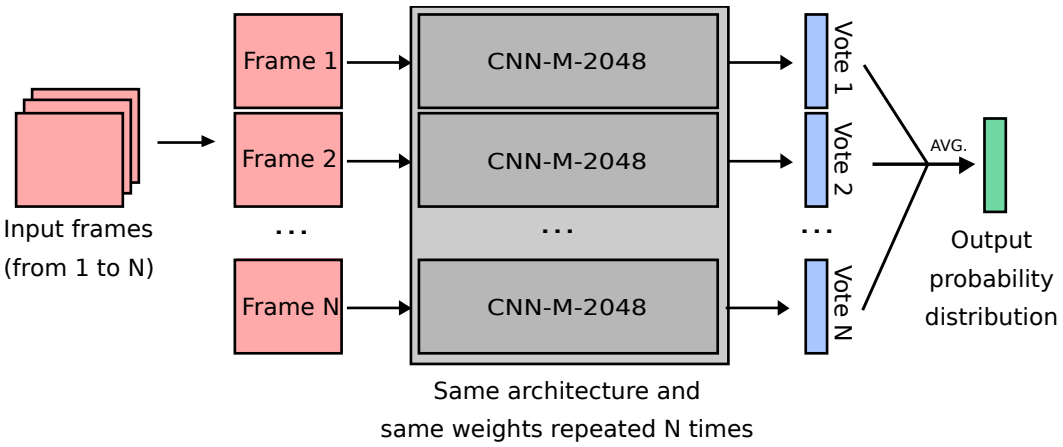


Figure 4.3: A CNN-M-2048 network (Chatfield et al., 2014) is repeated N times, each of them having the same architecture and weights. Each copy extracts visual features from each frame and outputs a probability distribution (a vote). All the votes are averaged to obtain the final probability distribution. For the training, only a set of weights is trained, as if there would be only a single CNN-M-2048.

generated from the training set distribution were not employed for the learning phase.

The GTEA Gaze+ dataset is used for the evaluation, applying the leave-one-subject-out evaluation. For this problem, we defined the following TP, FP, TN, and TN for the accuracy and F1 metrics:

- TP: for the class i, if a sequence of images is labeled with class i and the class i is the one with the highest confidence after the average voting.
- FP: for the class i, if a sequence of images is labeled with class j and the class i is the one with the highest confidence after the average voting, being j any class different to i.
- TN: for the class i, if a sequence of images is labeled with class j and the class k is the one with the highest confidence after the average voting, being j and k any classes different to i and being possible that $j = k$.
- FN: for the class i, if a sequence of images is labeled with class i and the class j is the one with the highest confidence after the average voting, being j any class different to i.

Active object classification									
Average		*Ahmad*		*Alireza*		*Carlos*		*Rahul*	
Accuracy	*F1*	*Accuracy*	*F1*	*Accuracy*	*F1*	*Accuracy*	*F1*	*Accuracy*	*F1*
58.92%	47.95	53.90%	47.00	63.97%	54.35	52.07%	43.16	65.74%	47.29
(±6.05)	(±4.14)	(±0.75)	(±0.18)	(±0.82)	(±0.97)	(±0.26)	(±1.02)	(±0.91)	(±1.14)

Table 4.3: Results of the experiments with our SpatialNet and the GTEA Gaze+ dataset for active object classification. The results are given per subject and as the average of all of them,. Each of the subject's experiments are run three times and averaged. This mean is provided in the top row, while the standard deviation is shown at the bottom.

Action classification									
Average		*Ahmad*		*Alireza*		*Carlos*		*Rahul*	
Accuracy	*F1*	*Accuracy*	*F1*	*Accuracy*	*F1*	*Accuracy*	*F1*	*Accuracy*	*F1*
46.55%	36.11	38.89%	32.67	52.19%	41.9	38.88%	31.84	52.63%	38.03
(±7.84)	(±4.44)	(±1.27)	(±2.07)	(±0.55)	(±0.55)	(±0.43)	(±0.38)	(±0.60)	(±2.55)

Table 4.4: Results of the experiments with our SpatialNet and the GTEA Gaze+ dataset for action classification. The results are given per subject and as the average of all of them. Each of the subject's experiments are run three times and averaged. This mean is provided in the top row, while the standard deviation is shown at the bottom.

The results are shown in Tables 4.3 (for active object classification) and 4.4 (for action classification). These are presented per fold (subject) and as the average of all the folds. Each fold is run three times and the average of those three experiments are provided alongside the standard deviation.

These results can be seen as baselines. Other authors such as Ma et al. (2016) also followed this type of strategy, i.e., having a CNN extracting visual features from single frames and then averaging the votes. However, they also included motion features as an extra branch. The vote emitted in a single timestep would take into account both the visual appearance of the frame and the local motion (taking the previous and next frames to create an OF stack). In fact, we could add multiple streams of information to improve the vote per timestep. For the moment, we will keep using only the visual stream.

In contrast to the fall classification task of Chapter 3, the results of Tables 4.3 and 4.4 are much lower. This was expected taken into account (i) the num-

ber of classes is much higher (due to the change from binary to multi-class classification); (ii) the change from exocentric to egocentric actions increasing the difficulty of the classification problem as discussed in the literature (see Chapter 2); and (iii) the fact that understanding actions or being able to classify active objects is much more difficult tasks than classifying falls. The latter can be explained by the fact that falls are simple to represent events, having a clear start and end, and being easier to recognise by the analysis of the motion of the subject involved. In contrast, actions often contain interactions with objects, motion patterns can have more variations, and so on. Thus, the analysis of the involved objects and motion patterns is crucial to fully understand actions. For active objects, understanding which among all the objects present in a scene is the active one can be difficult without understanding the dominant motion pattern, i.e., how subjects interact with objects. Moreover, for both active object and action classification, the variance of motion and object patterns is higher than that of fall events, having more possible configurations for each combination of verb and object. For example, the motion associated to "open" is not the same when performed with a fridge or with a bottle. Even using the same object, the motion can be different, as in the case of opening a cupboard downward or leftward, for instance. Object appearance can also be different accompanied with the same verb tag, as in the case of cutting vegetable in slices or in small pieces. For all this, the results are not as high as those of the fall classification experiments.

Observing the results per subject, it is noticeable that there is a significant difference among splits. The class distribution in each split and the variance introduced by each subject can be crucial on this. For example, if a subject follows a similar pattern cutting vegetables, both in the training set and test set, then it may be easier to recognise the action of cutting vegetables. In contrast, having different poses, patterns, or even tools to cut (e.g., knives of different shapes or colours) increases the difficulty to learn actions. Therefore, the results of the test set are expected to be lower if not all the variance can be explained in the training set or there is not enough data to generalise correctly. For objects, first, the variance of objects can pose a problem. For instance, a whole vegetable compared to a sliced vegetable will look different,

even more compared with a minced vegetable, divided in small pieces. The second problem is the occlusion with hands and the cluttered background. In the first case, while grabbing an object, hands may be occluding the object, difficulting the recognision. In the second case, having multiple elements in the background, identifying the most relevant one or the one that is being manipulated (the active object) can be challenging. As in the action case, if all this variance is not explained in the training set or if the system is not able to correctly generalise, there can be huge differences between data and subject splits.

Table 4.5 compares the obtained results with some approaches of the literature in the GTEA Gaze+ dataset. For the active object classification, our results are not that far from approaches such as that of Ma et al. (2016). However, the results reported by them are given by a Spatial CNN using single inputs instead of a sequence of frames using our time distributed Spatial CNN. In fact, they are superior to us even with this handicap. This may be due to the fact that their Spatial CNN exploits hand information to localise objects within images instead of using the whole frame. Moreover, after jointly training their full system for actions (the spatial and motion CNNs are jointly trained), the object classification result improves to **74.34%** of accuracy. Focusing the attention of the learning in specific regions of images may alleviate background clutter issues, as well as removing not desirable information in the learning phase. However, there is a dependence on a hand detection system or a similar method to localise objects.

For the action classification, as expected, we obtained some of the lowest results compared to other approaches. Our baseline, however, obtains such a low result given that (i) we have not adopted any attention mechanism, (ii) the temporal modeling of actions was very naive, (iii) we have not included any extra information (streams of information), and (iv) that using the best possible CNN was out of the scope of these experiments. Meanwhile, Wang et al. (2013) and Wang and Schmid (2013) proposed the Dense Trajectories (DT) and the IDT to model actions. Their method is based on trajectory recognition, i.e., feature points are densely sampled from frames and, then, the tracking is performed within the spam of L frames using a median filtering in a dense OF

Active object classification		
Approach	*Accuracy*	*F*1
Spatial stream (Ma et al., 2016)	61.87%	–
Spatial stream (after joint training) (Ma et al., 2016)	**74.34%**	–
Time Distributed Spatial CNN (Ours)	58.92% (±6.05)	47.95 (±4.14)
Action classification		
Approach	*Accuracy*	*F*1
Dense Trajectories (DT) (Wang et al., 2013)	42.40%	–
Improved Dense Trajectories (IDT) (Wang and Schmid, 2013)	49.60%	–
O+E+H (Li et al., 2015)	57.40%	–
O+M+E+H (Li et al., 2015)	60.50%	–
O+M+E+G (Li et al., 2015)	60.30%	–
Motion stream (Ma et al., 2016)	62.62%	–
Two-stream (Ma et al., 2016)	**65.05%**	–
Two-stream (Wray et al., 2018)	59.02%	–
TSN (Sudhakaran and Lanz, 2018)	55.25%	–
Two-stream (Sudhakaran and Lanz, 2018)	60.13%	–
Appearance stream (Lu et al., 2019a)*	57.63%	–
Motion stream (Lu et al., 2019a)*	57.42%	–
Two-stream (Lu et al., 2019a)*	64.74%	–
Time Distributed Spatial CNN (Ours)	46.55% (±7.84)	36.11 (±4.44)

Table 4.5: Comparative table with the state-of-the-art approaches for active object classification and action classification using the GTEA Gaze+ dataset. The best result is highlighted in bold. The results are given as the mean of a leave-one-subject-out cross-validation approach. Furthermore, for our experiments we repeated each fold three times and we provide the mean and the standard deviation. *They took six subjects into account instead of four (see Section 4.1 for the evaluation strategy).

field. The trajectory is represented by relative point coordinates and various descriptors (HOG, HOF, and MBH) are computed in $N \times N$ neighbourhoods within each trajectory. See Figure 4.4 for an example of dense trajectories for the action "kiss". Wang and Schmid (2013) improved DT by matching feature points between frames using SURF descriptors. These points allowed to estimate an homography using RANSAC. This improvement cancels outs the motion generated by the camera and removes inconsistent matches. Our baseline improved the result obtained by DT features, probably due to the short-term spam of DT features and the robust features computed by CNNs. Nonetheless, DT methods are very appropriate for action recognition due to the modeling of the dynamics of actions instead of the simple observation of the appearance (as in our baseline approach). However, the approaches followed by Wang et al. (2013) and Wang and Schmid (2013) are quite general, i.e., they were not designed to be EAR methods, but rather a more general way to represent the motion. In contrast, Li et al. (2015) exploited egocentric features (see Chapter 2) as well as DT features, motion features (OF), and object features, augmenting those of the DT by computing histograms of LAB color and Local Binary Patterns (LBP) along the trajectories. The use of several of these features obtained an accuracy of **60.50%**, creating a gap of **13.95** points with our baseline. This hint towards the efficacy of egocentric features is quite significant. In fact, the approach of Ma et al. (2016) for active object detection was also based on one of the egocentric cues: the position of hands.

Previously, the result of object classification of Ma et al. (2016) was discussed. However, the most significant contribution of their work is that their result for action recognition remains the highest one of the literature for the GTEA Gaze+ dataset. As aforementioned, they exploited hand positions (egocentric cue) to extract an object-centred region. This allowed to remove background information and focus on the learning of objects. This appearance stream and their motion stream (an OF stream as that of Chapter 3) are fused by concatenation and a FC layer on top is in charge of the classification of actions. Moreover, they added two more objectives: the learning of objects and motion in each stream. With this setting, their approach obtained state-of-the-art results.

Figure 4.4: Visualisation of dense trajectories for the action "kiss". From the work of Wang et al. (2013).

The two-stream network was also used in the work of Wray et al. (2018) and in the work of Lu et al. (2019a). The latter added various modifications such as attention mechanisms. Each stream is composed of an spatial attention network and a temporal network. The spatial attention networks at the beginning aim to output an attention distribution map using the gaze information (a gaussian bump generated from the gaze) as supervision to train this part. Those attention maps are then multiplied with the original inputs to get gaze-aware inputs. The next part is the temporal network, modeled by a bi-directional LSTM network. Both streams are fused at the end. This method highlights the importance of using attention and a more sophisticated temporal modeling. In contrast, our baseline employed a naive approach such as the average of votes and does not include any type of attention.

Many conclusions can be extracted from this baseline and the comparison with other approaches. For active object classification, there is a need to localise objects better. In fact, various objects can appear in each frame, making the task difficult. If the aim is to explicitly recognise the active object, trying to predict it based on several separate frame predictions from CNNs may not be the best idea. There is a need of attention mechanisms to do so.

Nevertheless, the visual information from frames can also be exploited without explicitly learning objects, as several methods do, even though the best result was obtained by explicitly predicting object labels alongside actions (Ma et al., 2016). This seems to pose two possible research lines: (i) the explicit search of active objects (covered in Section 4.3 and used in Section 4.4 for action recognition) and (ii) the implicit learning of visual features (including objects) for action recognition using attention (covered in Section 4.5).

4.3 Learning to detect objects through the Faster R-CNN

Implicitly learning objects (or, rather, learning the visual features that are supposed to contain object features) was the first proposed method to leverage the visual information of the scene with the objective of predicting active objects and actions (see Section 4.2). Nonetheless, given the obtained results and the research lines followed by other authors (Ma et al., 2016), this did not seem to be a promising method towards active object recognition and EAR, in which objects gain a high importance compared to its exocentric counterpart. Therefore, our next step is to propose a method that explicitly models object information. In this section, we aim to explain how objects can be discovered in images and encoded to be used later.

4.3.1 Faster R-CNN

The field that researches the task of detecting objects within images is called object recognition, not to be confused with object classification (given an image predicting a label) or object segmentation (given an image outputting a per-pixel prediction of objects). Object recognition aims to detect objects in images by means of BBs, rectangular areas within images that have an associated probability distribution for a set of objects (see Figure 4.5 for an example).

With the increased popularity of DL methods, various authors have proposed methods to apply DL techniques for the object recognition task. A

Figure 4.5: Sample image from the GTEA Gaze+ dataset with a bounding box inscribing a milk container object. An example of a probability distribution (in percentages) for that bounding box is provided.

highly addressed method was the R-CNN (Girshick et al., 2015) in 2015. It first leverages the selective search algorithm (Uijlings et al., 2013) to create proposals of connected regions (or better said, per-pixel predictions). Each region proposal is resized to an standard shape to match the input size of a CNN and, then, (i) a classifier such as an SVM is used on top to predict the probability distribution of the region and (ii) a linear regressor is used to tighten the BB to the putative object's shape. The second step allows for fine-grained predictions in which rectangular boxes envelop objects with the minimum possible noise (visual information not related to the object of interest in each case). The architecture of the R-CNN is represented in Figure 4.6.

A new version of this network was later presented, called Fast R-CNN (Girshick, 2015), that aimed at alleviating the computational burden of the first approach. This network takes the whole image as input to the CNN (instead of iterating the forward pass for each proposal). Regions are extracted from the output feature map and the same branched objectives (object

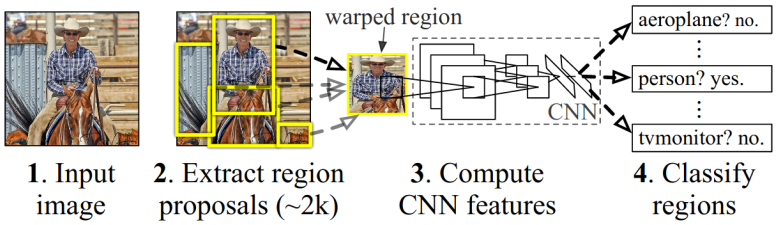


Figure 4.6: Region-based Convolutional Network (R-CNN). From the work of Girshick et al. (2015).

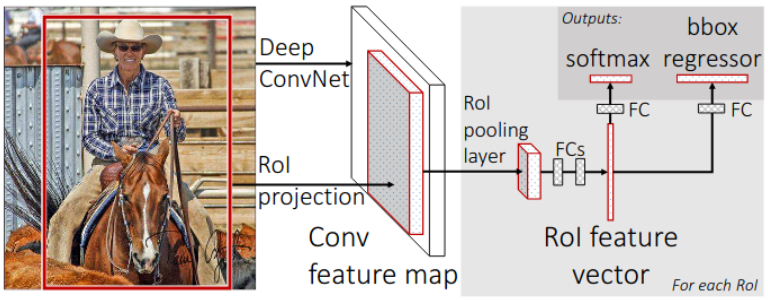


Figure 4.7: Fast Region-based Convolutional Neural Network (Fast R-CNN). From the work of Girshick (2015).

classification and BB adjustment) can be applied again. The architecture is presented in Figure 4.7. However, this version was still dependent of the Selective Search method, which was a computationally expensive method. That is why the Faster R-CNN was finally proposed (Ren et al., 2015). In this last update, the RPN module was added. The RPN creates BB proposals at the beginning (not needing anymore the selective search) and these are processed by the Fast R-CNN. However, the system cannot be seen as separated modules, as both modules (the RPN and the Fast R-CNN) are jointly trained in an end-to-end fashion. To summarise, the Faster R-CNN is a network that takes an image and is capable of outputting BBs that inscribe objects (and have associated probability distributions). This network will be employed through this section.

4.3.2 **Experiments**

In order to have a functional object detector, for a given dataset, the detector must be trained for the specific set of objects and their shape. However, pre-trained models can be reused to improve the convergence time and the overall results of fine-tuned detectors, given that the pre-training was done using large datasets such as COCO (Common Objects in Context) (Lin et al., 2014). This is a standard procedure with the R-CNN family of networks. Nevertheless, image annotations are still required for the fine-tuning and the evaluation, i.e., for each image, a set of BBs (including the position of the rectangle and the ground truth class). Some datasets already provide these annotations, but the GTEA Gaze+ did not, at least at the time this dissertation was written. That is why the authors manually annotated several samples (images) across different classes and subjects to be able to train and evaluate an object detector. A total of **30** objects were annotated, including hands as another object too. Figure 4.8 shows an example of the distribution of object annotations across several images in Ahmad's fold. It can be observed that the presence of hands dominates the rest of the classes, as they appear in almost all the action classes.

The Faster R-CNN, with a ResNet101 as the backbone feature extractor, was trained for each subject individually. The training data of each subject's split was used to fine-tune the network, which was already pre-trained in the COCO dataset. Hence, the convolutional layers were frozen for the fine-tuning. For the RPN's anchors, the stride was set to 8 and five scales (**0.1**, **0.25**, **0.5**, **1**, and **2**) and three aspect ratios (**0.5**, **1**, and **2**) were used. At the first stage, the Non Maximum Supression (NMS) (used to remove some proposals and predictions) score and the Intersection over Union (IoU) threshold (explained later) were set to **0** and **0.7**, while for the second stage these were changed to **0.3** and **0.6**, respectively. In addition, the first stage considered up to **100** proposals while this was reduced to **10** in the second one (the total number of output predictions). Regarding the hyper-parameters used for training, the learning rate was set fixed to $3e^{-4}$, the input size was 224×224 (any image larger or smaller than that was resized), and a batch

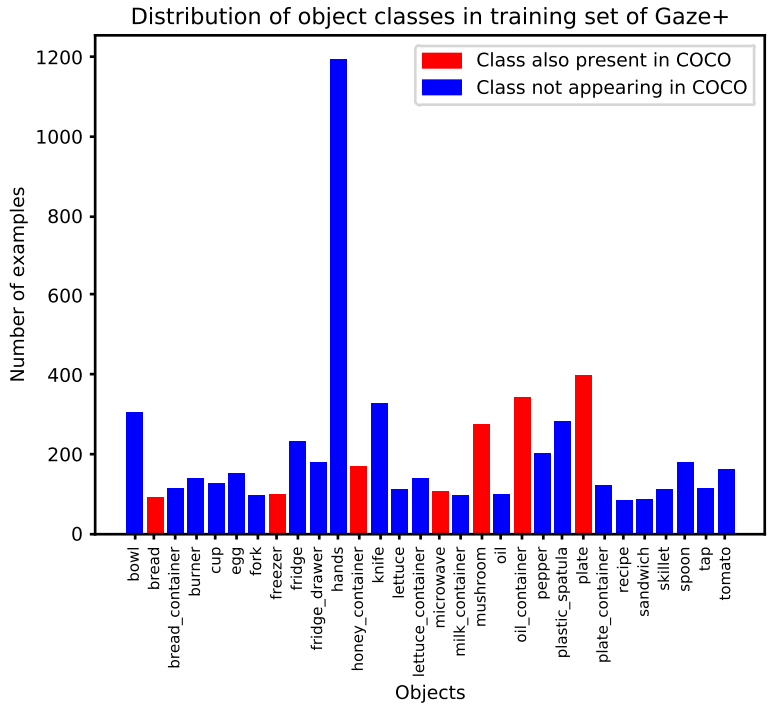


Figure 4.8: Object classes annotated for the GTEA Gaze+ dataset. The classes that intersect with the COCO dataset are highlighted in red.

Subject	*Samples*	*AP*@[.5:.95]	*AP*@.5	*AP*@.75	*AR*@1	*AR*@10	*AR*@100$_{small}$	*AR*@100$_{medium}$	*AR*@100$_{large}$
Ahmad	766	33.0	50.0	37.9	35.9	39.1	5.8	33.8	61.9
Alireza	185	36.2	54.9	39.8	38.3	41.3	13.4	39.6	67.3
Carlos	251	38.1	57.8	41.5	40.7	43.0	9.1	41.3	60.8
Rahul	231	29.3	47.0	31.2	30.3	33.7	8.8	29.3	52.1

Table 4.6: Object detection results on the GTEA Gaze+ dataset's folds using the COCO dataset's evaluation metrics.

size of 1 was used. For data augmentation, the standard random horizontal flipping and random cropping were applied.

After the training, the object detector was able to, given an image, output 10 predicted objects containing the BB position, its class (and the confidence), and a feature vector. The results obtained for each subject's model are summarised in Table 4.6 and some visualisations of the predicted BBs are provided in Figure 4.9. The metrics employed in Table 4.6 are the COCO evaluation metrics that are based on the Average Precision (AP) and the Average Recall (AR). To explain what AP and AR are, first, the values of TP, FP, FN, and TN for the object detection task must be defined. But, even before these, the concept of the IoU is pivotal to define them. It is a numeric value that determines the amount of overlap between two BBs, i.e., the area where both BBs intersect divided by the total area (the union) of both BBs. Figure 4.10 shows a graphical representation of the idea. The value is represented with a float number in the range $[0.0, 1.0]$: when both BBs are completely overlapped the IoU is 1.0, but if there is no overlapping it is 0.0. With this concept, the TP, FP, FN, and TN can be formulated as follows:

- TP: A BB has an $IoU > t_{iou}$ with the ground truth BB and both are tagged with the same object class, being t_{iou} a threshold value.
- FP: A BB has an $IoU <= t_{iou}$ with the ground truth BB and both are tagged with the same object class, being t_{iou} a threshold value.
- FN: If an object is present in an image and the object detector failed at detecting it.

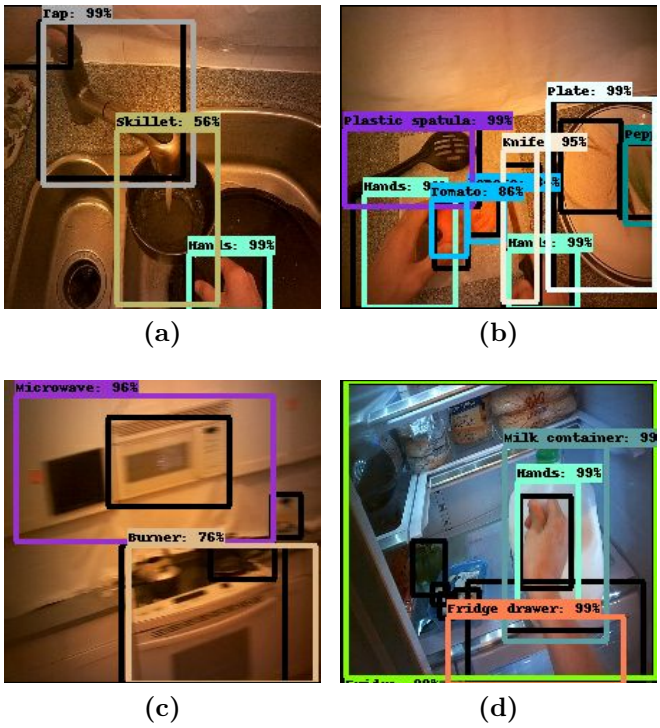


Figure 4.9: Object detections from the Faster R-CNN network on GTEA Gaze+ dataset's images. The black boxes are ground truth annotations.

- TN: Any part of the image not predicted as an object. However, for the object detection task, this value is not useful and is not employed for computing evaluation metrics.

Using these values, the precision (Equation 4.2) and recall (Equation 4.3) can be computed. Notice that having BB predictions larger or smaller than the ground truth will harm the precision, as the IoU with the ground truth box can be smaller than the threshold. For the recall, the inability to detect all the ground truth boxes will be penalised.

The evaluation metrics used in COCO can now be defined with all the aforementioned terms. Their main purpose is to provide meaningful metrics for the object detection task, adapting the usual classification scheme to the detection scheme. This includes the previous definition of TP, FP and FN. Specifically, the metrics used in COCO and included for these experiments are variations of the precision and the recall. The AP is a term that averages

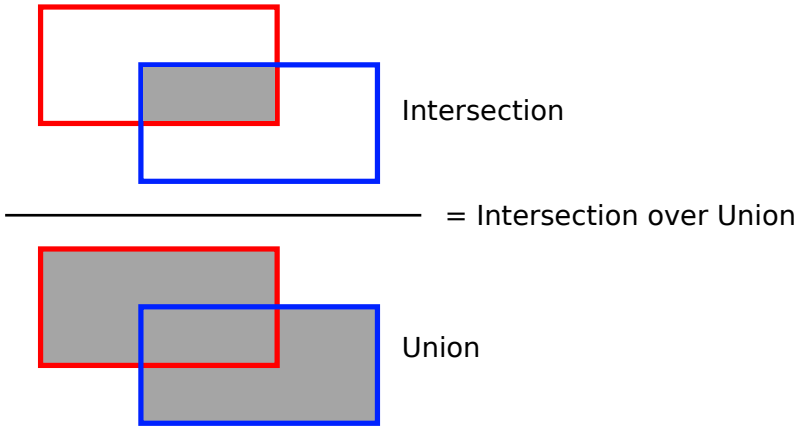


Figure 4.10: The Intersection over Union (IoU) between two bounding boxes.

the precision across several t_{iou} values. It also averages the results across classes, which would be known as mean AP or mAP, but COCO makes no distinction between them. Specifically, the $AP@[.5:.95]$, $AP@.5$, and $AP@.75$ metrics are used. The $AP@[.5:.95]$ averages t_{iou} values ranging from 0.5 to 0.95 with a step of 0.05, with a total of 10 values. It is usually considered the primary challenge metric for the evaluation in the COCO dataset. The $AP@.5$ and $AP@.75$ evaluate the performance using a single t_{iou} value, 0.5 and 0.75, respectively. Then, there is also the AR, which is presented with the $AR@1$, $AR@10$, $AR@100_{small}$, $AR@100_{medium}$, and $AR@100_{large}$ variations and is averaged across all classes and IoU thresholds. The $AR@1$ and the $AR@10$ metrics compute the AR using the top-1 and top-10 predictions (the ones with the highest confidence score). It is common to present the $AR@100$ too, but, in this case, the maximum number of predictions is 10 and, thus, $AR@10 = AR@100$ no matter what. Finally, the $AR@100_{small}$, $AR@100_{medium}$, and $AR@100_{large}$ compute the $AR@100$ for small, medium, and large objects, respectively. COCO defines small objects as those whose area is smaller than 32^2, then medium ones as those whose area is between 32^2 and 96^2, and the large ones as those whose area is larger than 96^2. Notice that $AR@100$ is employed instead of $AR@10$ to follow COCO's conventions, although both values are the same.

In addition to the BB information, feature vectors from the step previous to the classification of proposals are extracted. To get them, the Faster R-

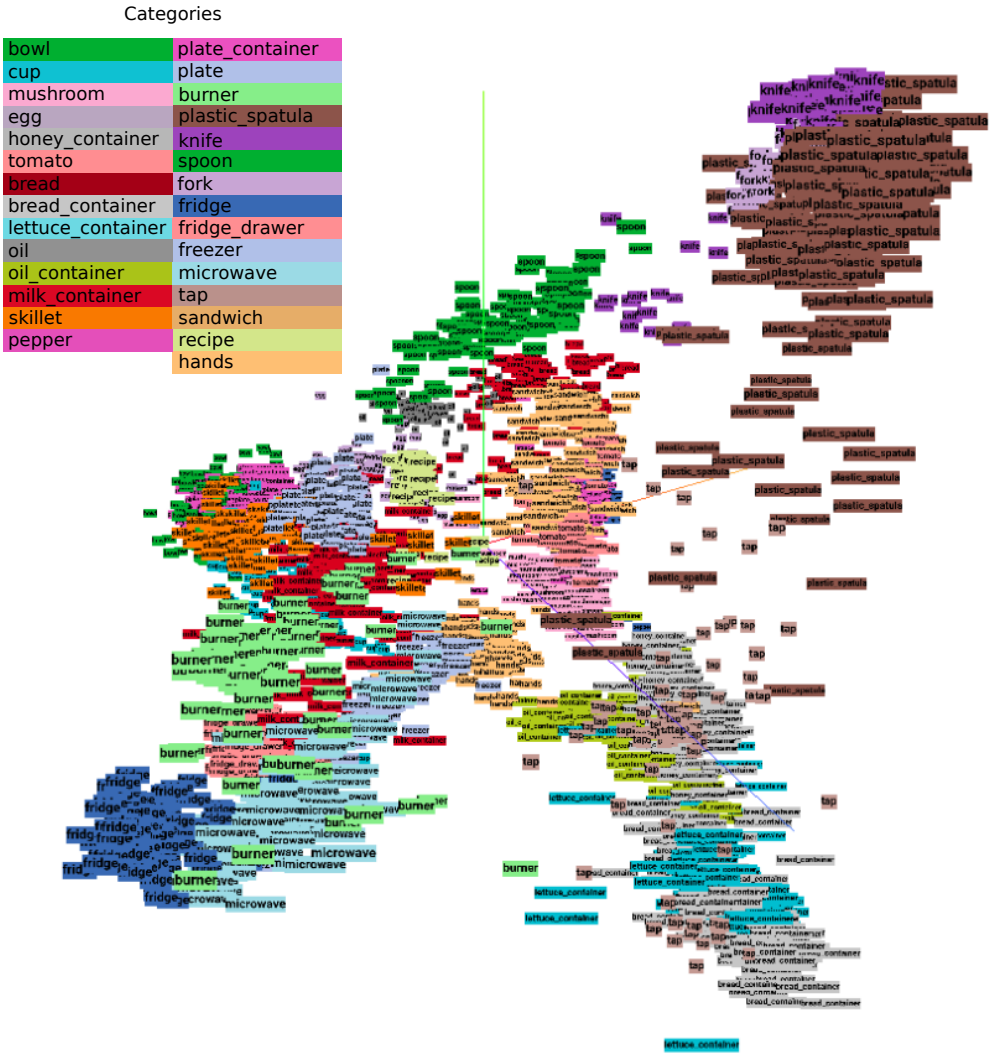


Figure 4.11: Feature vectors extracted from the Faster R-CNN represented after a Principal Component Analysis, leaving 3 components. Each class is assigned a color for a better visualisation.

CNN gets the feature cube of a region proposal, with shape $N \times M \times C$, being N and M the width and height of the region and C is the number of channels of the last feature cube of the backbone network, being 2048 for the ResNet101. A pooling operation (global average pooling) is applied to it to get a vector of size C, then fed through FC layers. This is the feature vector used for classification and the one our object detector outputs. It should encode, at least, the position of the BB and its class. We will be using this feature vectors in the following Section 4.4. We visualised a set of feature vectors using PCA, leaving 3 components, in Figure 4.11. A color was assigned to each class alongside its name for a better visualisation. As it can be observed, each class is fairly well clustered.

4.4 The Bag of Objects Matrix representation

Section 4.3 explained how objects could be automatically inferred from images without further help (for the inference phase only). With the aim of creating an active object recognition system, the set of detected objects can be leveraged to create a system that is able to process it and, then, predict active objects and actions. That will be the goal of this section. But first, the output obtained from the previous section's Faster R-CNN will be formalised.

A video contains a set of N frames. For each frame fed to the Faster R-CNN, P predictions (set to "10 in our work) are obtained. Therefore, for each frame, the following information is extracted:

- BB coordinates ($P \times 4$ matrix): an element p_i ($0 \leq i \leq P$) within this matrix is a vector of 4 values (y_1, x_1, y_2, and x_2) corresponding to the i^{th} BB. The pair (y_1, x_1) defines the top-left point of the BB, while the pair (y_2, x_2) defines the bottom-right one.

- Score or confidence (P-sized vector): an element p_i ($0 \leq i \leq P$ and $0 \leq p_i \leq 1$) within this vector corresponds to the confidence in the predicted class of the i^{th} BB.

- Classes (P-sized vector): an element p_i ($0 \leq i \leq P$ and $0 \leq p_i < |O|$) within this vector corresponds to the class identifier of the i^{th} BB, taking $|O|$ possible object class values, where O is a set of objects.
- Number of detections (integer): the number of not null predictions M ($0 \leq M \leq P$). That is, out of P entries, only the first M BBs are valid predictions.
- Feature vector ($P \times C$ matrix): an element p_i ($0 \leq i \leq P$) within this matrix encodes features related to the i^{th} BB. C is fixed to 2048 due to the choice of the ResNet101 as the backbone of the Faster R-CNN (number of channels in the last feature cube).

So far, outputs of single images have been analysed; therefore, the first objective of this section is to produce representations of videos given all these features. That is, a single representation that encodes all the object information from a video. As videos have different lengths, there is a need to collapse or pool the information to a standardised size. Formally, videos have a varying length of L_v (number of frames) and we aim to obtain a representation of size L. This work proposes to pool the information following these rules: (i) if the size matches, no pool is applied; (ii) if $L_v > L$, uniform subsampling is applied; and, (iii) if $L_v < L$, we augment the original video by copying elements. For the latter, if the original video has a set of frames $\{f_0, f_1, f_2, ... f_{L_v}\}$, we duplicate the frames to get an arrangement like $\{f_0, f_0, f_1, f_1, f_2, f_2, ... f_{L_v}, f_{L_v}\}$. If the new length is greather than L_v, the method applied to subsample in (ii) is used. Now each video should have a common length of L. The question now is, how should the information from the Faster R-CNN be arranged? We propose the following two approaches.

4.4.1 Input representations

The raw representation. This first encoding of the video, as it can be seen in Figure 4.12, has a cube shape of $P \times C \times L$, where the depth dimension represents the time dimension (the evolution of the video frame to frame) and, thus, the depth is set to L (number of frames). A depth slice of shape $P \times C$

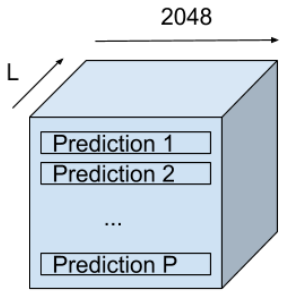


Figure 4.12: The raw representation proposed to encode the Faster R-CNN output of all the frames of a video.

encodes the information about a frame, where P and C are the maximum number of predictions and the size of the feature vectors, respectively, as described at the beginning of Section 4.4. That is, the feature vectors from the objects detected in the i^{th} frame are stacked so that they form a matrix of shape $P \times C$ and this matrix is located in the i^{th} depth slice of the final representation of the video.

The segments approach. The second approach aims at collapsing more information. Given a video of size L, we define K segments from where to extract T frames with a stride or separation of S frames. K and T are fixated for all the videos. Figure 4.13 represents a video as a rectangular block, in which the time dimension is the width (with a size of L slices) and the content of each temporal slice is defined later. It can be seen how K blocks are built, each one constructed with T frames. Depending on the stride S, the blocks can be overlapped, as in the example, in which block 1 and 2 have a little overlap. For instance, in the case of setting K to 5 and T to 10, sampling segments from a video with an L of 50, we would cover the whole video (and use all the frames) with a stride S of 10, creating non-overlapping segments. For these experiments, we uniformly sample blocks by setting S to $\frac{L-T}{K-1}$.

When the blocks are defined, we aim at colapsing each block so that the object information is compressed. For that, we start by defining a matrix of shape $|O| \times C$, where O is the set of objects and C the dimension of their feature vectors, initialised to zero. We call this type of structure, yet to be filled, the Bag of Objects Matrix (BOM) of the block. In fact, a BOM is just a matrix collecting objects in a row-wise arrangement, each of the raw representation's

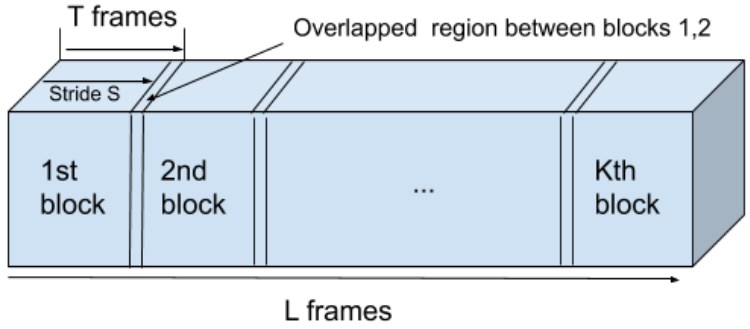


Figure 4.13: The segments representation proposed to encode the Faster R-CNN output of all the frames of a video.

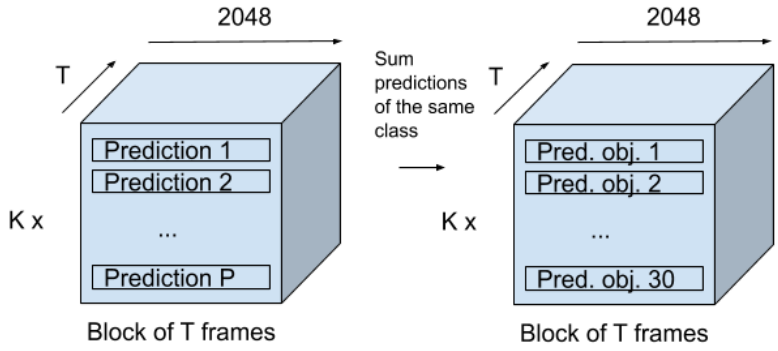


Figure 4.14: The feature vectors of each object class are added frame-wise (objects can be repeated within a frame, so their feature vectors are added).

slices can also be considered BOMs. Regarding how the BOM in this approach is filled, assume that the first dimension is a dictionary with $|O|$ elements (one per object), with entries of size C that represent the feature of the i^{th} object ($0 \leq i < |O|$). We want to pool the information about every type of object into a feature vector within the time span of T frames of a block.

For that, for each of the T frames within a block, the predictions of the i^{th} object (for all i such that $0 \leq i < |O|$) are added by vector addition, specially for those cases in which the object is repeated within the frame. The null predictions are ignored, so any object not appearing in a frame would have a feature vector of zeros after the addition. With this, the new structure has a shape $|O| \times T \times C$ and there are K of these, one per block. Figure 4.14 illustrates this process.

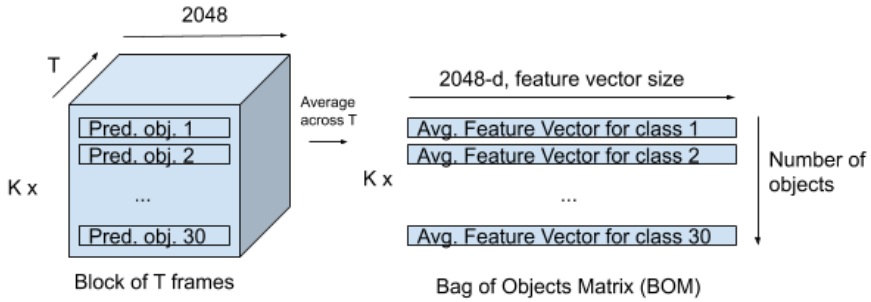


Figure 4.15: Within each block, a mean pooling operation across the depth dimension is done. This way, the BOMs of a single block are pooled.

In a second step, the frame-wise BOMs are going to end up collapsed in a single BOM (representing the information from a block). For that, we applied a mean pooling operation across the depth dimension (of size T). That is, for the i^{th} object, all the feature vectors of the same object class of the block are mean pooled. From this step, a set of K BOMs of shape $|O| \times C$ are obtained. The final input is obtained by arranging the obtained structure to have a shape $|O| \times C \times K$.

4.4.2 Architecture

In Section 4.4.1, the raw and the segments representations were explained. These allowed for a encoding of the object information of a video. Nonetheless, apart from the representation of the data, a feature extractor (CNN in this case) and a classifier capable of inferring active objects and actions from those structures are needed.

Due to the structure of the input, not being a space in which the adjacency of pixels is meaningful as in images, the use of pre-trained networks and the standard image processing networks is discarded from the beginning. We draw inspiration from the Natural Language Processing (NLP) field, and, more specifically, from the work of Kim (2014), who proposed a CNN architecture for NLP. The connection between the latter task and ours might not be obvious at first. However, the extended use of word embeddings (feature vectors encoding words) used in that work resembles our object feature vectors. He arranges word embeddings in a matrix row-wise, where the i^{th} row

contains the i^{th} word of the phrase. In our case, the i^{th} row of a depth slice contains the i^{th} object's information.

The network, called Multi-Scale Convolutional Neural Network (MSCNN), is shown in Figure 4.16 adapted to our task. It is a branched architecture with four paths at the beginning. Each of them applies first a convolutional layer with a variable filter size: the width is fixed to the maximum (2048) in every filter, while the height goes from 2 in the first branch to 5 in the last branch. The purpose of the filter is to convolve together a set of objects, sets of different sizes depending on the height of the filter. The result is a matrix that is max pooled in the second step. The shape of the max pooling filter is also adapted to the new height of the feature cube (varying from branch to branch); thus, resulting in a feature vector per branch as output. All of them are concatenated to create a single feature vector and this is fed to an MLP classifier to be trained on active objects and action labels.

Nevertheless, we believe the first convolution of each branch colapses too much information at once, in contrast to what happens in the work of Kim (2014), i.e., words embeddings often have a lower dimensionality compared with object feature vectors. That is why another variation of the network is proposed by adding another convolution step first, slowly reducing the dimensionality. The new architecture can be seen in Figure 4.17.

4.4.3 **Experiments**

The GTEA Gaze+ dataset is used for the training and evaluation of the MSCNN model. Hence, a leave-one-subject-out cross-validation is applied. For each fold, a Faster R-CNN model is trained using the BB annotations of the rest of the subjects as explained in Section 4.3. For this problem, we defined the following TP, FP, TN, and FN for the accuracy and F1 metrics:

- TP: for the class i, if an input stack of BOMs is labeled with class i and the class i is the one with the highest confidence.

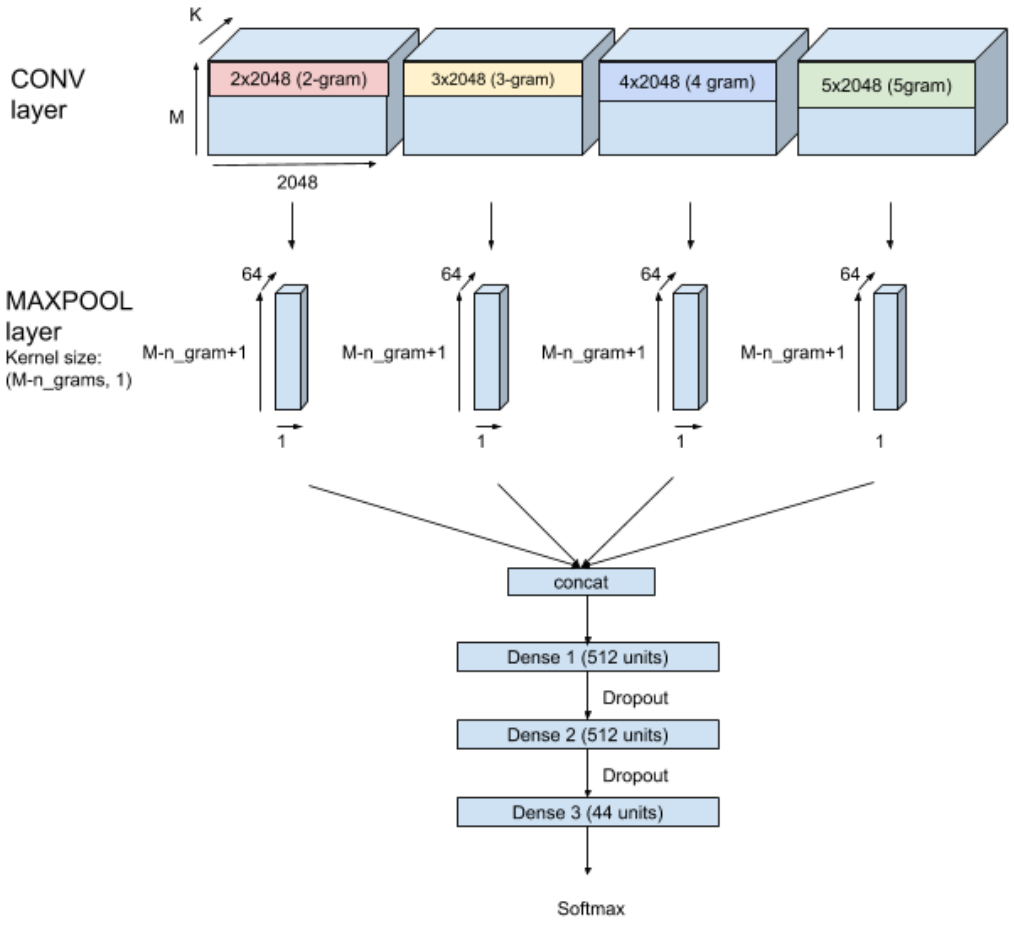


Figure 4.16: The CNN architecture used by Kim (2014) adapted to the task of active object and action recognition with encoded object detection information as input.

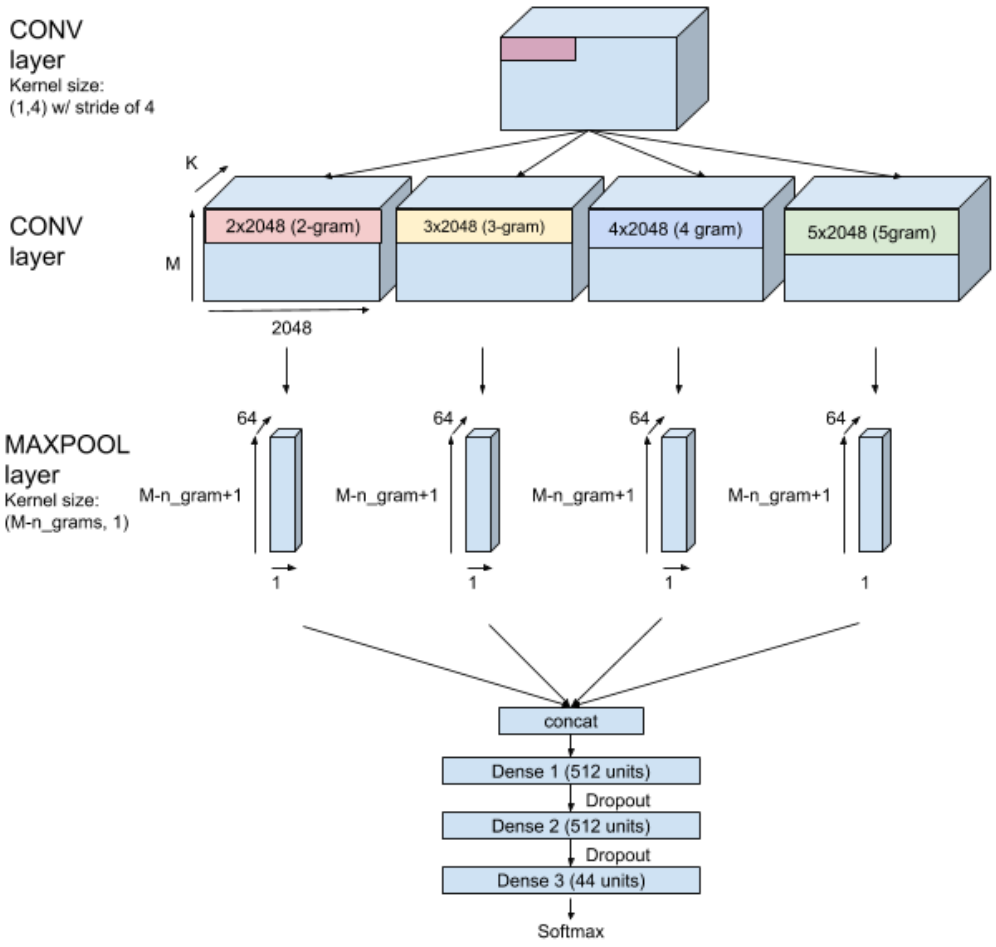


Figure 4.17: A variation of the CNN architecture used by Kim (2014) adapted to the task of active object and action recognition with encoded object detection information as input. Another convolution has been added at the beginning to reduce the dimensionality of the object feature vectors.

- FP: for the class i, if an input stack of BOMs is labeled with class j and the class i is the one with the highest confidence, being j any class different to i.
- TN: for the class i, if an input stack of BOMs is labeled with class j and the class k is the one with the highest confidence, being j and k any classes different to i and being possible that $j = k$.
- FN: for the class i, if an input stack of BOMs is labeled with class i and the class j is the one with the highest confidence, being j any class different to i.

Several experiments to tune the hyper-parameters are done using the cross-validation approach. Initially, 500 epochs are set, but the training is stopped using the the macro-F1 metric in the validation set, with a patience of 10 epochs. The validation set was created taking a stratified set with the 15% of the training samples. Two validation sets were created: one stratifying the object label (for the active object classification task) and the other one stratifying the action label (for the action classification task). For the results, the accuracy and macro-F1 metrics are provided, both as the average of all the subjects and per subject. For each subject, 3 runs are executed and averaged. The best accuracy and macro-F1 results are considered those with the highest average result, not taking into account the standard deviation due to the higher variability it supposes. That is, two results with the same value may have different standard deviations: the one with the highest deviation may obtain higher results sometimes, but it may also obtain much lower results in other occasions. Thus, we believe the average is a better tool to compare two results.

For the active object classification task, the results are shown in Tables 4.7 (using the raw representation) and 4.8 (using the segments representation). For the action classification task, the results can be found in Tables 4.9 (using the raw representation) and 4.10 (using the segments representation).

The segments representation seems to get slightly better results than the raw one. We can observe in both cases that sampling fewer frames (30 in comparison with 50) leads to lower results. The median of frames for the GTEA

L	FC units	Dropout	Learning rate	Batch size	Accuracy	Macro−F1	Accuracy	Macro−F1	Accuracy	Macro−F1	Accuracy	Macro−F1	Accuracy	Macro−F1
Hyperparameters					*Average*		*Ahmad*		*Alireza*		*Carlos*		*Rahul*	
50	1024	0.5	0.001	256	50.08%	31.25	47.02%	27.75	50.62%	30.85	48.56%	32.46	54.11%	33.96
					(±2.86)	(±2.73)	(±0.28)	(±1.77)	(±1.14)	(±0.73)	(±1.04)	(±4.27)	(±1.47)	(±2.21)
50	1024	0.5	0.0001	256	**52.47%**	35.39	46.36%	27.90	54.88%	36.06	**51.65%**	39.26	**56.97%**	**38.36**
					(±4.08)	(±4.65)	(±0.25)	(±0.76)	(±0.73)	(±1.51)	(±0.88)	(±1.49)	(±1.12)	(±1.03)
30	1024	0.5	0.001	256	50.94%	32.38	47.02%	28.67	**55.11%**	**36.19**	47.72%	34.94	53.92%	29.71
					(±3.84)	(±4.47)	(±1.27)	(±2.22)	(±0.42)	(±4.38)	(±1.14)	(±2.74)	(±1.95)	(±2.50)
30	1024	0.5	0.0001	256	51.83%	**35.40**	**47.16%**	**31.32**	53.87%	34.46	49.61%	**42.46**	56.69%	33.34
					(±3.75)	(±5.41)	(±0.41)	(±2.99)	(±1.20)	(±4.26)	(±0.10)	(±4.27)	(±0.35)	(±0.20)

Table 4.7: Results of the experiments with the Multi-Scale Convolutional Neural Network and the inputs built using the raw approach for the active object classification task. For each row and column, the average result is presented on top and the standard deviation is shown below. The highest column-wise mean is highlighted in bold.

Gaze+ dataset is 31, so one could have expected a good result for a value close to that. However, in this case, it is better to bias that towards longer videos, i.e., prioritise a value of L larger than the median of the length that half of the dataset's videos have. In fact, there are very long videos in the dataset, specially those related to cutting vegetables. It would make sense that, after undersampling them, the performance of the system on those classes would not be hurt due to the low variance long videos have across frames, as the objects should not vary much. Regarding the other parameters to build the segments approach, K and T, it is difficult to extract conclusions from these experiments. This may be due to the low effect they have in the performance. The best results with the raw and the segments representations are compared with the state-of-the-art approaches and with our baseline SpatialNet in Table 4.11.

Despite the promising approach used to represent videos, the results are not as high as expected, being even lower than the baseline proposed in Section 4.2. There may be various reasons for this. For example, the part of the Faster R-CNN from which the features are extracted may be one of the keys. For instance, Ma et al. (2018) exploited object features from an object detector and explained that they did not extract these features from the R-CNN part

L	K	T	FC units	Dropout	Learning rate	Batch size	Average Accuracy	Average Macro−F1	Ahmad Accuracy	Ahmad Macro−F1	Alireza Accuracy	Alireza Macro−F1	Carlos Accuracy	Carlos Macro−F1	Rahul Accuracy	Rahul Macro−F1
Hyperparameters							*Average*		*Ahmad*		*Alireza*		*Carlos*		*Rahul*	
50	5	5	1024	0.5	0.001	256	54.84% (±4.07)	43.15 (±4.21)	51.32% (±0.94)	37.40 (±2.54)	57.35% (±0.57)	48.20 (±0.24)	50.67% (±1.21)	43.65 (±2.32)	60.02% (±0.86)	43.34 (±0.21)
50	5	10	1024	0.5	0.001	256	53.34% (±4.02)	42.11 (±5.11)	50.40% (±0.43)	34.53 (±0.76)	55.78% (±0.69)	45.02 (±1.97)	49.47% (±1.47)	45.51 (±2.36)	57.71% (±3.62)	43.36 (±3.91)
50	5	2	1024	0.5	0.001	256	53.53% (±2.68)	43.01 (±4.32)	51.52% (±1.17)	38.63 (±3.60)	55.78% (±1.04)	47.49 (±1.39)	50.67% (±0.78)	45.22 (±1.70)	56.14% (±1.29)	40.71 (±2.74)
30	5	5	1024	0.5	0.001	256	53.83% (±6.05)	41.55 (±7.15)	47.62% (±1.13)	32.29 (±1.57)	57.69% (±1.14)	47.58 (±3.30)	48.42% (±0.45)	37.59 (±0.38)	61.59% (±0.91)	48.75 (±1.09)
50	10	5	1024	0.5	0.001	256	55.29% (±3.84)	43.83 (±3.26)	**52.31**% (±0.52)	39.04 (±1.11)	56.34% (±0.69)	45.43 (±0.23)	51.58% (±1.03)	45.72 (±2.57)	60.94% (±1.26)	45.15 (±1.99)
50	10	10	1024	0.5	0.001	256	55.09% (±3.96)	43.57 (±3.69)	51.26% (±0.89)	**39.40** (±2.59)	56.90% (±1.67)	45.84 (±3.94)	51.72% (±0.26)	43.85 (±1.92)	60.48% (±0.73)	45.18 (±1.84)
50	10	2	1024	0.5	0.001	256	54.81% (±3.57)	43.31 (±4.30)	51.98% (±.33)	37.33 (±2.75)	57.58% (±0.00)	48.09 (±1.01)	50.88% (±0.65)	43.06 (±1.17)	58.82% (±1.33)	44.75 (±1.78)
50	5	5	1024	0.5	0.0001	256	**57.40**% (±6.05)	43.70 (±6.58)	50.20% (±1.33)	36.27 (±1.46)	**60.94**% (±0.27)	**53.82** (±0.74)	**53.19**% (±0.53)	40.98 (±0.58)	**65.28**% (±0.35)	43.75 (±2.24)
50	5	10	1024	0.5	0.0001	256	56.72% (±5.46)	43.06 (±5.71)	52.12% (±0.19)	34.92 (±0.53)	57.80% (±0.69)	48.32 (±1.11)	51.79% (±0.17)	40.73 (±0.88)	65.19% (±0.65)	48.26 (±1.34)
30	5	5	1024	0.5	0.001	256	53.83% (±6.05)	41.55 (±7.15)	47.62% (±1.13)	32.29 (±1.57)	57.69% (±1.14)	47.58 (±3.30)	48.42% (±0.45)	37.59 (±0.38)	61.59% (±0.91)	48.75 (±1.09)
30	5	10	1024	0.5	0.001	256	55.80% (±5.54)	44.70 (±7.59)	50.73% (±1.17)	34.39 (±1.13)	59.26% (±1.26)	49.42 (±1.60)	50.53% (±2.48)	43.53 (±6.08)	62.70% (±0.79)	**51.45** (±3.73)
30	5	2	1024	0.5	0.001	256	53.69% (±4.11)	41.48 (±5.30)	50.26% (±0.77)	33.40 (±1.56)	56.34% (±0.84)	44.61 (±1.16)	49.26% (±0.52)	40.78 (±0.46)	58.91% (±0.65)	47.12 (±1.08)
30	10	5	1024	0.5	0.001	256	54.91% (±4.72)	44.60 (±7.24)	50.13% (±0.52)	33.95 (±0.40)	59.37% (±1.38)	52.12 (±0.52)	50.60% (±1.01)	42.87 (±3.02)	59.56% (±1.71)	49.46 (±1.78)
30	10	10	1024	0.5	0.001	256	54.88% (±5.42)	43.25 (±5.86)	49.14% (±0.92)	33.63 (±1.22)	57.58% (±0.99)	47.10 (±1.01)	50.67% (±1.74)	45.11 (±1.33)	62.14% (±1.38)	47.18 (±2.62)
30	10	2	1024	0.5	0.001	256	55.25% (±3.90)	**45.42** (±5.25)	51.46% (±1.56)	37.93 (±3.93)	58.02% (±0.42)	49.39 (±0.75)	51.86% (±0.60)	**49.25** (±1.08)	59.65% (±2.23)	45.12 (±2.55)

Table 4.8: Results of the experiments with the Multi-Scale Convolutional Neural Network and the inputs built using the segments approach for the active object classification task. For each row and column, the average result is presented on top and the standard deviation is shown below. The highest column-wise mean is highlighted in bold.

L	FC units	Dropout	Learning rate	Batch size	Accuracy	Macro−F1	Accuracy	Macro−F1	Accuracy	Macro−F1	Accuracy	Macro−F1	Accuracy	Macro−F1
Hyperparameters					*Average*		*Ahmad*		*Alireza*		*Carlos*		*Rahul*	
50	1024	0.5	0.001	256	38.36% (±5.35)	24.03 (±2.74)	32.47% (±1.62)	19.97 (±0.49)	43.32% (±1.51)	25.72 (±2.69)	33.96% (±0.69)	25.47 (±0.21)	43.67% (±1.51)	24.95 (±0.45)
50	1024	0.5	0.0001	256	**41.25%** (±4.42)	**27.20** (±2.27)	**37.37%** (±0.73)	**25.16** (±1.53)	43.21% (±1.24)	**27.35** (±1.01)	**37.05%** (±1.36)	29.76 (±2.46)	**47.37%** (±0.60)	26.54 (±0.23)
30	1024	0.5	0.001	256	37.80% (±6.12)	24.17 (±2.51)	30.36% (±0.28)	20.36 (±0.91)	42.31% (±0.97)	25.02 (±2.12)	33.47% (±1.03)	25.66 (±0.56)	45.06% (±0.79)	25.62 (±0.24)
30	1024	0.5	0.0001	256	39.65% (±4.58)	27.11 (±2.87)	34.66% (±0.37)	23.59 (±0.23)	**43.66%** (±0.16)	26.58 (±0.40)	35.58% (±0.62)	**31.39** (±1.03)	44.69% (±0.47)	**26.88** (±0.81)

Table 4.9: Results of the experiments with the Multi-Scale Convolutional Neural Network and the inputs built using the raw approach for the action classification task. For each row and column, the average result is presented on top and the standard deviation is shown below. The highest column-wise mean is highlighted in bold.

(as we did), but rather from the object proposal network (RPN) itself. Taking the feature vectors from the R-CNN leads to a cross-domain problem, which may be one of the reasons for having low results. In their setting, they also mention that by taking pruned results from the R-CNN (only considering annotated objects) there is a possibility of missing objects that are not recognised as objects. This means objects are taken without labeling, just exploiting the information of the feature vectors. This is a reasonable approach to represent the objects of the scene. In fact, Ma et al. (2018) also leveraged attention mechanisms to represent the interactions between objects, leading to competitive results comparable to those of I3D (Carreira and Zisserman, 2017) and TSN (Wang et al., 2016b) in the Kinetics dataset (Carreira and Zisserman, 2017). This work may provide a hint on the importance of using feature vectors from the RPN and the contribution of attention mechanisms towards compressing all the features. In a similar fashion, Shen et al. (2018) extracted features of specific regions using the RPN's detections instead of those of the R-CNN. We believe this was one of the main shortcomings of our experiments. However, it was not the unique one. The way the feature vectors are used in the previous works or in recent ones such as the one carried out by Wang et al. (2019c) lead to the conclusion that the features must be pooled or

L	K	T	FC units	Dropout	Learning rate	Batch size	Average Accuracy	Average Macro−F1	Ahmad Accuracy	Ahmad Macro−F1	Alireza Accuracy	Alireza Macro−F1	Carlos Accuracy	Carlos Macro−F1	Rahul Accuracy	Rahul Macro−F1
Hyperparameters							*Average*		*Ahmad*		*Alireza*		*Carlos*		*Rahul*	
50	5	5	1024	0.5	0.001	256	44.57% (±5.91)	34.71 (±3.02)	40.28% (±0.58)	29.99 (±0.34)	**46.91**% (±0.57)	36.22 (±0.72)	38.11% (±0.30)	34.96 (±0.49)	53.00% (±1.45)	37.66 (±1.56)
50	5	10	1024	0.5	0.001	256	44.00% (±5.87)	33.73 (±1.87)	**41.93**% (±0.25)	**31.60** (±0.68)	43.66% (±1.66)	32.77 (±0.37)	37.40% (±0.20)	36.05 (±0.43)	53.00% (±2.42)	34.49 (±1.32)
50	5	2	1024	0.5	0.001	256	42.77% (±4.05)	32.58 (±3.16)	38.69% (±0.16)	27.72 (±1.19)	44.89% (±0.88)	32.63 (±0.88)	39.30% (±1.11)	35.39 (±0.99)	48.20% (±0.90)	34.60 (±1.08)
50	10	5	1024	0.5	0.001	256	43.62% (±5.99)	32.15 (±3.88)	38.43% (±1.60)	26.53 (±0.78)	45.12% (±0.73)	32.15 (±0.25)	38.32% (±1.05)	34.91 (±1.80)	52.63% (±0.82)	35.00− (±3.00)
50	10	10	1024	0.5	0.001	256	42.84% (±5.58)	32.14 (±2.87)	**41.93**% (±0.89)	30.18 (±0.68)	40.40% (±0.99)	29.69 (±2.15)	37.12% (±0.43)	35.32 (±1.84)	51.89% (±1.02)	33.38 (±1.75)
50	10	2	1024	0.5	0.001	256	44.00% (±6.45)	34.08 (±5.89)	39.35% (±2.11)	28.14 (±2.68)	42.09% (±2.62)	29.72 (±0.97)	40.00% (±0.17)	36.11 (±0.91)	**54.57**% (±1.38)	**42.37** (±1.58)
50	5	5	1024	0.5	0.0001	256	**45.47**% (±5.02)	**35.25** (±3.47)	41.40% (±0.52)	30.33 (±0.70)	**46.91**% (±0.16)	34.04 (±0.74)	40.56% (±0.53)	37.62 (±0.51)	53.00% (±0.68)	39.01 (±0.19)
50	5	10	1024	0.5	0.0001	256	44.93% (±5.19)	33.99 (±4.59)	39.75% (±0.34)	26.84 (±0.40)	46.58% (±1.36)	33.90 (±0.94)	**40.84**% (±0.75)	**38.05** (±0.68)	52.54% (±1.02)	37.19 (±2.23)
30	5	5	1024	0.5	0.001	256	41.96% (±5.07)	33.77 (±3.78)	36.71% (±0.32)	27.98 (±1.49)	44.78% (±0.99)	35.57 (±0.90)	37.61% (±1.05)	34.67 (±2.44)	48.75% (±0.45)	36.87 (±1.05)
30	5	10	1024	0.5	0.001	256	42.20% (±5.29)	32.58 (±3.57)	37.50% (±0.43)	27.13 (±1.12)	45.45% (±0.48)	35.02 (±1.61)	36.91% (±2.24)	33.33 (±1.84)	48.94% (±0.69)	34.83 (±1.54)
30	5	2	1024	0.5	0.001	256	42.17% (±5.01)	32.45 (±3.44)	38.23% (±0.09)	27.57 (±0.95)	45.01% (±2.34)	33.34 (±1.49)	36.98% (±0.95)	33.61 (±0.99)	48.48% (±1.93)	35.29 (±3.05)
30	10	5	1024	0.5	0.001	256	41.68% (±5.75)	32.69 (±5.92)	36.18% (±0.73)	22.90 (±1.96)	45.45% (±2.23)	35.77 (±2.23)	36.07% (±0.87)	36.17 (±1.57)	49.03% (±0.60)	35.91 (±1.17)
30	10	10	1024	0.5	0.001	256	41.00% (±5.78)	31.32 (±4.44)	36.57% (±1.04)	26.30 (±2.13)	44.67% (±2.77)	35.45 (±3.22)	34.74% (±1.66)	32.07 (±4.24)	48.01% (±0.73)	31.46 (±1.76)
30	10	2	1024	0.5	0.001	256	43.66% (±5.85)	35.01 (±5.50)	37.37% (±0.73)	25.69 (±0.50)	46.24% (±1.11)	**37.88** (±0.90)	39.23% (±1.60)	37.27 (±1.42)	51.80% (±0.68)	39.19 (±0.19)

Table 4.10: Results of the experiments with the Multi-Scale Convolutional Neural Network and the inputs built using the segments approach for the action classification task. For each row and column, the average result is presented on top and the standard deviation is shown below. The highest column-wise mean is highlighted in bold.

Active object classification		
Approach	*Accuracy*	*F*1
Spatial stream (Ma et al., 2016)	61.87%	–
Spatial stream (after joint training) (Ma et al., 2016)	**74.34**%	–
Time Distributed Spatial CNN	58.92% (±6.05)	47.95 (±4.14)
MSCNN Raw (Ours)	52.47% (±4.08)	35.39 (±4.65)
MSCNN Segments (Ours)	57.40% (±6.05)	43.70 (±6.58)
Action classification		
Approach	*Accuracy*	*F*1
Dense Trajectories (DT) (Wang et al., 2013)	42.40%	–
Improved Dense Trajectories (IDT) (Wang and Schmid, 2013)	49.60%	–
O+E+H (Li et al., 2015)	57.40%	–
O+M+E+H (Li et al., 2015)	60.50%	–
O+M+E+G (Li et al., 2015)	60.30%	–
Motion stream (Ma et al., 2016)	62.62%	–
Two-stream (Ma et al., 2016)	**65.05**%	–
Two-stream (Wray et al., 2018)	59.02%	–
TSN (Sudhakaran and Lanz, 2018)	55.25%	–
Two-stream (Sudhakaran and Lanz, 2018)	60.13%	–
Appearance stream (Lu et al., 2019a)*	57.63%	–
Motion stream (Lu et al., 2019a)*	57.42%	–
Two-stream (Lu et al., 2019a)*	64.74%	–
Time Distributed Spatial CNN (Ours)	46.55% (±7.84)	36.11 (±4.44)
MSCNN Raw (Ours)	41.25% (±4.42)	27.70 (±2.27)
MSCNN Segments (Ours)	45.57% (±5.02)	33.99 (±4.59)

Table 4.11: Comparative table with the state-of-the-art approaches for active object and action classification using the GTEA Gaze+ dataset. The best results are highlighted in bold. The results are given as the mean of a leave-one-subject-out cross-validation approach. Furthermore, for our experiments we repeated each fold three times and we provide the mean and the standard deviation. *They took six subjects into account instead of four (see Section 4.1 for the evaluation strategy).

attended so that all the information they contain is better represented for the following task (such as active object or action classification, as in this case).

4.5 Hand-based attention masks

The Faster R-CNN's feature vectors contain many details about objects, although that information is encoded, requiring a system to extract patterns from them. Nevertheless, object detectors also provide human-understandable information such as BBs, which can be easily visualised. In fact, these boxes can be seen as a type of hard attention, in which specific regions of images are surrounded by "attention masks". Given the importance that hands have in egocentric settings, a BB around a hand or both hands can be seen as a way of attending the manipulation of objects or the interaction with them, as objects are supposed to be close to hands when they become active objects (Zhou et al., 2016b). In fact, this idea has already been explored in the literature in other forms. Ma et al. (2016) trained a network to segment hands, then a gaussian bump was created around hands, where objects should have lain. A gaussian bump can be interpreted as a soft BB around the location of hands. This part was cropped for object classification, removing any other spatial information. Lu et al. (2019a) aimed at creating attention masks using gaussian bumps as supervision. This may not be directly correlated to the previous approach, but both have something in common. In this approach, instead of the position of hands, they used the gaze position to create the gaussian bump. The main idea behind both works is that there are points in the space that are being attended by a human and so each one applied their own methodology to focus the attention on those regions.

In this part, an attention mechanism to exploit the attention point created by the position of hands is developed. For that, we leveraged the Faster R-CNN's predictions of Section 4.3 that included the prediction of hands. Looking back at Figure 4.8, the "hand" class predominated among others, which led to a high-recall hand detector.

4.5.1 Attention masks

An attention mask $A \in \mathbb{R}^{H \times W}$, with $A_{ij} \in [0,1]$, is a matrix with the spatial shape of the input of a neural network $X \in \mathbb{R}^{H \times W \times C}$, being H, W, and C the height, width, and the number of channels, respectively. Positions (i,j) with high values in this attention matrix A (close to 1) are positions that should be attended in the equivalent (i,j) positions of the original image X. When creating A, the BB coordinates of hands detected in X are taken into account. These coordinates (y_1, x_1) and (y_2, x_2) are two points in the space of X that can be used to draw a BB that includes a single hand (from the wrist to the fingers, not including the arm), being (y_1, x_1) the top-left corner point and (y_2, x_2) the bottom-right corner. In the case of the detection of two hands in a single frame, the BB that needs to be created would include the BBs of both hands. That is, a new BB with coordinates (y_1, x_1) and (y_2, x_2) must be created, using the coordinates (y'_1, x'_1) and (y'_2, x'_2), and (y''_1, x''_1) and (y''_2, x''_2) of the first and second hand BBs, respectively. To compute the coordinates of the new BB, the following formulas are used:

$$x_1 = min(x'_1, x''_1) \tag{4.5}$$

$$y_1 = min(y'_1, y''_1) \tag{4.6}$$

$$x_2 = max(x'_2, x''_2) \tag{4.7}$$

$$y_2 = max(y'_2, y''_2) \tag{4.8}$$

With this idea, a naive approach to create A would be to create a binary mask, in which all the pixel values of A within the BB would be set to 1 and any value outside of it to 0. However, we propose a softened version of the binary mask. It implies creating a 2D anisotropic gaussian bump centred in the BB with a sigma equal to the height and width of the BB. Those height and width are multiplied by a scaling factor of 0.5 in our experiments. In contrast to a binary mask, the attention mask is expanded through the whole image instead of just having null values out of the hand area. However, positions outside of the BB will have very small values, smaller as the distance to the

Figure 4.18: Gaussian masks applied to images of the EGTEA Gaze+ dataset.

centre of the BB (or the gaussian bump) increases. Sample masks can be seen applied to images of the EGTEA Gaze+ dataset in Figure 4.18. To obtain these representations, an element-wise product between A and X is done, for visualisation purposes.

If a single input contains several frames (as in the case of a stack of OF), then the mask that needs to be created for that input would be based on a BB that includes all the BBs of the frames of the stack. This can be seen as an extension of Equations 4.5, 4.6, 4.7, and 4.8 to more than two terms, as many as frames are stacked. If the mask is applied to an input that consists of a single frame, then the mask is directly computed with the BB of that frame.

4.5.2 Architecture

So far, the CNN-M-2048 has been employed as an off-the-shelf DL architecture for the feature extraction part, apart from the specialised architectures of Section 4.4.2 (due to the type of data they used). However, the former only

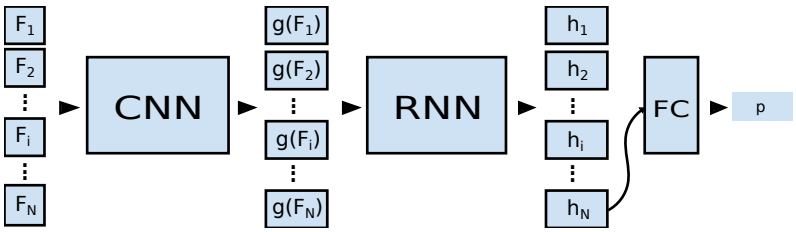


Figure 4.19: The Convolutional Neural Network (CNN) - Recurrent Neural Network (RNN) paradigm has two main parts: the feature extractor g, implemented as a CNN, and the long-term modeling RNN. The system is fed with a sequence of frames $\{F_1, F_2, ...F_N\}$ sampled from a video, being N the amount of frames. Each frame is transformed by g, extracting the features $g(F_i)$ for each frame F_i. A RNN learns long-term temporal patterns, predicting the $\{h_1, h_2, ...h_N\}$ hidden states. The last hidden state, h_N, is sent to the FC layer (the classifier) and a prediction is given, a probability distribution p over a set of classes.

extracts short-term patterns and can only provide predictions for individual timesteps of clips, leaving another function (such as the majority voting) the task of deciding the prediction for the whole clip. To go one step further and inspired by the work of Sudhakaran and Lanz (2018), the architecture used for the experiments of this section will include a long-term modeling of clips. The type of structure suggested for that in the literature is a CNN-RNN similar to that of Sudhakaran and Lanz (2018). Figure 4.19 illustrates the general scheme followed by CNN-RNN architectures. Given a sequence of frames $\{F_1, F_2, ...F_N\}$ (being N the number of frames), a feature extractor g (implemented with a CNN) extracts features for each frame F_i $(0 \leq i \leq N)$, obtaining a sequence of features $\{g(F_1), g(F_2), ...g(F_N)\}$. These are passed to the RNN to get a sequence of temporal or spatiotemporal features $\{h_1, h_2, ...h_N\}$, in which h_i represents the information of the clip until the i^{th} step. Usually, the last one, h_N, is sent to the classifier, which outputs a probability distribution.

Going into details, the chosen CNN architecture is a ResNet50 network (He et al., 2016), a standard architecture in the literature. The feature cube $g(F_i)$ obtained from the network has a high dimensionality in the depth dimension due to the network applying many kernels. To alleviate the task for the RNN module, a 1×1 convolution with 256 kernels is applied so that the depth dimension reduces to 256. The RNN that takes this sequence is a ConvLSTM

(Xingjian et al., 2015), whose hidden states h_i are 3D structures containing spatiotemporal patterns. The FC layer has a number of units that matches the number of classes of the problem and applies a softmax activation to obtain a normalised distribution. Formally, the output is $p = \{p_1, p_2, ..., p_M\}$, where $p_i \in [0, 1]$ is the probability of the class i, $\sum_{i=1}^{M} p_i = 1$, and M is the number of classes. The predicted class is computed as $max_i\{p_i\}$, where i is the index of a class.

To apply the masking operation, we set the mask at each timestep as another input to the system. Each mask will be multiplied element-wise with its corresponding feature cube, applying the same spatial mask to each channel. For that, the mask is resized to match the spatial dimensionality of the feature cube. Moreover, to disturb as little as possible the pre-training of the feature extractor, the ResNet50, the multiplication is performed with the last feature cube of the network, $g(F_i)$, in the final convolution (prior to the 1×1 convolution). More formally, given a feature cube $g(F_i)$ of shape $H' \times W' \times C'$ from the last convolution and a mask M of shape $H \times W$, a resizing r and a depth-wise broadcasting b is applied to the mask to match the dimensionality of $g(F_i)$, i.e., $r(M) \in \mathbb{R}^{H' \times W'}$ and $b(r(M)) \in \mathbb{R}^{H' \times W' \times C'}$. Then, the Hadamard product is applied between $g(F_i)$ and $b(r(M))$ to get $g(F_i)'$, a masked feature cube of shape $H' \times W' \times C'$. This will be fed to the 1×1 convolution as the next step in the architecture, and so on.

4.5.3 **Experiments**

So far the GTEA Gaze+ dataset has been used for the experiments. However, with the launch of the EGTEA Gaze+ dataset, a larger version of the previous dataset, this was adopted for the last set of experiments of the chapter and the ones of the following one. As EGTEA Gaze+ contains GTEA Gaze+, the statements done so far hold, as the former just extends the latter with more classes and samples. In fact, as both datasets are similar, the hand object detector trained for GTEA Gaze+ can be reused. Otherwise, training a Faster R-CNN for the EGTEA Gaze+ dataset would imply manually annotating object BBs again.

The following sets of experiments are carried out in this part: one set of experiments aiming at classifying active objects and the other one actions. For this problem, we defined the following TP, FP, TN, and FN for the accuracy and F1 metrics:

- TP: for the class i, if a sequence of images is labeled with class i and the class i is the one with the highest confidence.
- FP: for the class i, if a sequence of images is labeled with class j and the class i is the one with the highest confidence, being j any class different to i.
- TN: for the class i, if a sequence of images is labeled with class j and the class k is the one with the highest confidence, being j and k any classes different to i and being possible that $j = k$.
- FN: for the class i, if a sequence of images is labeled with class i and the class j is the one with the highest confidence, being j any class different to i.

For the training, 500 epochs are set with early stopping with a patience of 10 epochs using the Macro-F1 metric in the validation set. The latter is built using a stratified set containing 10% of the training samples. For the object classification task, the validation set is built stratifying according to the object label and, for the action classification task, using the action tag. The Adam optimiser is used with a learning rate of 10^{-5}, a batch size of 16 and 25 timesteps per sample. The results are given per split and as the average of the three splits. For each split, three consecutive experiments are averaged. For the active object classification, the results are summarised in Table 4.12 and, for the action classification, in Table 4.13.

So far, our results using gaussian masks have shown a consistent slight improvement of the results in every single experiment, both for active object and action classification. Specially for the former, the accuracy increased by 1%, similar to new state-of-the-art results in other Computer Vision tasks (Tran and Cheong, 2017) (Zhang et al., 2020) (Li et al., 2020). The use of

	Average		*Split* 1		*Split* 2		*Split* 3	
Experiment	*Accuracy*	*F*1	*Accuracy*	*F*1	*Accuracy*	*F*1	*Accuracy*	*F*1
Baseline	66.66%	59.57	69.39%	63.99	66.11%	57.18	64.49%	57.53
	(±2.25)	(±3.32)	(±0.84)	(±1.30)	(±0.87)	(±0.64)	(±1.16)	(±1.28)
Gaussian	67.49%	60.74	70.85%	65.75	66.50%	57.34	65.10%	59.14
mask	(±2.52)	(±3.82)	(±0.77)	(±1.43)	(±0.69)	(±1.31)	(±0.23)	(±0.84)

Table 4.12: Results of the experiments for object classification with gaussian masks. For each row and column, the average result is presented on top and the standard deviation is shown below.

	Average		*Split* 1		*Split* 2		*Split* 3	
Experiment	*Accuracy*	*F*1	*Accuracy*	*F*1	*Accuracy*	*F*1	*Accuracy*	*F*1
Baseline	55.32%	44.78	58.21%	47.99	55.56%	44.12	52.20%	42.24
	(±2.60)	(±2.52)	(±0.69)	(±0.16)	(±1.26)	(±0.81)	(±0.35)	(±1.06)
Gaussian	55.98%	45.12	58.37%	47.49	56.46%	44.96	53.11%	42.91
mask	(±2.26)	(±2.20)	(±0.56)	(±0.62)	(±0.31)	(±0.16)	(±0.87)	(±1.90)

Table 4.13: Results of the experiments for action classification with gaussian masks. For each row and column, the average result is presented on top and the standard deviation is shown below.

gaussian bumps as an attention mechanism has already been explored in the literature (Li et al., 2018) (Zuo et al., 2018) as a supervision signal to generate a mask that is later applied to the features, as in our case. Ma et al. (2016) generated gaussian bumps to crop RoIs of images instead. Nonetheless, to the best of our knowledge, we were the first ones implementing it using an object detector to recognise hands and applying the attention mask generated from the position of the hands.

Nevertheless, the results are not as high as expected. The possible reasons for not obtaining higher results can be attributed to (i) the performance of the hand detector, (ii) the quality of some videos of the dataset (due to the lighting, hands are not correctly detected), or (iii) the implementation of the mask in the network, although the reason is not clear. Regarding the latter issue, the specific hyper-parameters to build the mask and apply it may not be the most appropriate ones (although there has been an exploration of some configurations) or it may be that not using masks when no hands are detected or changing the features mid-way could destabilise the training. Others au-

thors applied a mask generated by the network itself, probably avoiding this issue. Another possible reason for the latter approach to perform better is that guiding the training of the feature extractor for the objective of recognising active objects or actions alongside predicting attention masks may be the key to improve this type of attention mechanism.

For the sake of comparison, Table 4.14 compares the best result obtained on actions using gaussian masks with other state-of-the-art approaches on the EGTEA Gaze+ dataset. To the best of our knowledge, there were no reported results of active object classification for the EGTEA Gaze+ dataset and, thus, we only show the comparison for action classification. We did improve or equal the result of some off-the-shelf state-of-the-art networks such as I3D and TSN, while other approaches employing more information obtained up to 5-6 points of accuracy more than us. One of the best results, obtained by Kapidis et al. (2019a), just employing RGB images like us, went up to 68.99% of accuracy (although just in the first split of the dataset). They argue that a multi-tasking approach forces the network to generalise better, learning verbs, objects, coordinates for hand locations, and the gaze-based visual saliency at the same time. Similarly, for the Gaze+ dataset, (Ma et al., 2016) learnt verbs, objects, and actions and obtained the best result so far for that dataset. This could point towards the use of multi-tasking approaches as a promising approach for the EAR and, possibly, for the active object detection. Other approaches made use of two-stream architectures including flow information or even object detection features (Wang et al., 2020) and/or architectures that include attention, such as the LSTA (Sudhakaran et al., 2019c) or the two-stream network of Lu et al. (2019b).

4.6 **Summary and Conclusions**

In this chapter several proposals to leverage object or visual information for active object and action classification have been proposed. Starting from the baseline SpatialNet to infer active objects and actions, then the Faster R-CNN to predict objects spatially, followed by the BOM representation and the proposed MSCNN architecture to exploit it, and, finally, the use of spatial

Action classification		
Approach	*Accuracy*	*F*1
Single-stream, RGB and flow inputs (Li et al., 2018)	53.30%	–
Two-stream (Verma et al., 2018)	66.00%	–
Original two-stream (Sudhakaran and Lanz, 2018)	41.84%	–
I3D (Sudhakaran and Lanz, 2018)	51.68%	–
TSN (Sudhakaran and Lanz, 2018)	55.93%	–
Two-stream (Sudhakaran and Lanz, 2018)	60.76%	–
LSTA (Sudhakaran et al., 2019c)	61.86%	–
Multi-tasking network (Kapidis et al., 2019a)*	**68.99%**	–
Two-stream with STAM (Lu et al., 2019b)	68.60%	–
Multi-stream with SAP (Wang et al., 2020)	62.70%	–
CNN-ConvLSTM with hands-aware attention (Ours)	55.98% (±2.26)	45.12 (±2.20)

Table 4.14: Comparative table with the state-of-the-art approaches for action classification using the EGTEA Gaze+ dataset. The best results are highlighted in bold. For our experiments we repeated each fold three times and we provide the mean and the standard deviation. *Reports results only for the split 1.

gaussian masks to focus the attention in regions where hands' presence is detected.

After all the experiments, we concluded that the recognition of active objects is not trivial and that more research is required on this topic. However, some valuable observations can be extracted from this chapter.

- The **temporal modeling of videos** must be done carefully. In Chapter 3, the evaluation for video was not covered but, in Section 4.2, our first approach included a mean pooling of the predictions of single inputs to provide an average prediction for the video. However, the latter highly depended on the local patterns extracted, with no correlation between the subsequent timesteps apart from the mean pooling. We believe the addition of the ConvLSTM to the network proposed in Section 4.5 was an step in the correct direction.

- The **object feature vectors from RPN networks seem to be a promising complementary information** that can be used alongside

RGB and optical flow features in the usual multi-stream approach. Furthermore, going one step further and combining object feature vectors with features extracted from other streams may lead to competitive results (Wang et al., 2019c).

- **Attention mechanisms** have increasingly become more popular as a way to improve results without scaling networks to become bigger and adding little computation and memory overhead in general. The type of attention included in Section 4.5 was our first attempt in introducing a type of attention that does not require specific annotations in the dataset such as gaze points. Although a hand detector is still required for this, it could be obtained from other bigger datasets without requiring any fine-tuning.

- **Multi-tasking approaches may lead to improved generalisation**, as demonstrated by the state-of-the-art results obtained by Ma et al. (2016) and Kapidis et al. (2019a) in the GTEA Gaze+ and EGTEA Gaze+ datasets, respectively. In fact, aiming at learning egocentric features, verbs, objects, and actions labels at the same time, following the literature of the EAR field, may be the key to improve active object recognition and EAR even further.

I may not have gone where I intended to go, but I think
I have ended up where I intended to be.

Douglas Adams

CHAPTER

5

An egocentric action classification system with external knowledge

So far the attention has been fixed in the disentanglement of actions, in verbs and objects, and their respective modeling, with separate models and objectives, even inferring actions from objects or the visual appearance, a highly popularised strategy in the state of the art. Joining verb and object predictions is also a straightforward path to action prediction. Nevertheless, this is not that trivial, as the cohesion or relation between verbs and objects may not be taken into account, i.e., both are inferred assuming their independence. This chapter will go deeper into the action prediction: how the verb and object predictions can be combined taking into account their high-level relation and how never seen actions can be inferred, a very desirable property for AAL applications, allowing for action prediction systems to adapt to the real-world action frequencies.

The chapter is divided into four sections. First, Section 5.1 introduces the zero-shot approach for EAR and its motivation, the proposed architecture for ZS-EAR, and the use of external knowledge to bias the system towards

real-world frequencies. Section 5.2 defines the data splits that are going to be used for the zero-shot approach and the experiments that will be carried out, whilst Section 5.3 presents the results obtained. Finally, Section 5.4 provides the conclusions of the chapter.

5.1 The zero-shot learning approach to predicting actions

In a real-world setting, action inferring systems implemented for wearable devices aim at providing the best possible prediction at each timestep. Many types of interactions between humans and objects must be recognised by them. In fact, those systems are manufactured to serve for many purposes, but in the scope of this dissertation, we focus on those related to healthcare, contained within the AAL paradigm, in which ADLs are the core. Common daily actions such as washing the dishes, taking food from the fridge, cleaning the house, and many others are supposed to be taken into account by developers of those kind of systems due to the frequency with which they are performed. For supervised systems, to be able to infer, first, datasets that include labeled actions are required to train a system. The huge amount of data required to correctly learn actions is difficult to gather and laborious to label, and even more challenging if a large amount of data is required for each action. It can happen that, within our daily lives, there are some rare actions that these systems are not trained to infer, as their frequency is very low. Getting data for those actions can become a serious issue.

To alleviate the latter problem, the ZSL paradigm can be applied to the EAR task. The basis of this approach is that labels (and the data with those labels) never used during the training phase can be predicted at inference time. This allows for a wider range of actions to be predicted at inference time, a very desirable property in AAL applications. There are many ZSL proposals within the EAR literature, reviewed in Section 2.2, but this work was inspired by the exocentric work of Shen et al. (2018). The authors propose a branched system: one branch is in charge of learning the patterns related to the verb

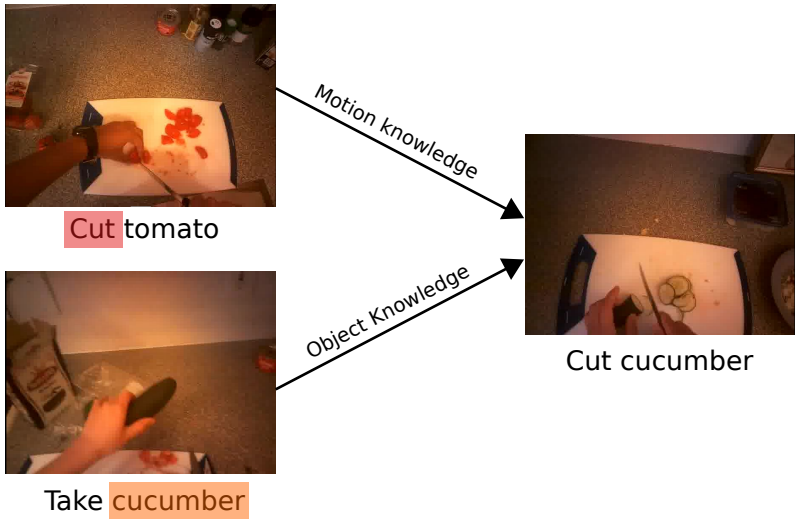


Figure 5.1: In our branched zero-shot architecture, the knowledge acquired from learning objects and verbs (motion) separately is used at inference time to discover combinations never seen during the learning phase. For example, the motion associated to the verb ”cut” can be learnt from actions such as ”cut tomato” and the appearance of a ”cucumber” from actions such as ”take cucumber”. The new action ”cut cucumber”, that have never been used in the learning phase, can be inferred from this knowledge.

labels or the motion of actions, while the other is in charge of learning the patterns of the active objects of actions. Both parts are joined at the end, one providing a verb and the other one an object, creating an action when combined. For example, if the verb branch predicts the verb ”take” and the object branch the ”pepper” object, then the action would be ”take pepper”. This means that the knowledge acquired from each branch is used separately at inference time. Figure 5.1 illustrates an example of this idea. With this approach, using two models, one capable of inferring $|V|$ verbs (from a set V) and the other one $|O|$ objects (from a set O), the final system would be able to predict $|V| \times |O|$ actions only requiring $|V| + |O|$ labels. Even though this high-level idea of having two separate streams for verbs and objects was explored by several authors in the EAR literature, to the best of our knowledge, the exocentric work of Shen et al. (2018) was the first one to apply it to the ZSL.

Going in the same direction, we propose a similar system in this chapter. Nonetheless, our aim is not to build the best possible action inferring system,

that is out of the scope of this dissertation. In contrast, this work's contribution uses this system as a basis. That is, we propose a method for any ZS-EAR approach that follows the two-stream paradigm (verb and object streams) to improve its results leveraging external knowledge. This section will go deeper, point by point, on the system's structure and the use of external knowledge.

5.1.1 Two-stream action inferring system

In general, a two-stream architecture is a network with two streams or branches, each one with its own weights to be learnt, which can be jointly learnt or not. For the ZSL approach, both streams remain separate and do not share weights. In our approach, the structure of both streams is based on the CNN - RNN paradigm, in which a CNN is in charge of extracting features, a RNN learns temporal or spatiotemporal patterns given the previous CNN's features, and, finally, a FC layer learns how to classify the sequence representation from the RNN into various categories, verbs or objects, depending on the task. More formally, the system takes as input a sequence $I = \{F_1, F_2, ...F_N\}$, where F_i is the i^{th} frame of a sequence ($0 < i < N$, being N the sequence length) and $F_i \in \mathbb{R}^{H \times W \times C}$. The input is fed to a feature extractor g (a CNN), frame by frame, obtaining a sequence $\{g(F_1), g(F_2), ...g(F_N)\}$ of the same length, where $g(F_i)$ is the output of the CNN for the i^{th} frame after a forward pass. This sequence is fed to the RNN module, which tries to learn temporal or spatiotemporal patterns. For each timestep of the sequence, a hidden state h_i ($0 < i < N$) is obtained, representing the temporal features of the sequence until the i^{th} timestep. The output of the module is the last hidden state h_N, which is fed to the FC classifier. The final output is $p^v(v_{a_j})$, the verb prediction for the j^{th} action with a vision-based model, or $p^v(o_{a_j})$, the analogous for objects, depending on the task. Figure 4.19 of Section 4.5.2 summarises the high-level structure, as it is the same architecture used in that part (excluding the attention map).

Going into details, the chosen CNN architecture is the ResNet50 network (He et al., 2016), a standard architecture in the literature. The feature cube

$g(F_i)$ obtained from the network has a high dimensionality in the depth dimension due to the network applying many kernels. To alleviate the task for the RNN module, a 1×1 convolution with 256 kernels is applied so that the depth dimension reduces to 256. The RNN that takes this sequence is a Convolutional Long-Short Term Memory (ConvLSTM) (Xingjian et al., 2015), whose hidden states h_i are 3D structures containing spatiotemporal patterns. The FC layer has a number of units that matches the number of classes of the problem and applies a softmax activation to obtain a normalised distribution. Formally, the output is $p = \{p_1, p_2, ..., p_M\}$, where $p_i \in [0, 1]$ is the probability of the class i, $\sum_{i=1}^{M} p_i = 1$, and M is the number of classes. The verb class is computed as $max_i\{p^v(v_{a_j})\}$ and the object class as $max_i\{p^v(o_{a_j})\}$. We define the systems as D_V, the verb detector capable of inferring verbs from a set V, and D_O, the object detector that predicts objects from a set O.

With both detectors defined, the prediction of the action is straightforward. The Cartesian product of the verb and object probability distributions is computed so that, for each possible combination, a probability is obtained. More formally, for a given input sequence I, the probability of an input being of class action a_j is computed as $p^v(a_j) = p^v(o_{a_j}) \times p^v(v_{a_j})$, where $p^v(v_{a_j}) = D_V(I)_k$ (the probability of the k^{th} verb, which belongs to the action a_j) and $p^v(o_{a_j}) = D_O(I)_l$ (the probability of the l^{th} object, which belongs to the action a_j). So, the predicted action for the input I would be $max_j\{p^v(a_j)\}$, the action with the highest probability.

5.1.2 Using external knowledge priors

Using the previously explained action inferring system, it can be observed that any possible combination (an its probability) can be predicted, allowing for a wide range of predictions. The issue that arises from this property is that there may be combinations that do not exist or that are not frequent. Moreover, as it is, a rare action may have a probability that is close to that of a daily living action, which is not desirable for an application that aims at predicting ADLs. That is, just computing $max_j\{p^v(a_j)\}$ might not adjust to the reality, and it is natural that it works this way, as action inference systems

only predict the confidence they have on their predictions given some input. That is why another source of information apart from the vision-based model could potentially help to resolve this. This source of information or knowledge (external to the vision-based model) must contain a real-world bias, or better said, its inherent action frequency must be adapted for real-world tasks. This claim also suggests that it should be independent of the input to the model. Our contribution here is to use text corpora as the external knowledge to create another probability distribution, $p^t(a)$, which we call the action prior.

$p^t(a)$ models the probability distribution created from the Cartesian product between the set of verbs V and the set of objects O, formally defined as $\{a_i = (v_j, o_k) : \forall j, \forall k | v_j \in V \ and \ o_k \in O\}$. Thus, for each possible action a_i, a probability $p^t(a_i)$ is computed from external corpora. To combine this with the vision-based model, we propose the element-wise multiplication of both distributions, the probability of the vision-based model and the one of the external corpora, i.e., $p(a_i) = p^t(a_i) \times p^v(a_i)$. Hence, the final action prediction would be computed as $max_i\{p^t(a_i) \times p^v(a_i)\}$. Figure 5.2 illustrates the full system.

Nevertheless, there is a missing aspect about using external corpora that must be taken into account. A corpus introduces a domain bias in the task, i.e., in a given corpus there is an inherent action probability distribution, different in each corpus. Depending on the topic or domain of corpora, distributions may vary significantly between each other and, therefore, the effect of each external knowledge source will be different. To analyse this phenomenon, various sources are proposed:

- Cookbook wiki: using the Cookbook wiki page [1], we extracted a corpus containing recipes and, thus, actions related to cooking recipes. We selected this knowledge source to further narrow the domain of the egocentric videos and see how specialised knowledge can help for ZSL.

- Google searcher API[2]: we used the API to search for actions and get the number of results as the number of occurrences. This knowledge source

[1] https://en.wikibooks.org/wiki/Cookbook:Recipes
[2] https://developers.google.com/custom-search/v1/overview

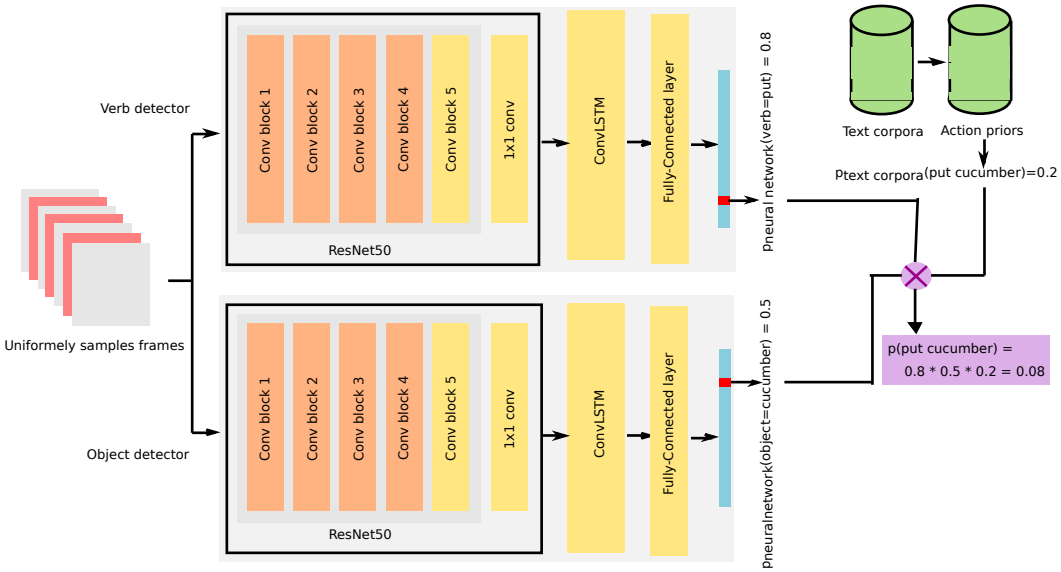


Figure 5.2: Architecture overview: two neural networks composed of a ResNet50 and a ConvLSTM take as input a video (uniformly sampled frames) and output two probability distributions (of verbs and objects). The Cartesian product of these two result in an action probability distribution. This is combined with an action prior sampled from text corpora to infer the most probable action. The layers or blocks of layers in orange are frozen while the yellow ones are trained. An example with the action "put cucumber" is provided in the illustration.

was chosen to have a more general prior estimation which is not focused on a specific domain, in contrast to the Cookbook wiki.

- *Phrasefinder* searcher API[1]: similar to the Google API, the *Phrasefinder* source has no specific domain, as it searches through Google Books' N-grams. We chose this as a more controlled alternative to the Google API prior, whose results come from a wilder environment (any site indexed in Google).

To create the actual prior, in the case of the Cookbook wiki, the wiki was scrapped to obtain the corpus and, then, this was cleaned. After that, non-ASCII characters were removed, the text was lowercased, and stop words were eliminated. Finally, the WordNet lemmatiser[2] was applied to obtain the roots of all the words. This is an important step towards homogenising actions, as the next step is to find each possible action coming from the EGTEA Gaze+ dataset. For that, instead of employing the full corpus, N-grams of size 4 were extracted. We took into account that articles, determiners and quantifiers (among others) may appear together with actions and, hence, the length of the N-gram is larger than the size of the actions of the dataset (in terms of words). To determine that an action appears in a N-gram, both the verb and the object must appear in it, not necessarily in adjacent positions. For example, in the N-gram "to cut a pepper", the action "cut pepper" appears, as both "cut" and "pepper" are contained within the N-gram. In fact, instead of just taking the verb and the object as they are, we manually defined a list of synonyms for each one. For each possible synonym of a verb and an object, if any combination is contained within the N-gram, this would also be a valid appearance. Following with the previous example, if there is not an N-gram with "cut pepper" but there is one containing "slice pepper", then we assume that "cut pepper" appears. This can be seen as having a representative name for an action ("cut pepper"), but different possible representations such as "slice pepper" or "mince pepper". Finally, for any action, its prior is computed taking the

[1] https://phrasefinder.io/api

[2] http://www.nltk.org/_modules/nltk/stem/wordnet.html#WordNetLemmatizer

number of N-grams in which it appeared divided by the total number of N-grams of the corpus.

For the Google and *Phrasefinder* sources, the priors were built using their respective APIs, i.e., depending on the number of results of the queries done per action. A query is created with an expression "verb * object" with the Google API and "verb ? object" with the *Phrasefinder* API (the "verb" and "object" strings are just placeholders for the actual verb and object names). The "*" and "?" symbols are wildcards or placeholders that the APIs use, they can be replaced by strings such as "a", "the", and so on, allowing for natural searches, as in the case of the Cookbook wiki, in which N-grams of bigger size allowed to include articles, determiners, quantifiers, and so forth. Again, we used synonyms of each verb and object, obtaining a number of results per combination of synonyms. The value of the action is computed with the mean of all the non-zero results of their possible representations' results (synonym combinations). Then, this value is normalised by the sum of all actions' values to obtain the action prior.

In addition, to estimate a theoretical upper bound of the proposed system, a so called *perfect prior* has been created, in which the probability distribution matches that of the test set. That is, the frequency of an action is equal to the number of videos of the action divided by the total number of videos. Although very informative, this upper bound would be empirically unreachable by other means, as the perfect prior is unique to the specific data distribution from which it has been sampled. In other words, it is the best possible prior to maximise the performance of the system.

5.2 **Experimental setup**

This section describes the necessary elements to define the experiments of the next section. First, the proposed data splits of the dataset for the zero-shot setting are presented in Section 5.2.1. Then, the set of experiments are defined in Section 5.2.2.

5.2.1 **Zero-shot splits**

The EGTEA Gaze+ has three official train and test splits, used in the literature to evaluate EAR proposals. The test sets of each of the three possible divisions, created from stratified subsets of samples, share the action classes with their respective training sets. As all the classes are represented in both the training and the test set, we believe these splits are not appropriate for the ZSL task and, hence, new splits are proposed for the experiments. For that, we follow the guidelines proposed by Shen et al. (2018) for another problem. First, the videos containing verb and objects that only appear in one action are removed, as they are not suitable for ZSL tasks. The reason is that, even if those discarded verbs or objects were learnt during training phase, its knowledge could not be evaluated at test time, as there is not another action class with those verbs or objects. This left the dataset with 9 verbs and 29 objects. Second, to generate the test set, 20% of the classes were randomly chosen under the condition that all the verbs and objects must appear in the training set. For example, if all the actions containing the object "cucumber" only appear in the test set, the split is not correct, as the object cannot be learnt. The same reason of discarding classes in the first condition applies here as well, verb and objects must appear in the training phase to be learnt and, then, in the test set to be evaluated. However, the action classes must be disjoint between both sets. The validation set is created taking 10% of the samples from the training set, following a stratified distribution and, thus, sharing the relative class distribution with the training set. Note that this validation set is not important for the ZSL task, but rather for the training of the detectors D_V and D_O.

To test the effect of the domain bias and the generic knowledge sources in the system, we proposed two types of splits that are based on the previously explained train/test split: the Recipe split (R split) and the Non-Recipe split (NR split). The first one, the R split, is built by explicitly discarding some verbs, objects, and actions that are not related to recipes. Specifically, we banned the verb *inspect/read*, the objects *cabinet*, *sponge*, *grocery bag*, *eating utensil*, *drawer*, and *fridge drawer* and the action *wash pan*. The intuition

behind this is that the test set of the R split would fit into the domain of actions related to recipes and, thus, a bias towards recipes in the applied action prior (for instance, the Cookbook prior) should be significantly effective. Under these rules, the instance of this split we created had 6121 training videos and 1464 test videos. The second split, the NR split, avoids any bias, i.e., we did not impose any other constraint apart from the ones given by Shen et al. (2018). In the case of the generated instance of this split, it had 6277 training videos and 1308 test videos.

5.2.2 Experiments

To validate the hypothesis proposed in this dissertation, i.e., that using external knowledge extracted from web corpora can potentially improve the standard metrics of the ZS-EAR, several experiments are proposed. We set as a baseline a system which only relies in D_V and D_O to infer actions, without using external corpora, predicting the class given by the expression $max_j\{p^v(a_i)\}$. Moreover, we have three action priors to test (the Cookbook, Google, and *Phrasefinder* priors) and two ZSL splits (the R and NR splits).

For any split, both detectors, D_V and D_O, are trained for $\mathbf{100}$ epochs with a patience of $\mathbf{10}$ epochs (the macro-F1 metric of the validation set is used to stop the training). The feature extractor's CNN's weights are initialised with an Imagenet pre-training and frozen up to the 4^{th} convolutional block. Only the layers from the 5^{th} block onwards (including the latter) are fine-tuned. We use the Adam optimiser with a batch size of 16, the initial learning rate set to $1e^{-4}$, and 25 uniformly sampled timesteps per video. To diminish the effect of the overfitting, we employ class weights and data augmentation in the training phase. The class weights are applied to the loss function to penalise errors in classes with fewer samples and are computed from the train set distribution of videos, taking the inverse of that distribution. For the data augmentation, standard random horizontal flipping and multi-scale random corner cropping are used. The latter crops one patch randomly selecting one of the five possible positions (the four corners and the centre). The initial

crop size is 224×224, which is scaled with a random factor chosen among 1, 0.875, 0.75, and 0.65625. Finally, the crop is scaled back to 224×224.

In Chapter 3 we saw that OF driven CNNs could be a promising approach for the learning of motion. That is why we propose to create another verb detector D'_V that leverages OF information. We add another stream to the previously proposed verb detector D_V to create a two-stream architecture. Due to the size of the network, the number of timesteps is reduced to 5. For synchronisation purposes, another object detector D'_O was trained using 5 timesteps as well. The same training configuration of the D_V and D_O detectors is used for D'_V and D'_O.

5.3 Experiments and Discussion

This section presents the results of the proposed experiments in Section 5.2.2, beginning with the results of D_V, D'_V, D_O, and D'_O in the verb and object classification tasks, followed by the action prediction results with and without action priors. These have been divided into three parts: (i) the results per split (R and NR) and per prior source (Cookbook, Google, and *Phrasefinder*), (ii) results per class in the R split, and (iii) results per class in the NR split.

5.3.1 Verb and object detectors

Beginning with the verb and object classification, the results for D_V and D_O can be seen in Tables 5.1 (for verbs) and 5.2 (for objects). The results are given per split (row-wise) for each of the sets (column-wise): training, validation, and testing. The metrics shown are those of Section 4.1. The mean of three runs alongside the standard deviation are shown in each cell. As it can be observed, the performance in the test set suffers a significant drop compared to the validation set's performance. We hypothesise that this phenomenon could be explained by the variance of verb and object classes, i.e., the different shapes and poses objects have or the movement variations within each of the verb classes. If not enough of the variance is presented in the training set, the model won't be able to capture the true nature of the verb or object

Verb detector D_V	Split	Train		Validation		Test	
		Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
	R	99.37%	99.22	75.69%	67.93	60.31%	31.08
		(±0.10)	(±0.19)	(±0.35)	(±0.58)	(±1.10)	(±0.06)
	NR	98.44%	97.64	76.65%	66.42	53.49%	42.90
		(±0.61)	(±0.80)	(±0.42)	(±3.11)	(±1.44)	(±1.46)

Table 5.1: Verb classification results with the D_V verb detector. The results are given as the mean of 3 runs. For each row and column, the average result is presented on top and the standard deviation is shown below.

they are aiming to learn. This variance is highly correlated with the verb in the case of objects and for the object in the case of verbs. For instance, a "tomato" observed alongside the verb "take" may look completely different from a tomato that is being "cut", specially since an object like a tomato, apart from being occluded, being far, or any other variation with respect to the camera, it can also change its shape after being sliced. It is also natural that two objects being cut do not generate the same motion, for example a tomato may be sliced while it is on a cutting board and a pepper can be sliced while it is being held, without touching the counter nor the cutting board. Learning these variations is important for the ZS-EAR, but it is out of the scope of this dissertation to go further into improving the verb and object detectors. Nevertheless, the object classification task is the one that has the highest difference between the validation and testing sets; this observation suggests that the active object detection may be highly correlated with the verb and, thus, active object detectors are the ones that suffer the most in ZSL conditions. Finally, the difference in the performance between D_V and D_O could also be explained by the number of classes each of them has to learn, being D_V the one that has fewer classes and, hence, higher results.

The results obtained with the second proposed verb detector D'_V are presented in Table 5.3. There is a significant improvement in the test performance compared to the single-stream verb detector, shortening the gap between the validation and the test sets. The results for the new object detector D'_O can be seen in Table 5.4. These are quite similar to those of Table 5.2 (the pre-

Object detector D_O	Split	Train		Validation		Test	
		Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
	R	99.37% (±0.10)	99.22 (±0.19)	75.69% (±0.35)	67.93 (±0.58)	27.14% (±0.74)	13.48 (±0.73)
	NR	98.70% (±0.56)	98.35 (±0.76)	76.43% (±0.34)	68.64 (±0.39)	31.75% (±0.81)	15.34 (±0.51)

Table 5.2: Active object classification results with the D_O object detector. The results are given as the mean of 3 runs. For each row and column, the average result is presented on top and the standard deviation is shown below.

Verb detector D'_V	Split	Train		Validation		Test	
		Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
	R	98.59% (±1.12)	96.72 (±2.90)	76.56% (±0.66)	69.99 (±0.567)	64.12% (±2.31)	36.39 (±0.92)
	NR	99.16% (±0.51)	94.55 (±3.36)	74.84% (±0.13)	55.40 (±0.32)	56.70% (±1.41)	48.56 (±0.70)

Table 5.3: Verb classification results with the D'_V verb detector with 5 timesteps. D'_V is built as a two-stream network. The results are given as the mean of 3 runs. For each row and column, the average result is presented on top and the standard deviation is shown below.

vious object detector), allowing for a better comparison of models for action inference in the next part. That is, we will combine these two last detectors to create a second action inference system that employs OF images in the verb detector's learning and prediction.

5.3.2 Action inference

Results by set. Leveraging the presented detectors D_V, D_O, D'_V, and D'_O, the action classification experiments were carried out. The results per split (row-wise) are shown in Tables 5.5 (using D_V and D_O) and 5.6 (using D'_V and D'_O), including the baseline explained in Section 5.2.2 and the experiments with each action prior. The metrics in bold highlight the best value per metric and per row. As in Chapter 4, the best accuracy and macro-F1 results

Object detector D'_O	Split	Train		Validation		Test	
		Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
	R	99.77%	99.73	73.68%	65.01	29.37%	13.77
		(±0.07)	(±0.04)	(±0.41)	(±0.45)	(±0.40)	(±0.51)
	NR	99.76%	99.70	75.48%	69.43	31.52%	15.03
		(±0.04)	(±0.03)	(±0.34)	(±0.31)	(±1.78)	(±0.67)

Table 5.4: Active object classification results with the D'_O object detector with 5 timesteps. The results are given as the mean of 3 runs. For each row and column, the average result is presented on top and the standard deviation is shown below.

are considered those with the highest average result, not taking into account the standard deviation (see Section 4.4.3).

For the action inference system A built using the D_V and D_O detectors (Table 5.5), the R split row of the table shows overall higher results than the NR splits row. Starting from the baseline, there is a significant difference of more than 4 points in accuracy and 5 points in F1 between R and NR. The chosen actions can greatly influence this baseline, as there are verbs and objects that do not have a high accuracy and, thus, the prediction of the actions related to them is hampered. Regarding the action priors involved, our hypothesis that the R split would benefit the most from a prior of its same domain seems promising given the results that were obtained: there is an improvement of more than 7 points of accuracy and 6 points of F1 compared to the baseline. The general knowledge priors, Google and *Phrasefinder*, obtained an improvement of approximately 2 points in each metric, although they are far from the improvement obtained by the Cookbook prior. This implies that some knowledge about actions in real-world is helpful; for instance, solely discarding non-existing actions can improve the results. At the right-most column, the perfect prior confirms that the maximum achievable performance is still far from the performance achieved using an action prior obtained from corpora. In fact, the perfect prior is the perfect example of how a prior that adapts to a person's life style and their ADLs can improve the classification of actions.

	Baseline		Cookbook prior		Google prior		*Phrasefinder* prior		Perfect prior	
Split	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
R	12.61% (±0.56)	16.52 (±0.23)	**18.08**% (±0.99)	**22.65** (±0.80)	14.89% (±0.93)	18.89 (±0.64)	14.34% (±0.90)	18.29 (±1.05)	51.46% (±1.45)	44.14 (±0.97)
NR	8.03% (±0.54)	11.46 (±1.21)	**11.47**% (±1.41)	9.73 (±1.25)	9.17% (±0.91)	**12.58** (±1.46)	10.37% (±1.01)	11.48 (±1.16)	54.31% (±0.42)	45.51 (±1.68)

Table 5.5: Table of zero-shot action classification results using the D_V and D_O detectors: comparison between the baseline and the experiments using the Cookbook, the Google, the *Phrasefinder*, and the perfect priors. For each row and column, the average result is presented on top and the standard deviation is shown below. The highest row-wise mean is highlighted in bold.

Meanwhile, for the NR split, the best F1 is obtained by the Google prior, with an slight improvement of 1 point in both accuracy and F1. This may confirm, as in the case of the R split, that adding external knowledge, even if it is not specific to the domain of the actions, can enhance the results. The highest accuracy is obtained by the Cookbook prior, having also a F1 lower than the baseline. This may suggest that specific domain has boosted the performance of recipe related classes while other classes have worsened the results. In the following part, the results per class are analysed to provide insights on this theory.

In the case of the action inference system A' created with the D'_V and D'_O detectors, Table 5.6 shows an improvement in the baseline results of 2-3 points in both the accuracy and the F1 compared to the detector A. However, there is still room for improvement by creating detectors that are adapted for the zero-shot setting rather than employing detectors built for general HAR. For the R split, there is an improvement with every prior but, specially, with the Google prior, obtaining the highest accuracy and macro-F1 metrics. Close to that is the result of the Cookbook prior. Later, analysing the results per classes, it will be obvious why the Google prior performed better than the Cookbook prior. Observing the results, one may think that, within a closed domain of knowledge (actions related to recipes in this case), the use of priors provides a boost in performance, i.e., it seems it is easier for a prior to fit the test set distribution. Nonetheless, a better fitting should still be beneficial

	Baseline		Cookbook prior		Google prior		*Phrasefinder* prior		Perfect prior	
Split	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
R	15.53% (±1.14)	19.52 (±2.68)	17.65% (±1.48)	21.88 (±1.39)	**18.01**% (±1.66)	**22.23** (±2.90)	15.78% (±1.53)	19.47 (±2.16)	50.09% (±2.02)	42.04 (±1.41)
NR	10.32% (±1.88)	**13.39** (±1.95)	**14.22**% (±1.88)	11.10 (±1.66)	10.52% (±1.63)	12.66 (±1.99)	11.90% (±1.79)	11.87 (±1.58)	47.22% (±3.71)	40.70 (±3.07)

Table 5.6: Table of zero-shot action classification results using the D'_V and D'_O detectors: comparison between the baseline and the experiments using the Cookbook, the Google, the *Phrasefinder*, and the perfect priors. For each row and column, the average result is presented on top and the standard deviation is shown below. The highest row-wise mean is highlighted in bold.

judging by the results of the perfect prior.

For the NR split, the best accuracy is obtained by the Cookbook prior once again, obtaining an improvement of 4 points. Nevertheless, the macro-F1 is not improved with any prior, the baseline's one remains the highest. The results per class will shed light on this in the following part. For a more open domain of actions, it is difficult to approximate a good prior, as the result highly depends on the test set distribution. In a realistic setting, it would be better to try to approximate a prior for such an open set of actions using previous measurements. That is, considering the daily frequency of actions, a better prior could be approximated that would highly resemble the perfect prior.

Results by class. The effect of each prior on the results per class of the action inference system A is shown in Tables 5.7 (for the R split) and 5.8 (for the NR split). The metrics in bold highlight the highest accuracy value per row (for each class). Only the accuracy is provided in this part as an average of three runs, with the standard deviation.

In the case of the R split in Table 5.7, the large improvement of the Cookbook prior can be seen in the large boost obtained by many classes, explaining the high accuracy and F1 values in the general results of Table 5.5. Both the Google and the *Phrasefinder* priors improve the results of three classes each, explaining the general results table, in which they got similar results.

In the NR split, Table 5.8 shows various classes with **0** accuracy in the

Class (R split)	Baseline	Cookbook prior	Google Prior	*Phrasefinder* prior	Perfect prior
cut bell pepper	14.22% (±6.04)	**23.28**% (±6.64)	14.46% (±6.36)	6.86% (±1.93)	63.97% (±9.92)
cut onion	0.57% (±0.00)	**8.05**% (±0.47)	1.92% (±0.27)	2.87% (±0.47)	57.47% (±2.15)
put bread	18.09% (±4.51)	25.89% (±7.39)	25.53%(±4.84)	**29.79**% (±6.26)	64.18% (±8.74)
put cup	7.92% (±3.58)	14.17% (±5.62)	14.58% (±5.62)	**15.42**% (±4.60)	41.67% (±10.27)
put lettuce	21.36% (±3.46)	**41.75**% (±3.46)	25.57% (±2.29)	37.86%(±1.59)	77.67% (±1.37)
put onion	2.56% (±3.63)	**10.26**% (±5.54)	2.56%(±2.09)	5.98%(±3.20)	5.13% (±2.09)
put plate	21.32% (±5.73)	29.41% (±4.33)	25.25% (±5.58)	**29.66**% (±5.71)	61.52% (±10.54)
put pot	14.52% (±3.99)	**28.05**% (±1.23)	16.50% (±2.60)	25.08% (±1.23)	50.17% (±1.23)
put tomato	4.37% (±1.48)	3.17% (±1.48)	**5.95**% (±0.97)	1.59% (±1.12)	13.49% (±0.56)
take bowl	30.00% (±7.08)	18.00% (±5.19)	**32.22**% (±7.31)	18.44% (±6.02)	75.11% (±6.19)
take egg	0.00% (±0.00)	0.98% (±1.39)	**2.94**% (±4.16)	0.98% (±1.39)	6.86% (±3.67)
take onion	6.11% (±1.57)	**17.22**% (±1.57)	8.33% (±2.72)	7.78% (±2.08)	39.44% (±2.08)
take pan	17.11% (±1.07)	**17.98**% (±3.28)	17.54% (±3.10)	8.33% (±4.34)	73.25% (±4.07)
take pot	2.99% (±1.60)	**7.26**% (±1.60)	2.99% (±2.63)	2.99% (±1.60)	16.24% (±1.21)
take tomato	2.63% (±2.15)	**3.07**% (±1.24)	2.63% (±1.07)	0.88% (±1.24)	17.54%(±1.64)
wash pot	10.85% (±1.10)	**13.95**% (±3.29)	9.30% (±1.90)	11.63% (±1.90)	57.36%(±3.95)

Table 5.7: Table of zero-shot action classification results by class with the action inference system A in the R split using the accuracy: comparison between the baseline and the experiments using the Cookbook, the Google, the *Phrasefinder*, and the perfect priors. For each action and each experiment, each result is presented with the mean of three runs and the standard deviation. The highest row-wise mean is highlighted in bold (not taking into account the perfect prior experiments).

baseline, confirming that the lower baseline results were produced by actions that were hurt by the performance of D_V and D_O. In addition, half of the classes do not have any performance gain in any experiment (those in which the baseline is highlighted in bold). There may be some reasons for this: (i) the presence of meta-objects (such as *eating* or *cooking utensil*), as discussed by Sudhakaran and Lanz (2018), can affect the performance, as a single label (hyperonym) covers various objects (hyponyms) and learning them is more difficult; and (ii) the difficulty of D_V and D_O to learn verbs and objects, which affects the performance of the inference of actions. Not even the action priors can improve the situation in those cases. The issue can be attributed to the amount of samples to train they had or their intrinsic variance (and, thus, difficulty to be learnt).

For the system A' the results are presented in Tables 5.9 (for the R split)

Class (NR split)	Baseline	Cookbook prior	Google Prior	*Phrasefinder* prior	Perfect prior
close drawer	7.41% (±5.24)	0.00% (±0.00)	**15.56**% (±6.54)	8.89% (±5.44)	53.33% (±10.10)
cut cucumber	6.41% (±2.27)	0.96% (±0.00)	**8.65**% (±3.07)	3.21% (±0.60)	84.78% (±9.50)
cut lettuce	**5.80**% (±3.69)	1.45% (±1.02)	3.62% (±1.02)	1.45% (±2.05)	38.41% (±9.11)
divide/pull apart lettuce	**0.42**% (±0.59)	0.00% (±0.00)	0.00% (±0.00)	0.42% (±0.59)	16.25% (±3.06)
divide/pull apart onion	**0.00**% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	22.44% (±7.08)
open fridge drawer	**0.00**% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	42.32% (±7.42)
put bell pepper	5.67% (±5.31)	0.00% (±0.00)	**17.02**% (±6.95)	0.71% (±1.00)	24.82% (±14.15)
put bowl	16.55% (±3.65)	**48.55**% (±4.66)	22.82% (±3.05)	31.77% (±3.12)	77.85% (±3.05)
put cheese container	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	9.40% (±7.93)
put cup	5.83% (±1.56)	8.33% (±0.59)	**10.42**% (±0.59)	**10.42**% (±0.59)	30.42% (±4.71)
put cutting board	28.67% (±3.40)	29.33% (±12.26)	**34.00**% (±5.89)	25.33% (±12.26)	63.33% (±8.22)
put plate	14.95% (±0.92)	**37.50**% (±6.24)	22.30% (±2.27)	26.47% (±2.40)	70.83% (±3.81)
take bell pepper	**11.95**% (±3.21)	0.00% (±0.00)	6.92% (±4.71)	0.00% (±0.00)	70.83% (±6.23)
take cheese container	**2.38**% (±2.23)	0.00% (±0.00)	0.60% (±0.84)	0.00% (±0.00)	48.81% (±3.04)
take sponge	**8.33**% (±3.90)	0.00% (±0.00)	4.17% (±1.95)	4.69% (±1.28)	56.25% (±9.20)
wash eating utensil	**4.97**% (±1.09)	2.34% (±0.41)	2.92% (±1.09)	1.75% (±0.72)	49.42% (±11.42)

Table 5.8: Table of zero-shot action classification results by class with the action inference system A in the NR split using the accuracy: comparison between the baseline and the experiments using the Cookbook, the Google, the *Phrasefinder*, and the perfect priors. For each action and each experiment, each result is presented with the mean of three runs and the standard deviation. The highest row-wise mean is highlighted in bold (not taking into account the perfect prior experiments).

Class (R split)	Baseline	Cookbook prior	Google Prior	*Phrasefinder* prior	Perfect prior
cut bell pepper	22.55% (±4.00)	**26.96**% (±5.11)	23.77% (±4.34)	16.67%(±4.00)	73.04%(±3.08)
cut onion	5.36% (±2.12)	**10.54**% (±2.75)	6.90% (±3.08)	9.77% (±4.53)	58.05% (±11.97)
put bread	14.89% (±2.61)	**26.24**% (±2.01)	20.57% (±1.33)	23.40% (±1.74)	57.45% (±6.95)
put cup	8.33% (±5.14)	8.33% (±3.12)	12.08% (±4.60)	**12.92**% (±4.60)	34.17% (±9.26)
put lettuce	15.21% (±0.46)	**31.07**% (±2.86)	23.62% (±0.46)	**31.07**% (±1.37)	79.29% (±6.65)
put onion	0.85% (±1.21)	**9.40**% (±5.27)	3.42% (±3.20)	4.27% (±1.21)	3.42% (±2.42)
put plate	17.40% (±7.16)	**28.92**% (±9.76)	22.55% (±9.72)	24.75% (±9.87)	51.23% (±17.00)
put pot	24.75% (±6.42)	31.35% (±6.78)	28.05% (±7.34)	**32.67**% (±7.05)	46.53% (±4.92)
put tomato	3.97% (±3.12)	2.78% (±3.93)	**6.75**% (±3.93)	4.37% (±4.59)	12.70% (±4.49)
take bowl	37.78% (±8.44)	12.44% (±4.16)	**38.00**% (±6.06)	21.11% (±6.89)	78.00% (±7.56)
take egg	**6.86**% (±3.67)	2.94% (±2.40)	4.90% (±1.39)	3.92% (±1.39)	12.75% (±5.00)
take onion	6.11% (±1.57)	**16.11**% (±0.79)	7.78% (±1.839)	7.22% (±2.08)	31.11% (±6.14)
take pan	21.49% (±7.15)	12.28% (±4.84)	**22.37**% (±5.58)	10.96% (±3.77)	74.12% (±2.24)
take pot	9.40% (±4.95)	**18.80**% (±3.96)	9.40% (±4.23)	5.56% (±3.68)	19.23% (±4.56)
take tomato	10.53% (±5.58)	**11.40**% (±5.08)	10.53% (±5.58)	4.39% (±3.45)	28.51% (±10.32)
wash pot	**10.08**% (±4.39)	2.33% (±0.00)	**10.08**% (±2.19)	3.88% (±2.19)	19.38% (±6.10)

Table 5.9: Table of zero-shot action classification results by class with the action inference system A' in the R split using the accuracy: comparison between the baseline and the experiments using the Cookbook, the Google, the *Phrasefinder*, and the perfect priors. For each action and each experiment, each result is presented with the mean of three runs and the standard deviation. The highest row-wise mean is highlighted in bold (not taking into account the perfect prior experiments).

and 5.10 (for the NR split). It was noticeable that the Cookbook prior did not obtain the best result in the R split with A', being second to the Google prior by little. Nonetheless, as observed in Table 5.9, many classes did improve using the Cookbook prior, more classes than with the Google or *Phrasefinder* priors. In fact, the Google prior's improvement was not so significant in the classes in which it obtained the best result. That is, due to the distribution of the dataset, by obtaining better results in classes with many samples, the Google prior obtained a higher result. This means that, by adjusting better to the distribution of the data, it is possible to obtain even better results.

For the NR split, once again, there is a problem with some classes starting with zero accuracy and many of them not improving with any prior. Nevertheless, the perfect prior's results are still quite high, meaning that, using the appropriate prior, there is still room for improvement of the results.

Class (NR split)	Baseline	Cookbook prior	Google Prior	*Phrasefinder* prior	Perfect prior
close drawer	3.70% (±2.77)	0.00% (±0.00)	3.70% (±2.77)	**6.67**% (±3.63)	30.373% (±5.83)
cut cucumber	6.73% (±1.57)	4.97% (±0.23)	4.01% (±1.20)	**8.49**% (±1.98)	60.42% (±10.16)
cut lettuce	**12.32**% (±7.39)	9.42% (±6.72)	10.14% (±7.17)	6.52% (±4.70)	48.55% (±16.49)
divide/pull apart lettuce	**0.00**% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	24.58% (±3.12)
divide/pull apart onion	**0.00**% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	59.62% (±11.85)
open fridge drawer	**0.37**% (±0.53)	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	35.96% (±4.20)
put bell pepper	14.18% (±4.37)	2.13% (±1.74)	**19.15**% (±3.47)	7.80% (±2.65)	18.44% (±2.01)
put bowl	25.50% (±4.28)	**61.07**% (±4.87)	31.77% (±3.25)	39.60% (±4.35)	82.33% (±3.57)
put cheese container	0.85% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	0.00% (±0.00)	5.98% (±6.73)
put cup	7.08% (±4.12)	7.08% (±1.56)	**9.58**% (±2.57)	**9.58**% (±2.12)	20.83% (±2.57)
put cutting board	26.00% (±9.93)	24.00% (±6.53)	24.67% (±8.99)	21.33% (±8.99)	55.33% (±7.36)
put plate	23.77% (±9.01)	**42.16**% (±8.99)	27.94% (±7.66)	34.80% (±9.94)	65.69% (±9.61)
take bell pepper	**8.81**% (±3.21)	0.00% (±0.00)	4.40% (±0.89)	0.00% (±0.00)	31.45% (±4.71)
take cheese container	**1.19**% (±1.68)	0.00% (±0.00)	0.60% (±0.84)	0.00% (±0.00)	39.29% (±6.36)
take sponge	**5.73**% (±0.74)	0.00% (±0.00)	1.56% (±0.00)	2.08% (±0.74)	46.35% (±6.03)
wash eating utensil	**7.31**% (±3.31)	3.80% (±2.19)	4.39% (±3.28)	2.05% (±2.30)	33.04% (±9.54)

Table 5.10: Table of zero-shot action classification results by class with the action inference system A' in the NR split using the accuracy: comparison between the baseline and the experiments using the Cookbook, the Google, the *Phrasefinder*, and the perfect priors. For each action and each experiment, each result is presented with the mean of three runs and the standard deviation. The highest row-wise mean is highlighted in bold (not taking into account the perfect prior experiments).

Conclusion. From this experiments, we can conclude that specific domain knowledge can be beneficial, as in the case of the Cookbook prior in the R split. In fact, not only is it helpful that the domain of the prior and the one of the test classes matches, approximating the prior's distribution to the test set distribution is also a very promising approach, given the results obtained by the perfect prior. For this dataset, the actors had controlled actions, but people have different ways to approach ADLs and different action distributions. An approximation to the perfect prior would imply adjusting a prior for the lifestyle of a person who is going to use a system to infer their actions. In the opposite side, we have the Google prior, whose generic knowledge seems to be more balanced and helpful in almost all the classes but not as beneficial as a prior specific to a class distribution.

5.4 Summary and conclusions

This chapter has focused on the use of external knowledge to improve the results of an egocentric action inference system. The proposed system follows the popular two-stream architecture using the network proposed in Section 4.5. On top of it, a prior extracted from text corpora is applied to introduce a distribution that closely resembles that of the real-world. Experiments on two different splits, one focused on recipes and another one with a more open space of classes, with two action inference system variations, have been carried out. The results show that using an appropriate prior distribution, better adjusted to the distribution of actions, may be a promising way to improve the ZS-EAR.

Once you have traveled, the voyage never ends.

Pat Conroy

CHAPTER

6

Conclusions and Future Work

This chapter aims at providing an overview of the work done and the contributions this dissertation has generated. Moreover, the hypothesis and the objectives set in Chapter 1 are reviewed and analysed in order to reason to what extent they have been achieved. This is accompanied with the publications that have been generated as a result of the research, in order to disseminate its findings. Finally, the future lines proposed to continue the work of this dissertation is shown, as well as some final remarks.

The chapter is divided into six sections: Section 6.1 summarises the work done in this dissertation, Section 6.2 lists the contributions from this work, Section 6.3 validates the hypothesis and the objectives that were set, Section 6.4 presents the publications generated from this dissertation, while Section 6.5 discusses the future work and Section 6.2 draws the final remarks.

6.1 Summary of work and conclusions

The HAR field is an extensive domain in which multiple solutions have been proposed with the aim of understanding the content of videos. Although the two main branches of this field, the exocentric- and egocentric-vision based HAR, may seem very different from each other, the reality is that both of them are tightly related. In the very basis, both of them analyse the dynamics and the visual appearance of videos to model the uncertainty and inspiration taken from one of them can help the other advance. This dissertation had set as an starting point the use of egocentric vision as a mean for avoiding the problems that the third-vision has for monitoring actions in closed spaces (Nguyen et al., 2016). The aim of the dissertation was to propose a system that would be able to infer actions related to ADLs with the objective of monitoring elder people's activities and provide them with support through the analysis of their actions. For that, the egocentric vision approach was considered.

Following the EAR literature, the standard approach to learn representations from videos is based on a branched learning of motion features and appearance or object features separately. This type of approach has taken inspiration from the exocentric-vision based two-stream solutions, in which two separate streams learn motion and appearance features. Therefore, as our prime goal was to design a similar structure to learn action patterns from videos, we split this goal into two sub-objectives: creating the motion and the active object detectors or streams. However, the range of actions within ADLs is vast and the learning of each individual action class requires a large pool of samples. In contrast, the datasets available for scientific purposes in the literature are limited and do not provide enough samples or they follow long-tail distributions (see Section 2.1.5).

With the objective of creating an action prediction system that scales to many actions and can be used in realistic settings, the dissertation proposes a zero-shot approach that leverages the aforementioned branched architecture to separately learn the motion and objects associated to actions. The motion is known as "the verb", referring to the way of labeling different types of motion. The object that should be learnt and recognised is the active object,

the one that is being manipulated and that is relevant for the action. Having the verb and the object branches properly trained, both predictions can be combined to obtain an action prediction. This way, actions never seen during the training phase can be inferred. With the classical approach, annotations for each possible combination of verb and object would have been required for the learning phase.

Following this setting, the dissertation first proposes to create a motion detector prototype, presented in Chapter 3. The objective of the motion detector is to extract temporal patterns from videos. OF images are used for that purpose, as they are well recognised in the literature for being suitable features to represent motion in videos. A CNN architecture is proposed to extract patterns from OF images, as DL architectures are well-known for obtaining state-of-the-art results in video-related problems. Thus, this chapter settles the basis of the motion detector and aims at proving the generalisation ability of the proposed system using a fall classification task.

For the learning of active objects, various proposals are presented in Chapter 4. First of all, experiments with a SpatialNet are presented, a baseline single-stream architecture used for two-stream approaches in the literature. Using RGB features seems to provide a good baseline result, but the temporal modeling had to be improved. That is, it is not enough to just exploit single timesteps' local features for the final prediction, global temporal features need to be learnt. Next, the Faster R-CNN for object detection was presented. Thanks to this architecture, BB of objects can be inferred from images. Exploiting this, two approaches are proposed: the first one leverages the feature vectors obtained from each object proposal and the second one employs the recognition of hands to create an egocentric attention mechanism. In general, there is more difficulty than expected in recognising active objects, as demonstrated by the experiments carried out throughout the chapter. The Faster R-CNN can predict several objects but distinguishing the most relevant one requires understanding the dominant motion of the action through time. Taking into account all of this, we believe more research on active object classification is required.

Furthermore, the chapter also provided action classification results exploiting the systems used for active object classification. There seems to be a need for more information to classify actions (such as motion or sound) or even to exploit better the detected objects to add more information. Attention mechanisms have also been widely proposed through the literature to tackle both active object and action classification. And it seems that those systems that have a multi-objective during the learning phase end up being more robust and generalise better, obtaining state-of-the-art results.

Eventually, as previously mentioned, the proposed action classifier has two branches: one in charge of learning motion patterns and the other one of learning active objects. As this dissertation aimed at creating a system capable of being deployed in a realistic scenario, we deemed the convenience of adding another layer on top. That is, as it is, the combination of the verb and the object predicted by the subsystems composes the action, but this prediction is only done using the information extracted from the input video. Or better said, the probability distribution of the output only takes into account the input and not the meaning of the action, its frequency, and so forth. Another layer on top could be added to address the issue of actions that do not exist being predicted, rare or not common actions having high probability, and so on. Because, apart from the confidence of the classifier in the prediction based on the input, there should be information about the frequency of actions in the real-world. This dissertation contributes a weighting of the probability distribution that is obtained from action classifiers by using an action prior extracted from text corpora. A corpus has an intrinsic action probability distribution that should be close to a real-world distribution, i.e., not common actions may appear few times while common ones should dominate.

In this sense, Chapter 5 presented the final action classifier using the zero-shot setting and the weighting of the action probability distribution using text corpora. In fact, the results obtained from this chapter aimed at validating the hypothesis presented at the beginning in Chapter 1. Several text sources are presented in the analysis, having each of them their own distribution, and two types of splits of the dataset employed are proposed to analyse the effect of the different action priors.

In conclusion, the final system is created using various building blocks: the motion detector analysed in Chapter 3, the knowledge acquired about active object detection from Chapter 4, and the external knowledge source added as a layer on top in Chapter 5.

6.2 Contributions

This dissertation led to the publication of various scientific and technical contributions of the work.

6.2.1 Scientific contributions

The following scientific contributions are generated from this dissertation:

- The design of **an end-to-end approach for classifying falls in videos** in Chapter 3. A DL approach is employed for that. The network is fed with optical flow stacks and outputs a probability distribution which, in this case, is binary, but it could be generalised to multiple classes. This contribution was aimed at solving part of objectives 2 and 3, the design and implementation of a motion detector. The contribution done in this work is a system capable of obtaining state-of-the-art results in three fall classification datasets; namely, the URFD, the FDD, and the Multicam datasets. In fact, these results were obtained without any ad-hoc parameter tuning for each dataset and the generalisation capability of the system was proved with two types of experiments. In the first set of experiments, different lighting conditions were employed: a darkening to simulate night conditions and the addition of a dynamic lighting for scenarios with a lighting that abruptly changes. In the second experiment, the three datasets were combined for the learning part and the evaluation was done using the combination of the datasets and also each individual dataset. This way, we refuted the possibility of the system learning specific features from each dataset, as the variance of falls increases when the datasets are combined.

- **An approach to use external knowledge to weight the action probability distribution in a zero-shot setting for egocentric videos**. This is presented in Chapter 5. The inference of actions using the combination of verb and object has already been explored in the literature, yet the combination of verb and object probability distributions was only exploited, to the best of our knowledge, in exocentric-vision based approaches. Our contribution is the addition of another layer to that approach, in which the output is weighted with a prior coming from text corpora (the external knowledge). This allows to introduce an action probability distribution that is closer to reality, i.e, a distribution sampled from text corpora, sources that contain intrinsic knowledge about the world. Various text sources are considered for the experiments and two different dataset splits are employed to assess the effect of the chosen text source: one specialised in recipe-related actions and another one without constraints. This contribution aims to validate the hypothesis of the dissertation and also to fulfill the rest of the initially stated objectives.

6.2.2 Technical contributions

The source code used for this dissertation is released in Github: for the "Vision-based fall detection with convolutional neural networks" publication[1] and for the "Using External Knowledge to Improve Zero-shot Action Recognition in Egocentric Videos" publication[2]. Furthermore, the object bounding box annotations for the GTEA Gaze+ explained in Chapter 4.3 were published in Zenodo (Núñez-Marcos et al., 2020).

[1] https://github.com/AdrianNunez/Fall-Detection-with-CNNs-and-Optical-Flow

[2] https://github.com/AdrianNunez/zeroshot-action-recognition-action-priors

6.3 Hypothesis and objective validation

In Section 1.2, in the beginning of the dissertation, the following hypothesis was formulated:

Hypothesis. Using external knowledge extracted from text corpora, it is possible to improve the standard classification metrics of the zero-shot egocentric action recognition.

In order to validate it, a goal was proposed:

Goal. To design and implement a zero-shot egocentric action recognition system that leverages knowledge from text corpora for the improvement of the standard classification metrics.

To achieve this goal, more specific sub-goals were defined. In what follows, a discussion on how each sub-goal has been tackled in this dissertation is offered:

1. *To study the current state of the art in EAR and its zero-shot counterpart.* Chapter 2 explored the literature on EAR, giving an extensive review of several works. It also described the datasets found in the state of the art and the relevant works with the zero-shot setting. Moreover, for the fall classification task, a brief review on the literature was shown in Section 3.1.

2. *To design a suitable DL architecture for the zero-shot problem.* This sub-objective has been tackled throughout the dissertation, trying to find suitable subsystems for the learning of verb and object networks in Chapters 3 and 4. Finally, in Section 5.1.1, the DL architecture was defined.

3. *To implement the supervised classification models that take videos as input and are able to output probability distributions of motion and object labels.* Although the system used for verbs and objects is defined in Section 5.1.1, the architecture itself was first described in Section

4.5.2. The design of this network has been highly influenced by the conclusions extracted from Chapter 3 and also from the first sections of Chapter 4.

4. *To identify the evaluation methodology for the ZS-EAR which better addresses the requirements of the system.* For the evaluation, we identified that the type of split of the dataset was crucial in Section 5.2.1, in which we described how we have divided the dataset for training and for evaluation. In addition, the metrics used for our dataset were specified in Section 4.1.

5. *To identify and extract the appropriate knowledge sources that can be employed: within the domain of the dataset and those of general knowledge.* In Section 5.1.2, three different knowledge sources were identified, one with specialised knowledge and two of them with general knowledge, one of them sampled from restricted queries and another one extracted from the wild.

6. *To quantitatively validate the results obtained with and without the use of external knowledge sources, and with general and domain specific knowledge sources.* The results of the experiments for action inference are shown in Section 5.3.2. The three different knowledge sources are employed, and also the baseline without any action prior.

Taken this into account, it can be claimed that all the sub-objectives have been accomplished. As it can be observed, Chapters 3 and Chapter 4 were pivotal for providing hints about the choice of the architecture for the system of Chapter 5, which was intended to validate the hypothesis. In fact, many of the sub-objectives were achieved in this last chapter.

Finally, through the accomplishment of these sub-goals and the main goal, the hypothesis set at the beginning has been validated. The results from the experiments of Chapter 5 prove that the use of external knowledge from text corpora can be employed to improve the ZS-EAR. As it is shown in the experiments, the standard classification metrics (accuracy and macro-F1) increase with the addition of priors extracted from text corpora. This is due

to the weighting of the probability of actions given by the external knowledge, i.e., the probability distribution of actions in the real-world. Nonetheless, this is validated under the constraints of Section 1.2.

6.4 Publications

This dissertation led to the publication of various scientific papers to disseminate the contributions of the work. Starting from the publication of the motion detector prototype, evaluated using the fall classification task:

- Nunez-Marcos, A., Azkune, G., & Arganda-Carreras, I. (2017). "Vision-based fall detection with convolutional neural networks". In the Wireless communications and mobile computing journal, volume 2017. DOI: 10.1155/2017/9474806. JCR Impact Factor (2018): 0.869, Q4. Published in December 2017.

The contribution of an action prior sampled from text corpora for EAR in a zero-shot setting is presented in the following publication:

- Núñez-Marcos, A., Azkune, G., Agirre, E., López-de-Ipiña, D., & Arganda-Carreras, I. (2020). "Using External Knowledge to Improve Zero-Shot Action Recognition in Egocentric Videos". In the International Conference on Image Analysis and Recognition (pp. 174-185). Springer, Cham. Published in June 2020.

Furthermore, a parallel work focused in the Smart Cities and the potential application of this work in the field was published:

- Ruben Sánchez Corcuera, Adrian Núñez-Marcos, Jesus Sesma-Solance, Aritz Bilbao Jayo, Rubén Mulero, Unai Zulaika Zurimendi, Gorka Azkune, & Aitor Almeida (2019). "Smart cities survey: Technologies, application domains and challenges for the cities of the future". In International Journal of Distributed Sensor Networks. vol. 15. p. 1550147719853984. DOI: 10.1177/1550147719853984. JCR Impact Factor (2018): 1.614 , Q3. Published in June 2019.

6.5 Future work

Inspired by the observations and limitations found throughout this dissertation, the following research lines have been identified.

6.5.1 Improving the verb and object detectors

The baseline detectors used in Chapter 5 were not specifically designed for a zero-shot approach, as it is out of the scope of the contribution to have the best possible detectors. Nonetheless, there is ongoing research that may point out in the direction that should be taken to improve them. The idea is not to use bigger neural networks, but rather to go in other directions. We mention the ones we believe are the most relevant ones:

- One of the popular research lines is the use of graphs (Wray et al., 2016) (Wang and Gupta, 2018) (Herzig et al., 2019) (Chen et al., 2019) (Materzynska et al., 2019) as a way to represent high-level relations between verbs, objects, and actions, creating connections between nodes based on visual appearance or motion. This has become more popular since the publication of GCNs (Kipf and Welling, 2016). Nodes can be represented, for instance, using the object proposals from object detection networks. In addition, clustering features (or descriptors) and creating Bag of Words (BoW) like descriptors to represent edges or nodes of graphs may help to generalise better to unseen samples. An example of this approach would be the approach of Kitani et al. (2011) for the motion descriptors. The idea is that features can be clustered to get a more general representation. Then, the descriptor of a sample can be represented by a BoW-vector that indicates the presence or absence of clustered feature representations (also called visual words).

- The previous research line can be seen as forcing the system to use soft features that do not create tight representations. Similar to that, Wray et al. (2018) employed a soft assignment to various labels for each sample instead of having to tag a sample with one specific label

(hard assignment). If there are proper annotations to do this, this could be a promising extension of the work developed in Chapter 5. There would be an issue with respect to the prediction of actions, as some combinations of verbs and objects could potentially be pointing towards the same action but with different names, as in ”pick cucumber” and ”take cucumber”.

- The use of object detectors such as the Faster R-CNN explored in Section 4.3 can potentially aid at discovering objects with as little context of the image as possible, as that would introduce undesired noise for the ZSL. An approach exploiting this is the exocentric work of Shen et al. (2018), in which they use object detections for both the verb and the object branches. But not only the region of an input image can be exploited, feature vectors of object detections as in the work of Ma et al. (2018) can also be used to discover relations between objects. If done correctly and the interactions are correctly predicted, it would be an interesting approach for the zero-shot prediction of actions. Graph-based approachs such as the one of Materzynska et al. (2019) also exploited object detections, as this allowed for the recognition of elements (subject and objects) that are used to build the graph.

- Finally, multi-task learning has shown to be very promising, obtaining some of the best results for the GTEA Gaze+ and for the EGTEA Gaze+ datasets, although the latter was just demonstrated in the first split. Learning all the components of actions (verbs and objects) alongside actions was first done in the work of Ma et al. (2016) for GTEA Gaze+ and setting more sub-objectives to learn egocentric cues such as gaze attention maps and hand location coordinates was later done by Kapidis et al. (2019a) for EGTEA Gaze+. Combined with the previous ideas, the multi-tasking learning could be the key to create better zero-shot detectors or action inference systems.

6.5.2 Verb and object independence assumption

Throughout the dissertation, we have assumed the independence between the verb and the object in the branched learning paradigm. This means that we assumed that $p(v, o) = p(v) \times p(o)$, where v is the verb and o is the object, having the conditional probabilities equal to zero. We have not explored this research line, i.e., modeling $p(v|o)$ and $p(o|v)$ (the conditional probabilities) and incorporating them into the system, as it posed new research questions out of the scope of Chapter 5. This may be a future line of research that can be followed to extend the work of the previously mentioned chapter.

6.6 Final remarks

This dissertation was aimed at making significant contributions to the research field of the HAR, but, specially, to contribute to the egocentric community. We firmly believe the EAR field will have an strong impact in the future of the AAL and, therefore, we hope the contributions and research lines stated in this dissertation will help future researchers to further push the state of the art.

Bibliography

Abebe, G., Cavallaro, A., and Parra, X. (2016). Robust multi-dimensional motion features for first-person vision activity recognition. *Computer Vision and Image Understanding*, 149:229–248. 37

Aboubakr, N., Crowley, J. L., and Ronfard, R. (2019). Recognizing manipulation actions from state-transformations. *arXiv preprint arXiv:1906.05147*. 15, 18

Akl, A., Snoek, J., and Mihailidis, A. (2015). Unobtrusive detection of mild cognitive impairment in older adults through home monitoring. *IEEE Journal of Biomedical and Health Informatics*, 21(2):339–348. 3

Al-Naser, M., Ohashi, H., Ahmed, S., Nakamura, K., Akiyama, T., Sato, T., Nguyen, P. X., and Dengel, A. (2018). Hierarchical model for zero-shot activity recognition using wearable sensors. In *ICAART (2)*, pages 478–485. 41

Ambrose, A. F., Paul, G., and Hausdorff, J. M. (2013). Risk factors for falls among older adults: a review of the literature. *Maturitas*, 75(1):51–61. 45

Arabacı, M. A., Özkan, F., Surer, E., Jančovič, P., and Temizel, A. (2018). Multi-modal egocentric activity recognition using audio-visual features. *arXiv preprint arXiv:1807.00612*. 2, 15, 37

Arandjelović, R. and Zisserman, A. (2012). Three things everyone should know to improve object retrieval. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2911–2918. IEEE. 34

Asadi-Aghbolaghi, M., Clapés, A., Bellantonio, M., Escalante, H. J., Ponce-López, V., Baró, X., Guyon, I., Kasaei, S., and Escalera, S. (2017a). Deep learning for action and gesture recognition in image sequences: A survey. In *Gesture Recognition*, pages 539–578. Springer. 12

Asadi-Aghbolaghi, M., Clapes, A., Bellantonio, M., Escalante, H. J., Ponce-López, V., Baró, X., Guyon, I., Kasaei, S., and Escalera, S. (2017b). A survey on deep learning based approaches for action and gesture recognition in image sequences. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 476–483. IEEE. 4

Asnaoui, K. E., Hamid, A., Brahim, A., and Mohammed, O. (2017). A survey of activity recognition in egocentric lifelogging datasets. In *2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*, pages 1–8. IEEE. 12

Auvinet, E., Multon, F., Saint-Arnaud, A., Rousseau, J., and Meunier, J. (2010). Fall detection with multiple cameras: An occlusion-resistant method based on 3-d silhouette vertical distribution. *IEEE Transactions on Information Technology in B iomedicine*, 15(2):290–300. 48, 64, 65

Bambach, S. (2015). A survey on recent advances of computer vision algorithms for egocentric video. *arXiv preprint arXiv:1501.02825*. 12

Bambach, S., Lee, S., Crandall, D. J., and Yu, C. (2015). Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1949–1957. 15, 20

Behera, A., Chapman, M., Cohn, A. G., and Hogg, D. C. (2014). Egocentric activity recognition using histograms of oriented pairwise relations. In *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, volume 2, pages 22–30. IEEE. 15, 19

Behera, A., Hogg, D. C., and Cohn, A. G. (2012). Egocentric activity monitoring and recovery. In *Asian Conference on Computer Vision*, pages 519–532. Springer. 15, 19

Betancourt, A., Morerio, P., Regazzoni, C. S., and Rauterberg, M. (2015). The evolution of first person vision methods: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(5):744–760. 12

Bosch, A., Zisserman, A., and Munoz, X. (2007). Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, pages 401–408. 35

Bulling, A., Ward, J. A., Gellersen, H., and Troster, G. (2010). Eye movement analysis for activity recognition using electrooculography. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):741–753. 22

Bux, A., Angelov, P., and Habib, Z. (2017). Vision based human activity recognition: a review. In *Advances in Computational Intelligence Systems*, pages 341–371. Springer. 3

Cai, M., Lu, F., and Gao, Y. (2018). Desktop action recognition from first-person point-of-view. *IEEE Transactions on Cybernetics*, 49(5):1616–1628. 15, 36

Cao, Z., Simon, T., Wei, S.-E., and Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299. 42

Carreira, J. and Zisserman, A. (2017). Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308. 108

Cartas, A., Luque, J., Radeva, P., Segura, C., and Dimiccoli, M. (2019a). How much does audio matter to recognize egocentric object interactions? *arXiv preprint arXiv:1906.00634*. 2, 15, 37

Cartas, A., Luque, J., Radeva, P., Segura, C., and Dimiccoli, M. (2019b). Seeing and hearing egocentric actions: How much can we learn? In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0. v, 15, 37, 38

Cartas, A., Radeva, P., and Dimiccoli, M. Contextually driven first-person action recognition from videos. 15, 19

Chaaraoui, A. A., Climent-Pérez, P., and Flórez-Revuelta, F. (2012). A review on vision techniques applied to human behaviour analysis for ambient-assisted living. *Expert Systems with Applications*, 39(12):10873–10888. v, 12

Charfi, I., Miteran, J., Dubois, J., Atri, M., and Tourki, R. (2012). Definition and performance evaluation of a robust svm based fall detection solution. In *2012 Eighth International Conference on Signal Image Technology and Internet Based Systems*, pages 218–224. IEEE. 47, 64, 65

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*. vi, 80, 81

Chen, L., Hoey, J., Nugent, C. D., Cook, D. J., and Yu, Z. (2012). Sensor-based activity recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):790–808. 3

Chen, Y., Rohrbach, M., Yan, Z., Shuicheng, Y., Feng, J., and Kalantidis, Y. (2019). Graph-based global reasoning networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 433–442. 152

Damen, D., Doughty, H., Farinella, G. M., , Furnari, A., Ma, J., Kazakos, E., Moltisanti, D., Munro, J., Perrett, T., Price, W., and Wray, M. (2020). Rescaling egocentric vision. *CoRR*, abs/2006.13256. 40

Damen, D., Doughty, H., Farinella, G. M., Fidler, S., Furnari, A., Kazakos, E., Moltisanti, D., Munro, J., Perrett, T., Price, W., and Wray, M. (2018a). Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*. 12

Damen, D., Doughty, H., Maria Farinella, G., Fidler, S., Furnari, A., Kazakos, E., Moltisanti, D., Munro, J., Perrett, T., Price, W., et al. (2018b). Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736. ix, 37, 40

Damen, D., Leelasawassuk, T., Haines, O., Calway, A., and Mayol-Cuevas, W. W. (2014). You-do, i-learn: Discovering task relevant objects and their modes of interaction from multi-user egocentric video. In *BMVC*, volume 2, page 3. 17, 40

Davis, S. and Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366. 37

De la Torre, F., Hodgins, J., Bargteil, A., Martin, X., Macey, J., Collado, A., and Beltran, P. (2009). Guide to the carnegie mellon university multimodal activity (cmu-mmac) database. 40

Del Molino, A. G., Tan, C., Lim, J.-H., and Tan, A.-H. (2016). Summarization of egocentric videos: A comprehensive survey. *IEEE Transactions on Human-Machine Systems*, 47(1):65–76. 12

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee. 54

Devijver, P. A. (1984). *Connected components in binary images: the detection problem*. John Wiley & Sons, Inc. 47

Dezert, J. and Smarandache, F. (2004). Advances and applications of dsmt for information fusion. *Am. Res. Press, Rehoboth*, 1. 34

Diete, A., Sztyler, T., Weiland, L., and Stuckenschmidt, H. (2018). Improving motion-based activity recognition with ego-centric vision. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 488–491. IEEE. 15, 34

Dimiccoli, M., Marín, J., and Thomaz, E. (2018). Mitigating bystander privacy concerns in egocentric activity recognition with deep learning and intentional image degradation. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(4):1–18. 15, 39

Diraco, G., Leone, A., and Siciliano, P. (2010). An active vision system for fall detection and posture recognition in elderly healthcare. In *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*, pages 1536–1541. IEEE. 48

Fang, C. and Torresani, L. (2012). Measuring image distances via embedding in a semantic manifold. In *European Conference on Computer Vision*, pages 402–415. Springer. 35

Fathi, A., Farhadi, A., and Rehg, J. M. (2011a). Understanding egocentric activities. In *2011 International Conference on Computer Vision*, pages 407–414. IEEE. 15, 16, 17, 21

Fathi, A., Li, Y., and Rehg, J. M. (2012). Learning to recognize daily actions using gaze. In *European Conference on Computer Vision*, pages 314–327. Springer. 17, 40, 74

Fathi, A. and Rehg, J. M. (2013). Modeling actions through state changes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2586. 15, 18

Fathi, A., Ren, X., and Rehg, J. M. (2011b). Learning to recognize objects in egocentric activities. In *CVPR 2011*, pages 3281–3288. IEEE. 74

Fire, A. and Zhu, S.-C. (2015). Learning perceptual causality from video. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(2):1–22. 18

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395. 24

Furnari, A. and Farinella, G. M. (2019). What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6252–6261. 15, 29

Garcia-Hernando, G., Yuan, S., Baek, S., and Kim, T.-K. (2018). First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 409–419. 15, 36

Gasparrini, S., Cippitelli, E., Spinsante, S., and Gambi, E. (2014). A depth-based fall detection system using a kinect® sensor. *Sensors*, 14(2):2756–2775. 48

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448. vii, 89, 90

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2015). Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):142–158. vii, 89, 90

Gkioxari, G., Girshick, R., and Malik, J. (2015). Contextual action recognition with r* cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1080–1088. 15, 19

Gollwitzer, P. M. (1990). Action phases and mind-sets. *Handbook of motivation and cognition: Foundations of social behavior*, 2:53–92. 38

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680. 43

Goonawardene, N., Tan, H.-P., and Tan, L. B. (2018). Unobtrusive detection of frailty in older adults. In *International Conference on Human Aspects of IT for the Aged Population*, pages 290–302. Springer. 3

Guadarrama, S., Krishnamoorthy, N., Malkarnenkar, G., Venugopalan, S., Mooney, R., Darrell, T., and Saenko, K. (2013). Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2712–2719. 42

Guo, K., Ishwar, P., and Konrad, J. (2013). Action recognition from video using feature covariance matrices. *IEEE Transactions on Image Processing*, 22(6):2479–2494. 37

Harrou, F., Zerrouki, N., Sun, Y., and Houacine, A. (2016a). A simple strategy for fall events detection. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, pages 332–336. IEEE. 48, 63, 65

Harrou, F., Zerrouki, N., Sun, Y., and Houacine, A. (2016b). Statistical control chart and neural network classification for improving human fall detection. In *2016 8th International Conference on Modelling, Identification and Control (ICMIC)*, pages 1060–1064. IEEE. 46

Hayhoe, M. (2000). Vision using routines: A functional account of vision. *Visual Cognition*, 7(1-3):43–64. 17, 22

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778. 114, 124

Herzig, R., Levi, E., Xu, H., Gao, H., Brosh, E., Wang, X., Globerson, A., and Darrell, T. (2019). Spatio-temporal action graph networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0. 152

Huang, Y., Li, Z., Cai, M., and Sato, Y. (2019). Mutual context network for jointly estimating egocentric gaze and actions. *arXiv preprint arXiv:1901.01874*. 15, 30

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*. 56

Jang, Y., Sullivan, B., Ludwig, C., Gilchrist, I., Damen, D., and Mayol-Cuevas, W. (2019). Epic-tent: An egocentric video dataset for camping tent assembly. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0. 40

Javidani, A. and Mahmoudi-Aznaveh, A. (2018). A unified method for first and third person action recognition. In *Iranian Conference on Electrical Engineering (ICEE)*, pages 1629–1633. IEEE. 15, 33

Jegou, H., Perronnin, F., Douze, M., Sánchez, J., Perez, P., and Schmid, C. (2011). Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716. 34

Jiang, H., Song, Y., He, J., and Shu, X. (2020). Cross fusion for egocentric interactive action recognition. In *International Conference on Multimedia Modeling*, pages 714–726. Springer. 15, 31

Kanade, T. and Hebert, M. (2012). First-person vision. *Proceedings of the IEEE*, 100(8):2442–2453. 12

Kang, H., Hebert, M., and Kanade, T. (2011). Discovering object instances from scenes of daily living. In *2011 International Conference on Computer Vision*, pages 762–769. IEEE. 17

Kapidis, G., Poppe, R., van Dam, E., Noldus, L., and Veltkamp, R. (2019a). Multitask learning to improve egocentric action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0. 15, 31, 118, 119, 120, 153

Kapidis, G., Poppe, R., van Dam, E., Noldus, L. P., and Veltkamp, R. C. (2019b). Egocentric hand track and object-based human action recognition. *arXiv preprint arXiv:1905.00742*. 15, 34

Kapidis, G., Poppe, R., van Dam, E., Noldus, L. P., and Veltkamp, R. C. (2020). Object detection-based location and activity classification from egocentric videos: A systematic analysis. In *Smart Assisted Living*, pages 119–145. Springer. 15, 18

Kato, K., Li, Y., and Gupta, A. (2018). Compositional learning for human object interaction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 234–251. 42

Kazakos, E., Nagrani, A., Zisserman, A., and Damen, D. (2019). Epic-fusion: Audio-visual temporal binding for egocentric action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5492–5501. 15, 37

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*. vii, 101, 102, 103, 104

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 57

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*. 42, 152

Kitani, K. M., Okabe, T., Sato, Y., and Sugimoto, A. (2011). Fast unsupervised ego-action learning for first-person sports videos. In *CVPR 2011*, pages 3241–3248. IEEE. 15, 23, 152

Kumar, K. S. (2017). Activity recognition in egocentric video using svm, knn and combined svmknn classifiers. In *IOP Conference Series: Materials Science and Engineering*, volume 225, page 012226. IOP Publishing. 15, 24

Kwolek, B. and Kepski, M. (2014). Human fall detection on embedded platform using depth maps and wireless accelerometer. *Computer Methods and Programs in Biomedicine*, 117(3):489–501. 46

Kwon, H., Kim, Y., Lee, J. S., and Cho, M. (2018). First person action recognition via two-stream convnet with long-term fusion pooling. *Pattern Recognition Letters*, 112:161–167. 15, 26

Land, M., Mennie, N., and Rusted, J. (1999). The roles of vision and eye movements in the control of activities of daily living. *Perception*, 28(11):1311–1328. 17

Land, M. and Tatler, B. (2009). *Looking and acting: vision and eye movements in natural behaviour*. Oxford University Press. 22

Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE. 37

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444. 53, 54

Lee, T. and Mihailidis, A. (2005). An intelligent emergency response system: preliminary development and testing of automated fall detection. *Journal of Telemedicine and Telecare*, 11(4):194–198. 47

Li, J., Liu, X., Zhang, W., Zhang, M., Song, J., and Sebe, N. (2020). Spatio-temporal attention networks for action recognition and detection. *IEEE Transactions on Multimedia*. 116

Li, Y., Liu, M., and Rehg, J. M. (2018). In the eye of beholder: Joint learning of gaze and actions in first person video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 619–635. 15, 27, 40, 74, 117, 119

Li, Y., Ye, Z., and Rehg, J. M. (2015). Delving into egocentric actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 287–295. 1, 13, 74, 85, 86, 110

Lin, J., Gan, C., and Han, S. (2019). Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7083–7093. 32

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer. 91

Liu, C.-L., Lee, C.-H., and Lin, P.-M. (2010). A fall detection system using k-nearest neighbor classifier. *Expert Systems with Applications*, 37(10):7174–7181. 47

Liu, Y., Wei, P., and Zhu, S.-C. (2017). Jointly recognizing object fluents and tasks in egocentric videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2924–2932. 18

Lu, M., Li, Z.-N., Wang, Y., and Pan, G. (2019a). Deep attention network for egocentric action recognition. *IEEE Transactions on Image Processing*, 28(8):3703–3713. 15, 26, 27, 85, 87, 110, 111

Lu, M., Liao, D., and Li, Z.-N. (2019b). Learning spatiotemporal attention for egocentric action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0. 15, 28, 118, 119

Lu, Y. and Velipasalar, S. (2018). Human activity classification incorporating egocentric video and inertial measurement unit data. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 429–433. IEEE. 28

Ma, C.-Y., Kadav, A., Melvin, I., Kira, Z., AlRegib, G., and Peter Graf, H. (2018). Attend and interact: Higher-order object interactions for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6790–6800. 15, 20, 106, 108, 153

Ma, M., Fan, H., and Kitani, K. M. (2016). Going deeper into first-person activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1894–1903. 15, 26, 51, 52, 79, 80, 82, 84, 85, 86, 88, 110, 111, 117, 118, 120, 153

Mann, S. (1998). 'wearcam'(the wearable camera): personal imaging systems for long-term use in wearable tetherless computer-mediated reality and personal photo/videographic memory prosthesis. In *Digest of Papers. Second International Symposium on Wearable Computers (Cat. No. 98EX215)*, pages 124–131. IEEE. 11

Mastorakis, G. and Makris, D. (2014). Fall detection system using kinect's infrared sensor. *Journal of Real-Time Image Processing*, 9(4):635–646. 48

Materzynska, J., Xiao, T., Herzig, R., Xu, H., Wang, X., and Darrell, T. (2019). Something-else: Compositional action recognition with spatial-temporal interaction networks. *arXiv preprint arXiv:1912.09930*. 152, 153

Matsuo, K., Yamada, K., Ueno, S., and Naito, S. (2014). An attention-based activity recognition for egocentric video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 551–556. 15, 17, 73

McCandless, T. and Grauman, K. (2013). Object-centric spatio-temporal pyramids for egocentric activity recognition. In *BMVC*, volume 2, page 3. Citeseer. 15, 17

Meng, X.-L. and Rubin, D. B. (1993). Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2):267–278. 21

Miaou, S.-G., Sung, P.-H., and Huang, C.-Y. (2006). A customized human fall detection system using omni-camera images and personal information. In *1st Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare, 2006. D2H2.*, pages 39–42. IEEE. 47

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119. 42

Mishra, A. K., Aloimonos, Y., Cheong, L. F., and Kassim, A. (2011). Active visual segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):639–653. 17

Moltisanti, D., Wray, M., Mayol-Cuevas, W., and Damen, D. (2017). Trespassing the boundaries: Labeling temporal bounds for object interactions in egocentric video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2886–2894. 15, 38

Mubashir, M., Shao, L., and Seed, L. (2013). A survey on fall detection: Principles and approaches. *Neurocomputing*, 100:144–152. 46

Mueller, E. T. (2014). *Commonsense reasoning: an event calculus based approach*. Morgan Kaufmann. 18

Munappy, A., Bosch, J., Olsson, H. H., Arpteg, A., and Brinne, B. (2019). Data management challenges for deep learning. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 140–147. IEEE. 4

Nagarajan, T., Li, Y., Feichtenhofer, C., and Grauman, K. (2020). Ego-topo: Environment affordances from egocentric video. *arXiv preprint arXiv:2001.04583*. 15, 21

Nakamura, K., Yeung, S., Alahi, A., and Fei-Fei, L. (2017). Jointly learning energy expenditures and activities using egocentric multimodal signals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1868–1877. 15, 33

Nakatani, T., Kuga, R., and Maekawa, T. (2018). Preliminary investigation of object-based activity recognition using egocentric video based on web knowledge. In *Proceedings of the 17th International Conference on Mobile and Ubiquitous Multimedia*, pages 375–381. 15, 19

Narayan, S., Kankanhalli, M. S., and Ramakrishnan, K. R. (2014). Action and interaction recognition in first-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–518. 15, 23

Nawhal, M., Zhai, M., Lehrmann, A., and Sigal, L. (2019). Zero-shot generation of human-object interaction videos. *arXiv preprint arXiv:1912.02401*. 43

Nebel, J.-C., Florez-Revuelta, F., et al. (2018). Recognition of activities of daily living from egocentric videos using hands detected by a deep convolutional network. In *International Conference Image Analysis and Recognition*, pages 390–398. Springer. 15, 20

Nguyen, T.-H.-C., Nebel, J.-C., Florez-Revuelta, F., et al. (2016). Recognition of activities of daily living with egocentric vision: A review. *Sensors*, 16(1):72. 2, 4, 12, 16, 144

Nunez-Marcos, A., Azkune, G., and Arganda-Carreras, I. (2017). Vision-based fall detection with convolutional neural networks. *Wireless communications and mobile computing*, 2017. v, vi, 52, 53, 55, 56, 67, 68, 71

Núñez-Marcos, A., Azkune, G., and Arganda-Carreras, I. (2020). Object bounding box annotations for the GTEA Gaze+ dataset. 148

Ogaki, K., Kitani, K. M., Sugano, Y., and Sato, Y. (2012). Coupling eye-motion and ego-motion features for first-person activity recognition. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–7. IEEE. 15, 25

Perez-Rua, J.-M., Martinez, B., Zhu, X., Toisoul, A., Escorcia, V., and Xiang, T. (2020a). Knowing what, where and when to look: Efficient video action modeling with attention. *arXiv preprint arXiv:2004.01278*. 32

Perez-Rua, J.-M., Toisoul, A., Martinez, B., Escorcia, V., Zhang, L., Zhu, X., and Xiang, T. (2020b). Egocentric action recognition by video attention and temporal context. *arXiv preprint arXiv:2007.01883*. 15, 32

Perronnin, F. and Dance, C. (2007). Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE. 34

Pirsiavash, H. and Ramanan, D. (2012). Detecting activities of daily living in first-person camera views. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2847–2854. IEEE. 15, 17, 33, 40, 73

Planamente, M., Bottino, A., and Caputo, B. (2020). Joint encoding of appearance and motion features with self-supervision for first person action recognition. *arXiv preprint arXiv:2002.03982*. 15, 32

Planinc, R. and Kampel, M. (2013). Introducing the use of depth data for fall detection. *Personal and ubiquitous computing*, 17(6):1063–1072. 48

Poleg, Y., Arora, C., and Peleg, S. (2014). Temporal segmentation of egocentric videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2537–2544. 15, 24

Poleg, Y., Ephrat, A., Peleg, S., and Arora, C. (2016). Compact cnn for indexing egocentric videos. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE. 15, 24

Possas, R., Pinto Caceres, S., and Ramos, F. (2018). Egocentric activity recognition on a budget. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5967–5976. 15, 33

Rawtaer, I., Mahendran, R., Kua, E. H., Tan, H. P., Tan, H. X., Lee, T.-S., and Ng, T. P. (2020). Early detection of mild cognitive impairment with in-home sensors to monitor behavior patterns in community-dwelling senior citizens in singapore: Cross-sectional feasibility study. *Journal of Medical Internet Research*, 22(5):e16854. 3

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788. 18

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99. 20, 90

Ren, X. and Gu, C. (2010). Figure-ground segmentation improves handled object recognition in egocentric video. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3137–3144. IEEE. 17

Rougier, C., Meunier, J., St-Arnaud, A., and Rousseau, J. (2011). Robust video surveillance for fall detection based on human shape deformation. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(5):611–622. 47, 62, 64, 65

Ryoo, M. S. and Matthies, L. (2013). First-person activity recognition: What are they doing to me? In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, pages 2730–2737. 15, 23

Ryoo, M. S., Rothrock, B., and Matthies, L. (2015). Pooled motion features for first-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 896–904. 25, 33

Sahu, A., Bhattacharya, R., Bhura, P., and Chowdhury, A. S. (2020). Action recognition from egocentric videos using random walks. In *Proceedings of 3rd International Conference on Computer Vision and Image Processing*, pages 389–402. Springer. 15, 35

Sahu, A. and Chowdhury, A. S. (2018). Shot level egocentric video co-summarization. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 2887–2892. IEEE. 35

Sengto, A. and Leauhatong, T. (2012). Human falling detection algorithm using back propagation neural network. In *The 5th 2012 Biomedical Engineering International Conference*, pages 1–5. IEEE. 46

Shen, L., Yeung, S., Hoffman, J., Mori, G., and Fei-Fei, L. (2018). Scaling human-object interaction recognition through zero-shot learning. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1568–1576. IEEE. 42, 108, 122, 123, 130, 131, 153

Shiga, Y., Toyama, T., Utsumi, Y., Kise, K., and Dengel, A. (2014). Daily activity recognition combining gaze motion and visual features. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 1103–1111. 15, 32

Sigurdsson, G. A., Gupta, A., Schmid, C., Farhadi, A., and Alahari, K. (2018). Charades-ego: A large-scale dataset of paired third and first person videos. *arXiv preprint arXiv:1804.09626*. 40

Simonyan, K. and Zisserman, A. (2014a). Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576. v, 25, 26, 51, 52, 79

Simonyan, K. and Zisserman, A. (2014b). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. 54

Singh, S., Arora, C., and Jawahar, C. (2015). Generic action recognition from egocentric videos. In *2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*, pages 1–4. IEEE. 15, 24

Singh, S., Arora, C., and Jawahar, C. (2016). First person action recognition using deep learned descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2620–2628. 15, 31, 32

Song, S., Chandrasekhar, V., Cheung, N.-M., Narayan, S., Li, L., and Lim, J.-H. (2014). Activity recognition in egocentric life-logging videos. In *Asian Conference on Computer Vision*, pages 445–458. Springer. 15, 24

Soomro, K., Zamir, A. R., and Shah, M. (2012). Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*. 55

Spriggs, E. H., De La Torre, F., and Hebert, M. (2009). Temporal segmentation and activity classification from first-person sensing. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 17–24. IEEE. 15, 32

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958. 56

Sudhakaran, S., Escalera, S., and Lanz, O. (2019a). Fbk-hupba submission to the epic-kitchens 2019 action recognition challenge. *arXiv preprint arXiv:1906.08960*. 36

Sudhakaran, S., Escalera, S., and Lanz, O. (2019b). Hierarchical feature aggregation networks for video action recognition. *arXiv preprint arXiv:1905.12462*. 36

Sudhakaran, S., Escalera, S., and Lanz, O. (2019c). Lsta: Long short-term attention for egocentric action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9954–9963. 15, 26, 28, 36, 118, 119

Sudhakaran, S. and Lanz, O. (2018). Attention is all we need: Nailing down object-centric attention for egocentric activity recognition. *arXiv preprint arXiv:1807.11794*. 15, 26, 27, 85, 110, 114, 119, 138

Sun, L., Klank, U., and Beetz, M. (2009). Eyewatchme—3d hand and object tracking for inside out activity analysis. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 9–16. IEEE. 17

Sundaram, S. and Cuevas, W. W. M. (2009). High level activity recognition using low resolution wearable vision. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 25–32. IEEE. 15, 23

Surie, D., Pederson, T., Lagriffoul, F., Janlert, L.-E., and Sjölie, D. (2007). Activity recognition using an egocentric perspective of everyday objects. In *International Conference on Ubiquitous Intelligence and Computing*, pages 246–257. Springer. 15, 16, 17

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9. 41

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., and Liu, C. (2018). A survey on deep transfer learning. In *International Conference on Artificial Neural Networks*, pages 270–279. Springer. 4, 55

Tang, Y., Wang, Z., Lu, J., Feng, J., and Zhou, J. (2018). Multi-stream deep neural networks for rgb-d egocentric action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(10):3001–3015. 15, 30

Tekin, B., Bogo, F., and Pollefeys, M. (2019). H+ o: Unified egocentric recognition of 3d hand-object poses and interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4511–4520. 15, 19

Tran, A. and Cheong, L.-F. (2017). Two-stream flow-guided convolutional attention networks for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 3110–3119. 116

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497. 31

Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171. 89

Vallejo, M., Isaza, C. V., and Lopez, J. D. (2013). Artificial neural networks as an alternative to traditional fall detection methods. In *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1648–1651. IEEE. 46

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008. 20

Verma, S., Nagar, P., Gupta, D., and Arora, C. (2018). Making third person techniques recognize first-person actions in egocentric videos. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2301–2305. IEEE. 15, 27, 119

Vishwakarma, V., Mandal, C., and Sural, S. (2007). Automatic detection of human fall in video. In *International Conference on Pattern Recognition and Machine Intelligence*, pages 616–623. Springer. 47

Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2013). Dense trajectories and motion boundary descriptors for action recognition. *International journal of computer vision*, 103(1):60–79. vi, 84, 85, 86, 87, 110

Wang, H. and Schmid, C. (2013). Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3551–3558. 32, 84, 85, 86, 110

Wang, J., Chen, Y., Hao, S., Peng, X., and Hu, L. (2019a). Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, 119:3–11. 3

Wang, K., Cao, G., Meng, D., Chen, W., and Cao, W. (2016a). Automatic fall detection of human in video using combination of features. In *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1228–1233. IEEE. 63, 64

Wang, L., Xiong, Y., Wang, Z., and Qiao, Y. (2015). Towards good practices for very deep two-stream convnets. *arXiv preprint arXiv:1507.02159*. vi, 52, 54, 55, 56

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Van Gool, L. (2016b). Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer. vi, 54, 55, 56, 108

Wang, S., Chen, L., Zhou, Z., Sun, X., and Dong, J. (2016c). Human fall detection in surveillance video based on pcanet. *Multimedia tools and applications*, 75(19):11603–11613. 63, 64

Wang, W., Zheng, V. W., Yu, H., and Miao, C. (2019b). A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–37. 2

Wang, X. and Gupta, A. (2018). Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 399–417. 152

Wang, X., Wu, Y., Zhu, L., and Yang, Y. (2019c). Baidu-uts submission to the epic-kitchens action recognition challenge 2019. *arXiv preprint arXiv:1906.09383*. 108, 120

Wang, X., Wu, Y., Zhu, L., and Yang, Y. (2020). Symbiotic attention with privileged information for egocentric action recognition. *arXiv preprint arXiv:2002.03137*. 15, 29, 118, 119

Wray, M. and Damen, D. (2019). Learning visual actions using multiple verb-only labels. *arXiv preprint arXiv:1907.11117*. 15, 27

Wray, M., Larlus, D., Csurka, G., and Damen, D. (2019). Fine-grained action retrieval through multiple parts-of-speech embeddings. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 450–459. 41

Wray, M., Moltisanti, D., and Damen, D. (2018). Towards an unequivocal representation of actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1127–1131. 15, 26, 41, 85, 87, 110, 152

Wray, M., Moltisanti, D., Mayol-Cuevas, W., and Damen, D. (2016). Sembed: Semantic embedding of egocentric action videos. In *European Conference on Computer Vision*, pages 532–545. Springer. 15, 35, 41, 152

Wray, M., Moltisanti, D., Mayol-Cuevas, W., and Damen, D. (2017). Improving classification by improving labelling: Introducing probabilistic multi-label object interaction recognition. *arXiv preprint arXiv:1703.08338*. 15, 38

Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and Woo, W.-c. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pages 802–810. 27, 115, 125

Yan, Y., Ricci, E., Liu, G., and Sebe, N. (2014). Recognizing daily activities from first-person videos with multi-task clustering. In *Asian Conference on Computer Vision*, pages 522–537. Springer. 15, 25

Yan, Y., Ricci, E., Liu, G., and Sebe, N. (2015). Egocentric daily activity recognition via multitask clustering. *IEEE Transactions on Image Processing*, 24(10):2984–2995. 15, 33

Yu, C. and Ballard, D. H. (2002a). Learning to recognize human action sequences. In *Proceedings 2nd International Conference on Development and Learning. ICDL 2002*, pages 28–33. IEEE. 15, 22

Yu, C. and Ballard, D. H. (2002b). Understanding human behaviors based on eye-head-hand coordination. In *International Workshop on Biologically Motivated Computer Vision*, pages 611–619. Springer. 15, 22

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. (2015). Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*. 4

Yu, H., Jia, W., Li, Z., Gong, F., Yuan, D., Zhang, H., and Sun, M. (2019a). A multisource fusion framework driven by user-defined knowledge for egocentric activity recognition. *EURASIP Journal on Advances in Signal Processing*, 2019(1):14. 15, 34

Yu, H., Pan, G., Pan, M., Li, C., Jia, W., Zhang, L., and Sun, M. (2019b). A hierarchical deep fusion framework for egocentric activity recognition using a wearable hybrid sensor system. *Sensors*, 19(3):546. 15, 28

Zach, C., Pock, T., and Bischof, H. (2007). A duality based approach for realtime tv-l 1 optical flow. In *Joint Pattern Recognition Symposium*, pages 214–223. Springer. 52

Zaki, H. F., Shafait, F., and Mian, A. (2017). Modeling sub-event dynamics in first-person action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7253–7262. 15, 21

Zerrouki, N., Harrou, F., Houacine, A., and Sun, Y. (2016). Fall detection using supervised machine learning algorithms: A comparative study. In *2016 8th International Conference on Modelling, Identification and Control (ICMIC)*, pages 665–670. IEEE. 47, 63, 64, 65

Zerrouki, N. and Houacine, A. (2017). Combined curvelets and hidden markov models for human fall detection. *Multimedia Tools and Applications*, pages 1–20. 65

Zerrouki, N. and Houacine, A. (2018). Combined curvelets and hidden markov models for human fall detection. *Multimedia Tools and Applications*, 77(5):6405–6424. 48, 63, 64

Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Zhang, Z., Lin, H., Sun, Y., He, T., Mueller, J., Manmatha, R., et al. (2020). Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*. 116

Zhang, H.-B., Zhang, Y.-X., Zhong, B., Lei, Q., Yang, L., Du, J.-X., and Chen, D.-S. (2019). A comprehensive survey of vision-based human action recognition methods. *Sensors*, 19(5):1005. 3

Zhang, Y. C., Li, Y., and Rehg, J. M. (2017). First-person action decomposition and zero-shot learning. In *2017 IEEE erence on Applications of Computer Vision (WACV)*, pages 121–129. IEEE. 15, 33, 40

Zhong, C., Reibman, A. R., Cordoba, H. M., and Deering, A. J. (2019). Hand-hygiene activity recognition in egocentric video. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6. IEEE. 15, 29

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2016a). Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929. 27

Zhou, Y., Ni, B., Hong, R., Yang, X., and Tian, Q. (2016b). Cascaded interactional targeting network for egocentric video analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1904–1913. 15, 35, 111

Zuo, Z., Wei, B., Chao, F., Qu, Y., Peng, Y., and Yang, L. (2019). Enhanced gradient-based local feature descriptors by saliency map for egocentric action recognition. *Applied System Innovation*, 2(1):7. 15, 34

Zuo, Z., Yang, L., Peng, Y., Chao, F., and Qu, Y. (2018). Gaze-informed egocentric action recognition for memory aid systems. *IEEE Access*, 6:12894–12904. 15, 34, 117

Declaration

I, Adrián Núñez Marcos, herewith declare that this dissertation is my own original work, carried out as a doctoral student at the University of Deusto. All assistance received and notions from other sources have been identified as such, acknowledging their correspondent contributions and citing them properly.

This work contains no material which has been presented in identical or similar form to any examination board, except where due acknowledgement is made in the dissertation.

This dissertation was finished writing on September 15$^{\text{th}}$, 2020.