A Modular Approach for the Detection and Interconnection of Objects, Hands, Locations, and Actions for Egocentric Video Understanding

---

Georgios Kapidis

A Modular Approach for the Detection and Interconnection of Objects, Hands, Locations, and Actions for Egocentric Video Understanding

PhD Thesis, Utrecht University, the Netherlands

ISBN: 978-94-6423-273-8

Cover: Fenna Schaap

Print: ProefschriftMaken || www.proefschriftmaken.nl

A Modular Approach for the Detection and Interconnection of Objects, Hands, Locations, and Actions for Egocentric Video Understanding

Een modulaire aanpak voor de detectie en interconnectie van objecten, handen, locaties, en acties voor het begrijpen van egocentrische video
(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit Utrecht
op gezag van de
rector magnificus, prof.dr. H.R.B.M. Kummeling,
ingevolge het besluit van het college voor promoties
in het openbaar te verdedigen op
woensdag 9 juni 2021 des middags
te 2.15 uur

door

GEORGIOS KAPIDIS

geboren op 26 april 1990
te Thessaloniki, Griekenland

Promotor:
Prof. dr. R.C. Veltkamp

Copromotor:
Dr. R.W. Poppe

This thesis was accomplished with financial support from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 676157.

Summary

The topic of this dissertation is the analysis and understanding of egocentric (first-person) videos with respect to the performed human actions of the camera wearer, in a structured and automatic manner. Perhaps, the most identifying characteristic of the egocentric perspective is that it provides an information-rich view of the scene that the person holding the camera experiences. The resulting scenes are often indicative of the location of the persons and the activities they undertake. Recognition is based on high-level information, such as the hands of the camera wearer and the objects that are being manipulated, as well as low-level features made available through data-learning methods. In this thesis, we use deep convolutional neural networks trained on egocentric images, video segments, and/or (a)synchronously acquired high-level features of the scene as the backbone of action classification models. We demonstrate that the training process and architecture of the models is detrimental to their success; a topic largely investigated with the application of multitask learning, measuring the effect of a variety of learnable outputs to the final action recognition result. We additionally pursued the combination of video data from a variety of sources simultaneously. In the context of the thesis, it is called multi-dataset multitask learning and refers to a novel way to combine related and unrelated data sources to improve egocentric action recognition quality.

Samenvatting

Het onderwerp van dit proefschrift is het analyseren en begrijpen van egocentrische (first-person) video's met betrekking tot de uitgevoerde menselijke handelingen van de drager van de camera, op een gestructureerde en automatische manier. Misschien wel het meest karakteristieke kenmerk van het egocentrische perspectief is dat het een informatief overzicht geeft van de scène die de persoon met de camera ervaart. De resulterende scènes zijn vaak indicatief voor de plaats waar de persoon zich bevindt en de activiteiten die hij of zij onderneemt. Herkenning is gebaseerd op betekenisvolle informatie, zoals de handen van de drager van de camera en de voorwerpen die worden gemanipuleerd, en op data die beschikbaar worden via dataleermethoden. In deze thesis worden diepe convolutionele neurale netwerken, getraind op egocentrische beelden, videosegmenten, en/of (a)synchroon verworven kenmerken informatie over de scène gebruikt als basis voor modellen voor de classificatie van actieviteiten. We laten zien dat het trainingsproces en de architectuur van de modellen bepalend zijn voor hun succes; een onderwerp dat grotendeels wordt onderzocht met de toepassing van multitask leren, waarbij het effect van een verscheidenheid aan geleerde outputs op het eindresultaat van actieherkenning wordt gemeten. Ook onderzoeken we in deze thesis het gebruik van een combinatie van videogegevens uit verschillende bronnen tegelijk. In de context van dit proefschrift wordt dit multitask learning genoemd en verwijst het naar een nieuwe manier om gerelateerde en niet-gerelateerde gegevensbronnen te combineren om de egocentrische herkenning van acties te verbeteren.

Acknowledgments

> Everything existing in the universe is the fruit of chance and necessity.
>
> Democritus

About four and a half years ago the opportunity to pursue a PhD degree presented itself. I have to admit that it was not clear to me at first whether I should attempt to go through with it or not. Having been a student for most of my life, it seemed like a fitting path that would allow me to continue my pursuit of knowledge and grow myself as part of the academic world. It was a difficult decision to make as I would have to commit to a different life in another country, leaving family and friends behind. In my mind the PhD was, and continues to be, a worthy cause, deserving of the sacrifices one needs to make, in order to attain it.

Although the PhD tends to be a solitary adventure, I was never alone. From the very beginning, I was welcomed by Elsbeth van Dam and Ronald Poppe who helped make the transition as smooth as possible.

Elsbeth introduced me at Noldus, where she was both a mentor and a friend throughout my time at the company. For all the time we have spent comprehending science in adjacent desks, for every piece of advice she has offered, and for all her support I cannot thank her enough. Roos, Alexander and every other colleague at Noldus, it is great to have met you and have spent time with you, from chatting at corridors, sharing stories, and having a beer after work, to skiing on the Dutch hills and canoeing in the canals near Wageningen at our always well-designed outings. My deepest thanks go to Lucas Noldus for accepting me into his company and for providing the necessary tools to conduct my research work.

Ronald, who has been my guide in all things academic, I would like to thank you for your stoic patience with me, your cheerful tone in every situation and for all the plans we have made in order to reach this point in time where I get to write these words. You showed me what it means and what it takes to do a PhD. I couldn't have asked for more. I would also like to express my gratitude to Remco Veltkamp, whose

guiding hand was always there to promote my efforts as I walked through the PhD path.

My time was divided between Noldus and the University where I was part of the vision group. It was a great opportunity for me to socialize with other PhD students and share our works and struggles. I would like to thank Alex for being such a great lab partner. We came up with quite a few paper worthy ideas and had plenty of laughs along the way. I look back at my ping-pong breaks with Giannis, the discussions with Ilya and the life tips from Miroslav.

As part of the ACROSSING fellowship I was fortunate to travel around Europe for our project meetings, secondments and conferences. I collaborated with a number of brilliant young scientists, with whom I shared this great opportunity. My appreciation goes to Luke (Liming) Chen and every other member of the consortium for their efforts to organize this project and turn it into a success.

I would like to make a special mention to my former supervisor Ioannis Pratikakis at Democritus University of Thrace, Greece, who guided me through my Master's degree, instilled me with a passion for research and encouraged me to enter the academic world.

My friends back in Greece were always a pleasant distraction from the PhD struggles and our online sessions would always bring a positive light in every situation. Thanks guys!

My parents, Antonis and Sofia have always been there to support my choices. They taught me about the importance of hard work and attention to detail for success and showed me how to accept, appreciate and learn from all of life's hardships. My sister Alexandra looked out for me ever since I existed, despite me being the annoying younger sibling!

A special place in my heart is kept for Marina who has been my companion and my strength in everything. You went through this adventure just as much as I did and I am grateful for it. Thank you for being there for me and you know I am there for you too.

Contents

1 Introduction

1.1 Egocentric Vision

From the abacus to robotic exoskeletons, science always paved the way to overcome the mental and physical limitations of the human body. Moving into the 21st century, digital technology has advanced to surround and support humans in many aspects of daily life. One of the steps forward has been to promote technological equipment into an inseparable companion with access to and understanding of human experiences. This is the case of wearable devices which, due to their proximity to the human body, are able to accurately monitor the world in our immediate vicinity. Humans perceive their environment through five senses; sight, hearing, smell, taste, and touch, and process their inputs continuously with their brains. The responses are the conscious and subconscious actions that allow us to navigate the world. Devices, on the other hand, are mostly capable of capturing visual, audible, and tactile inputs with limited capability in processing and understanding them the way humans do. Until they become adept at perceiving the world in a manner that resembles that of humans, they are reduced to a passive role with decision making out of reach.

In this dissertation we focus on understanding human actions from video captured from the first-person perspective. Success in our task would mean that decision making may be shared and immediate assistance would be available especially in cases where human perception may be limited, as for example, in old age. We utilize features comprehensible by humans as well as features made available through data learning approaches. The video view is called "egocentric" from the words "ego" meaning self or person and "center" denoting that the protagonist of the video is the person recording it. Although not explicitly stated in this definition, egocentric videos generally do not show the person holding the camera, rather they capture the person's view of their surroundings, in practice providing an additional set of eyes to enhance the human experience.

The use of egocentric cameras is alluring as they are becoming smaller and less intrusive, two essential qualities for wearables targeting everyday use. They can be used as an alternative to expensive multi-sensor installations that convert an existing house into a smart home. By taking advantage of recent advances in machine

learning, a single sensor – the egocentric camera – will capture information about a person's location in the house, sociability or loneliness, the performed activity and even imminent dangers stemming from the latter.

Egocentric cameras can be placed above the forehead, attached on either side of the head, on top of a shoulder or on the chest, facing forward. Furthermore, they can be attached to Virtual/Augmented reality headsets to enhance comprehension of the user's surroundings. Lately, the most prominent camera positions for capturing egocentric videos are either above the forehead or strapped to the chest. Both positions allow for a field of view that clearly includes the hands of the camera wearer when they are raised in front of the body to perform a movement or an action. Generally, the view also includes the objects that are visible and/or manipulated, providing rich information about the immediate environment. In this work, we develop feature- and data-oriented techniques to detect hands and objects with the aim of understanding the human actions and environments that can be associated with them.

1.2 ACROSSING Innovative Training Network

This thesis is developed as part of the ACROSSING project. ACROSSING stands for "**A**dvanced Te**C**hnologies and Platfo**R**m f**O**r **S**marter A**S**sisted Liv**ING**". It is an Innovative Training Network (ITN) and part of the Marie Skłodowska Curie Actions funded under the European Commission's Horizon 2020 research and innovation program.

The aim of the ACROSSING project is to bring together a multi-disciplinary network of 26 leading European research groups, industry partners, and user organisations, to develop an open Smart Home technology infrastructure and train 15 Early Stage Researchers (ESRs) across sectors on concepts and methodologies of Smart Homes.

This thesis delineates the scientific aspects of the project of ESR 10: "Egocentric subject monitoring and tracking in Smart Homes". These involve the development of a tracking system that uses a mobile camera mounted on a subject to capture and analyze egocentric video. The video is utilized to perform object detection for location and activity determination in Smart Home environments. In this context, the algorithms and methods for video understanding can be used to monitor the well-being of Smart Home residents and encourage independent living with the assistance of modern technology.

1.3 Thesis Objectives

The scope of our work is to research the building blocks of a system that leads to video understanding from the perspective of the person recording the video. Given that egocentric videos employ characteristic views of a scene we aim to answer two fundamental questions:

- Can accurate recognition of visited locations and performed actions be achieved based only on the outputs of object and egocentric hand detection systems?
- Is it possible to improve on the action recognition performance by utilizing end-to-end video recognition approaches? Instead of relying on intelligible information and extracted features about objects and hands throughout the video, can we use it directly as input to provide action inference without intermediate steps?

The first question pertains to the step-wise approach that focuses on the explicit cues that characterize a scene or an action, such as the presence or the movements of indicative objects. The input video signal is broken down into understandable notions based on which the context of a scene can be clearly expressed. For example, this approach may lead to understandable narratives such as "lifting a cup towards the mouth results in a drinking action" or "seeing a sink and a fridge signifies that the person is located in a kitchen". We focus on object-based video understanding in Chapters 3 and 4 where we rely on explicit detections of objects and hands to identify locations and actions.

The second question refers to methods based on deep learning where the features that lead to inference are not as clear and are not necessarily the result of an explicit detection process. Rather, models with millions of learnable parameters acquire the values of features that best describe the dataset that defines the task to be learned. In this light, we focus on the task of egocentric action recognition using large scale neural network structures. These networks are given videos for input and output specific inferences without providing a clear explanation about the characteristics that define them. Initially, we begin with networks that classify videos directly into actions, without interpreting the learned features that lead to a specific result. Eventually, we modify the network to focus on the humanly intelligible features that characterize an action. We do this in a multitask learning fashion that aims to enhance the networks' knowledge by "nudging" it towards the information that needs to be acquired. Chapters 5 and 6 employ deep networks for visual feature extraction and classification into actions.

1.4 Thesis Outline

In the remainder of this chapter, we present the outline of this thesis with a short description of the work presented in each chapter.

In *Chapter 2* we present an overview of the related work in egocentric video perception. We make a distinction between feature-based and network-based methods for location and action classification and identify the progress from the third-person vision domain that has inspired the advances in egocentric vision. Furthermore, we introduce related work on Multitask Learning and Multi-dataset training approaches that characterize our approach in Chapters 5 and 6. Lastly, we provide a synopsis of the public datasets that define the context of our research.

In *Chapter 3* we introduce our approach for object-based location and activity recognition. We describe the use of visual object detectors the output of which is analyzed to detect indoor locations and activities of daily living. We employ Artificial Neural Networks (ANN) for inference on single frames and Long Short-Term Memory Networks (LSTM) to analyze the temporal associations of the detected objects. Furthermore, we reflect on the effect of object detection quality on the location and activity recognition outputs. Parts of this chapter are published in [68, 70].

Chapter 4 follows a similar cue-based approach with a focus on the temporal associations of egocentric hand motions. We describe our pipeline to detect and track hand regions. Furthermore, we classify the progression of hand coordinates into actions using LSTMs. We proceed with combining object and hand tracking information to further improve the inference capability of our model. Parts of this chapter are published in [66].

In *Chapter 5* we develop a model for end-to-end action recognition from video. The novelty of our approach is based on the concept of Multitask Learning (MTL). In MTL we train a single network to produce multiple inferences about a video in a way similar to training multiple networks, each for a single inference. MTL states that joint training for multiple tasks on the same network can be beneficial for performance as long as the tasks are describing related concepts. We investigate the association between action recognition, hand detection and gaze estimation in egocentric videos. This chapter is published in [67].

In *Chapter 6* we extend the MTL paradigm to tasks from a variety of related datasets using a single model. Our aim is to explore the learning capacity of neural network architectures when tasked to address similar objectives jointly. We investigate the

relationship between egocentric food preparation datasets, as well as more generic actions when captured from both the first- and the third-person perspectives. Parts of this chapter are published in [69].

Finally, *Chapter 7* offers a general discussion on our introduced methods and their results and concludes the thesis. Furthermore, it provides directions for future work in egocentric video understanding and multitask learning.

2 Related Work

Egocentric vision is an integral part of computer vision with applications in conventional fields such as activity recognition [96] and video summarization [27] as well as in more elaborate domains, for instance social interaction analysis [169], guideline generation for visual assistance [25] and infant visual attention [77]. In this chapter, we discuss related work about location detection (Sec. 2.1) and activity and action recognition (Sec. 2.2) in egocentric videos. We present feature-based approaches in Sec. 2.2.1. We present recent breakthroughs in network-based approaches in third-person vision in Sec. 2.2.2 and associate them with their equivalents in first-person video recognition (Sec. 2.2.3). Furthermore, we focus on the effects of hand and object detection for egocentric actions (Sec. 2.2.4). Multitask Learning and Multi-dataset training for computer vision tasks are discussed in Sec. 2.2.5 and 2.2.6, respectively. Lastly, we present an overview of open egocentric video datasets (Sec. 2.3) concentrating on the ones used in this dissertation.

2.1 Egocentric Location Classification

Recognition of locations, in terms of mapping the surrounding area with explicit attributes or representative image features is actively under research in egocentric vision. The main goals of location classification are localization of the camera wearer with respect to their immediate environment [30, 81, 93], providing information about it [2], facilitating execution of elaborate tasks [116] or as a supporting mechanism that enhances understanding of actions [9]. In [9], scene illumination and distinct location features were learned in an unsupervised way to enhance the usability of wearable cameras for hand detection. Location recognition was indirectly the task in [2] where images of the user's field of view were captured with a Google Glass device, which retrieved information about the buildings in sight. In [81], the combination of a camera and a 2D laser scanner were applied to register images, queried by users, into the real-world coordinate system. A multi-view indoor localization system based on image features was proposed in [93]. It distinguished indoor locations using self-similarity matrices across extracted images to correlate among views of the scene from multiple perspectives. Afterwards, it learned the

equation system through these features and when a new query image was given, it provided the camera's relative position in the scene and its orientation. Location classification was described in [30]. Visual recognition was supported by low-level features and semi-supervised training procedures to take advantage of sparsely annotated data. The combination of wearable egocentric stereo cameras and inertial sensors was considered in [116] to map an outdoor workspace with image features and provide routing guidance for humans aiming to execute specific tasks in it. These methods consider low-level image features to describe the scene. In contrast, in Chapter 3 we use high-level features; the detected objects on every frame. We do not depend on previous knowledge about the specific locations that users find themselves into, but build our inference upon characteristic objects that are detected in them.

Locations in egocentric videos are not a static quality of the scene, but change dynamically following user movements in the environment. In this context, the current location can be modeled continuously while taking into account previously acquired features and past locations, allowing for more robust detections. In [41], personal locations of interest were highlighted by temporally segmenting egocentric videos based on the user's location. Location models were trained on user-provided frame samples of locations, denoting those that captured user interest as positive ones. Subsequently, the system learned to reject the frames that did not depict locations significant to the user's interest. User-specific locations were further analyzed in [40] as part of a user's daily routine. Classification of image sequences into locations relied on either convolutional features or hand-crafted ones. In [158, 159], the combined improvement of object detection and scene identification was investigated. Initially, scene identification was performed using temporally associated convolutional features. This location related information was used to improve object detection performance, by linking objects to specific locations. This demonstrated the concept of linking locations and objects in egocentric scenarios. Eventually, they showed that by using LSTM networks to learn the temporal associations of the detected objects directly, it was possible to further improve object detection performance. Our work in Chapter 3 is the opposite of this concept. We use the object detections to infer locations and activities. We exploit the associations between them and show the effects of objects detected out of their spatial context. Finally, we show that temporal modeling of objects improves location classification performance.

2.2 Egocentric Activity and Action Recognition

Human action recognition from video is a computer vision task with multiple challenges ranging from the innate difficulties of video analysis, such us illumination changes and motion blur, to the adversities of activity recognition, including class variability, viewpoint variation and scarcity of training data [113].

Even before the breakthrough of egocentric vision, the connection between objects, hand movements, and human motion was studied to recognize human tasks [100] in setups that differ from traditional third-person videos. Typically, the camera was positioned on top of the human to ideally capture what was intuitively deemed important for the attainment of actions (Fig. 2.1a). The egocentric vision paradigm was more clearly established later [33, 65, 122, 123, 142], using wearable cameras with the distinct characteristic of promoting a clear view of, most notably, the user's hands and the manipulated objects (Figs. 2.1b-c).

(a) Top view of the scene from [100] ©1999 IEEE (b) Head mounted egocentric view rotated from [142] ©2009 IEEE. (c) Head mounted egocentric view from [33] ©2011 IEEE.

Fig. 2.1.: Progression of egocentric views.

2.2.1 Feature-based Egocentric Action Recognition

Egocentric action recognition has seen incremental improvements over the years with the prominent works of [4, 25, 33, 38, 85, 86, 96, 110, 125]. Originally, feature-based techniques [86, 140, 169, 174] were developed to explicitly model and capitalize on the inherent characteristics of first-person videos such as motions [86, 125, 138, 140, 168, 174], ego-motion [86, 106, 169], human gaze [86, 98, 106, 168, 176], and the presence and movement of hands [33, 86, 98, 110, 169] and objects [33, 86, 98, 110, 169]. Hands, objects, ego-motion, and their interrelationships have been established as some of the most prominent characteristics for egocentric action recognition [33, 65, 142]. In this observation lies the origin of the hand-crafted feature approaches that prevail in earlier works in egocentric vision.

The first works on egocentric activity understanding focused on the intrinsic information that defines the egocentric vision paradigm, *i.e.*, the visual characteristics of hands and objects from the first-person perspective. Fathi *et al.* [33] modeled the relationships between hands, objects, and actions using extracted visual features to model activities. They showed through bottom-up and top-down models the mutual improvements that these relationships offer towards improving the initial hand and object detections. The importance of the detected objects and their interactions for the detection of activities was highlighted in [110], where activity recognition improved through additional information about objects that were either passive or actively engaged with in the scene. In [34], the gaze of the camera wearer was used to define the salient areas in first-person views, recognizing that egocentric actions were further correlated with modalities that describe human attention in the video. Throughout Chapter 4 we base our work on the concept that the relationship between hands and objects is a fundamental aspect towards egocentric action recognition and understanding.

Ego-motion is an additional source of information that describes egocentric actions. It comprises the rapid motions of the video view due to the unstructured movements of the camera wearer and the motions of the viewed objects. In [125], global and local motions were considered as the basis of features to describe egocentric actions. Motion-based features for egocentric action recognition were analyzed in [86] where the importance of motion and object cues, hand and head movements as well as gaze were evaluated in various combinations. Scene, object, hand, and head movements were modeled with dense trajectories, color histograms and local binary patterns and were used as inputs to Support Vector Machines for classification of food preparation activities. The aim was to measure the individual effect of new features on activity classification performance through gradual aggregation, based on the idea that their contributions are complementary. For the interested reader, a review of feature-based approaches in egocentric action recognition can be found in [104].

2.2.2 Advances in Third-person Activity Recognition

Recent work in third-person vision [113, 144] follows the successful employment of deep network approaches. Deep networks allow for the employment of millions of parameters with the ability to identify subtle patterns in the input data, at multiple levels of abstraction, facilitating recognition of actions and activities. The trade-off is a requirement for large amounts of training data and annotations, which are not

always available or accessible, or even compatible to the task that is being targeted. Of the latter, we investigate the problem of data compatibility in Chapter 6.

Regarding deep networks, we highlight the work of Karpathy *et al.* [72] who used 2D-CNN architectures to classify video frames and, in order to incorporate information from multiple frames, explored various fusion schemes to enhance the classification output. Other approaches include the use of attention mechanisms[43] and two-stream networks [45, 133, 157, 160] that captured appearance and motion in images with spatial and motion streams trained on single or multiple frames concurrently.

Further attempts to take advantage of the temporal information that reveals the actions or activities in videos considered the use of recurrent units attached to frame-wise feature-extracting CNNs [29, 87]. These recurrent units were usually applied after the last convolutional layer in the form of a memory module that associated features from past frames with those from the most recent one. We employ Long Short-term Memory (LSTM) layers in Chapters 3 and 4 to integrate past and present detections to recognize locations and actions. More recent approaches in video activity recognition used 3D-CNNs [17, 20, 155, 156]. Here, video frames were modeled as a result of convolutional kernels being learned not only on the spatial dimension of images, but also on the changing pixel values in frame sequences. We find that using a 3D-CNN to capture patterns in the temporal dimension also works well for egocentric action recognition. We elaborate on the use of 3D-CNNs for egocentric action recognition in Chapters 5 and 6.

Since the introduction of large-scale image [28] and video [72, 73] datasets, convolutional neural networks (CNNs) have consistently produced state-of-the-art results [17, 36, 49, 72, 87, 148, 155, 156, 160, 161] for third-person image and video recognition tasks. Likewise, CNN-based approaches have been adopted and adapted to tackle first-person video understanding [4, 7, 74, 86, 96, 130, 153].

2.2.3 Advances in First-person Activity Recognition

Spatial networks The widespread use of CNNs in third-person vision was followed by their extensive application into egocentric action and activity recognition [3, 4, 7, 96, 112, 114]. Earlier approaches handled CNN features as an additional modality to handcrafted ones [169] or for classification of feature-based region proposals for object detection [3]. A convolutional feature extractor from images was used to categorize egocentric actions in [126]. In [163], convolutional features from video frames were used to produce embeddings that were semantically linked among

videos and were used as the basis to model relationships between objects and actions in order to classify them. The videos were mapped onto a semantic graph, with nodes for each freely annotated object and action. The graph was trained on the visual similarities between node features. Finally, the activities in unseen videos were recognized as the probability distributions among the existing action labels.

Fully convolutional approaches view action recognition as a learning-based problem with CNNs being used as appearance [67, 172] and motion [1] feature extractors. More data hungry methods used multi-stream deep networks that utilized optical flow along RGB images as input modalities [37, 61, 96, 112, 139] to be able to focus on motion. A motion-based network that relied on optical flow as training input is described in [112]. Each frame was divided into a set of grid cells and a single sparse optical flow value was extracted for each cell per frame. These flow values were then used as training data for a CNN and were classified into distinct activity classes. In [31, 61] optical flow was employed to detect salient regions, which were cropped from the original RGB frames and were given to the network as a second, more focused RGB stream. In [96], a two-stream network was trained to capture appearance and motion features. The appearance stream was pretrained for the tasks of hand pose estimation and segmentation and was finetuned for object localization, to find the regions of interest in images. The motion stream was trained on optical flow to predict short-term actions. Finally, both feature representations were combined with late fusion to predict short-term actions, objects, and activities. In [134], a three-stream architecture trained on egocentric features from hand masks, head motion, and saliency maps for the first stream, and raw RGB images and optical flow maps for the other two was adapted to recognize actions. End-to-end methods also include [4]. In order to recognize actions a CNN was trained on pairs of frames and was jointly optimized for the training objectives of action recognition, object segmentation and inter-frame object interactions. In Chapters 5 and 6 we further highlight the improvements from utilizing specific egocentric cues such as hand movements and gaze-based visual saliency on top of image-based features, but from a contrasting perspective. Instead of using them as input, we consider them as additional learnable tasks with the advantage of not requiring the extra information at test time.

A number of distinct input modalities have been employed to assist in egocentric action recognition. These include depth [7, 151], egocentric cues comprising hand [66, 129, 134] and object regions [66, 68, 164], head motions [134] and gaze-based saliency maps [129, 134], sensor-based modalities [101, 114, 139] and sound [18, 74, 167]. Typically, these methods require specialized sensors such as depth cameras,

eye trackers, accelerometers, or inertial measurement units for the additional inputs, whereas sound is provided from the built-in microphones of the camera.

Spatio-temporal with recurrence Following the advances in third-person vision, a number of egocentric methods have utilized recurrent modules to enhance inference with information acquired from preceding frames. Typically, a 2D-CNN backbone is used to capture appearance features from each frame in a frame sequence. Each frame's appearance features are further processed with recurrent layers in order to additionally learn their temporal associations. A set of tasks that have been addressed with this combination of network-based feature extractors and recurrent networks include [54] for temporal action proposal generation, [159] for action related scene identification and [80, 139] for action recognition.

Recurrent with attention The temporal aspect of videos has been further studied with recurrent attention mechanisms [13, 39, 57, 85, 88, 94, 98, 129, 149, 150, 152, 164, 170, 176] that aim to find the most informative parts in images (spatial attention) or the most informative frames throughout videos (temporal attention). An encoder-decoder scheme was described in [13] for textual description of videos. Here, the current event's frames were encoded into convolutional features and modeled temporally with LSTM. From the current and previous step's embedding, an attention mechanism selected the features to be decoded as the optimal textual description of the current activity. The attention mechanism in [129] focused on the frames that carry the action-specific information by learning the associations between the input gaze, the detected objects, and the segmented hands. The combined focus on these regions allowed the network to discard redundant frames from the input video segment that would otherwise obfuscate the prediction task. In [164], attention originated from video-specific spatial features (such as person detections in third-person videos and objects with motion in egocentric videos), which were calculated intermittently over the course of a video. These temporally examined spatial features introduced past information to an attention mechanism that effectively combined them with the present and selected the most relevant features to represent the ongoing action. In [57, 85, 94, 129, 176], gaze supervision was explicitly required to construct attention maps to weigh the last layer's features before classification.

Self-attention approaches do not require additional data but learn the spatial or temporal importance of input video frames with carefully designed attention layers [88, 149, 150, 170] or by dynamically weighing the usefulness of input modalities

[39]. Spatial attention was considered in [150] where the important regions from every frame were given as input to an LSTM for action recognition, whereas in [149] spatial attention was further correlated sequentially through continuous frames. In [39], the attention mechanism evaluated the importance of the input modalities to select optimal features for action anticipation and recognition. In contrast to explicit attention mechanisms, in Chapters 5 and 6 we are using the additional supervisory tasks to enhance the representation in deeper layers by incorporating information from all tasks and learning them together, thus inciting the network to acquire universally useful features.

2.2.4 Hands and Objects for Egocentric Actions

The explicit exploration of hands and objects and their temporal associations has seen a significant volume of work in the egocentric action recognition domain. Initially, hand detection, segmentation, and identification techniques [8, 10, 84] were developed as a preprocessing step for modeling actions or activities. One of the first works that modeled objects in association with hands and the interactions between them to infer activities and their associations was [33].

Egocentric hand-based activity recognition was the objective in [3]. Initially, an egocentric hand detection and segmentation pipeline was established. The hand detector consisted of a region proposal pipeline and a CNN for classification into hands. The correct hand instances were turned into pixel-level segmentations, which in turn were the input of a second CNN classifier that inferred the performed activity. They showed early on the difference between relying on hand detection or segmentation and using manual labels for activity classification.

Egocentric hand and object trajectories were considered for classification in [38, 42, 103]. In [38], computed trajectories of detected objects were classified as active or passive based on hand manipulations towards them. In [42] the fingertip positions were used as a means to identify human gestures with bidirectional LSTMs. The sequences of fingertip coordinates were classified into a predefined set of gestures. In contrast, in Chapter 4 we are interested in the whole hand and arm regions and do not rely on a predefined set of trajectories but utilize tracking to produce them. Egocentric hand-based activity recognition was considered in [103], where the distance between detected hands or the distance between detected hands and objects marked as active were the features for activity classification. The detection and tracking methods are related to our scope in Chapter 4; however, we focus on the particular objective of recognizing hand-based actions with the help of objects.

In [7], a two-stream visual segmentation-based architecture was used to predict the interaction areas between hands and objects in a video stream and model them as actions. The concept of hand-object interactions was further explored in [16] with the detection of grasps in relation to the shape of objects for modeling actions. We also match the intuition that hands and objects are fundamental for egocentric actions, but we rely on the explicit detection of hand and object regions and their positions to recognize actions. In [4, 96] object and hand localization and segmentation were intermediate learning steps that forced the network to focus on important egocentric cues prior to action prediction. Their understanding that hands and objects are fundamental for actions is similar to ours in Chapter 4, but our pipeline is different in that we model the temporal associations of the tracked hands and objects in the video instead of creating structured feature representations of the raw image pixels with CNNs or other descriptors.

Activity recognition solely based on the detected objects in the scene was considered in [147, 165]. In [165], object detection relied on video input to identify RFID tags manually placed in the scene. In [147] a dynamic, feature prioritization policy was developed to choose which single-class object detectors to promote. They processed as few of them as possible, thus saving computations, while also maximizing the classification accuracy on the subsequent frame. The aim was to take advantage of the spatio-temporal correlations that occur during an activity and avoid extraction of features that would provide unnecessary information to the recognition process. Our method in Chapter 3 does not focus on single objects and their possible associations through time during detection, but takes advantage of single-frame, multi-class object detection to extract objects in real-time and uses LSTM to learn the temporal associations.

2.2.5 Multitask Learning

Multitask Learning (MTL) is a machine learning technique for training models that produce multiple outputs. The main benefit of MTL is the ability to reduce the training requirements (number of models with respect to outputs, training time) by reusing the parameters of a single model. This forces the network to learn features that generalize not only across data, but also across tasks, potentially improving performance.

Training a network for multiple tasks jointly has been shown to improve the performance on all of them or at least the main task, as long as they share a conceptual similarity [19] or are not competing [127]. Caruana [19] was one of the first to show

the benefits of multitask learning by assigning multiple tasks to be solved jointly by a single model. Among others, he demonstrated the use of MTL for classification of medical indexes in pneumonia cases. He showed that prediction of mortality would be improved when additional outputs such as the hematocrit and the white blood cell count of the patients were jointly requested from a model, when compared to predicting the mortality rate alone. Similarly, in vision tasks, *e.g.*, human action recognition, the action can be viewed as the combination of the foreground objects in the scene, the background, and the movements of the person with respect to their immediate environment. Explicitly requesting these outputs, affects the action classification task by forcing the model to consider features it acquired for them in the underlying shared representation. Otherwise, it might have missed task-specific fine-grained features beneficial to the action classification task.

Recently, this concept has found successful application in image and video understanding tasks [48, 64, 76, 91, 95, 97, 99, 107, 127, 145, 173]. Misra *et al.* [99], investigated the number of task-specific layers that should stem from the backbone network to find the optimal setup to train task dualities, pairing segmentation with surface normal prediction and object detection with attribute classification. In [107], video captioning with action prediction and action performance quality were combined as separate task outputs. In [64], a training scheme for joint object and action detection was proposed where action labels were predicted for each detected object in addition to the object class. In [48], the tasks were object detection and segmentation. Furthermore, in [76] the combined tasks were boundary, surface normal, and saliency estimation together with object segmentation and detection. In [95], human pose was used prior to action recognition in an intermediate, secondary task, where appearance features and pose predictions were the combined inputs of the final action classification layer. Zamir *et al.* [173], modeled the relationship between tasks in a latent space to transfer knowledge between them and reduce the number of required training samples. In [127], the problem of loss function weighing was analyzed, to train for multiple tasks efficiently through an optimization scheme that searched for an optimal solution of network parameters to balance tasks. In [145], network parameters were randomly selected to be either task-specific or shared across tasks introducing parameter specialization during training. In [91], an attention mechanism was applied to weigh each layer's activations according to the specified task and in [97] task-specific attention was modulated at the channel level.

MTL in egocentric vision appears in [4, 57, 85, 96, 101, 149, 168]. Yan *et al.* [168] considered the activities performed by each individual participant as separate tasks where the objective was to cluster common activities among participants without

supervision. In [4], from an RGB video input, multiple network branches learned activities, object proposals, and segmentations, with large parts of the network trained independently. In [96], an object and an action learning task were combined to produce an activity prediction (as a combination of the two), with the network consisting of two separate streams which contributed to the final prediction layer and shared their parameters. In [85], the network learned a gaze map that was used to pool from the activations of the final feature map for actions. Similarly, in [57], a prior gaze estimation was used to influence the action prediction, which in turn affected the final gaze prediction. Training took place jointly, but internally, each network part was deployed for a specific task, without parameter sharing. Another example of multitask learning in egocentric vision comes from [101] with joint learning of activities and energy expenditure from video. The input to the network was multimodal (video and accelerometer signals) and each stream was trained individually with parameter sharing only during a late fusion stage. Lastly, in [149] the action task was constructed with the addition of complementary verb and noun learning tasks to bias the action classifier. In our work, in Chapters 5 and 6, all task outputs are parallel and do not affect each other, besides sharing the backbone network.

2.2.6 Multi-dataset Training

Multi-dataset or multi-domain learning is related to transfer learning in the sense that we wish to utilize data from numerous sources in order to optimize the learning process. Usually, this is unfeasible due to the lack of universally compatible annotations that capture all tasks across datasets [26]. Thus, multi-dataset training refers to the combination of diverse data sources concurrently during training to jointly optimize the gradients of a multitask loss from the tasks of all datasets [62, 76]. Kokkinos [76] proposed UberNet to tackle the tasks of boundary, semantic boundary, and saliency estimation, surface normal prediction, segmentation, and object detection in a single network. The lack of a dataset with annotations for all tasks led to a gradient accumulation update rule that only updated gradients for a task when enough samples had been seen for it. However, it risked memory constraints from maintaining task-specific gradients until the threshold was met. Additionally, the gradient updates for the main block may not be representative of all the tasks in each training step, affecting the statistics in the batch normalization layers [58]. To alleviate this issue, [109] proposed training on interleaved mini-batches per dataset and to use group normalization [166] to facilitate network convergence. The main

difference in our approach in Chapter 6 is that we create mixed batches that enable the network to grasp information across datasets on every training iteration.

Chong *et al.* [21] modeled human attention, with separate output layers for gaze and saliency estimation. Each layer branched off from a single backbone that was trained with mixed batches. There were as many backpropagation steps per batch as the number of available output layers, which would negatively affect training of the backbone as in [76]. Guo *et al.* [46] proposed several approaches to combine datasets for human pose estimation including the unification of datasets towards a single prediction task, transfer learning between datasets in a sequence, and a multitask scheme to jointly supervise each dataset's output poses. Of the latter, outputs were eventually combined with a voting mechanism. This approach used fully compatible datasets, from a task perspective, making task fusion feasible. We also investigate mapping related tasks across datasets in Chapter 6.

A more related approach to ours [108] considered concatenating output layers for cross-dataset classification, but without leveraging the possible class similarities throughout tasks. Alternatively, [22] performed inter-dataset experiments on the EPIC-Kitchens and EGTEA Gaze+ datasets, but only on the subset of common classes. Our approach is different in that we construct a single model that fully encapsulates both datasets. Lastly, [52] considered explicit task outputs for face attribute classification, with mixed batches across datasets and masked losses, while attempting to diversify the learned manifold by adding a domain adaptation output to discriminate the datasets during training.

2.3 Egocentric Video Datasets

Scientific literature provides a plethora of egocentric video datasets [34, 35, 40, 41, 82, 101, 111]. The datasets of [40, 41] were created with the aim of detecting locations and observing indoor and outdoor everyday scenes. The dataset of [111] focused on activities that take place either indoors or outdoors, such as walking, running and sitting, whereas [101] was enhanced with accelerometer and heart rate data to infer the level of sedentariness in the performed activities. The dataset of [82] included annotations and segmentations of the important objects that characterize activities, with a large number of videos captured outdoors. Datasets from [34, 35, 85] consist of videos in a kitchen, in which the participants were asked to prepare food according to predefined recipes. We summarize these datasets in Table 2.1. In the remainder of this section, we provide a description of the egocentric video

datasets most commonly used in the thesis: ADL [110], EPIC-Kitchens [24], and EGTEA Gaze+ [85].

Tab. 2.1.: Egocentric Video Datasets (P: Participants, Acc: Accelerometer, HR: Heart Rate, G: Gaze, HM: Hand Masks).

Dataset	Modalities	#P	Mount	Annotations	Classes	Location
Furnari *et al.* [41]	Video	1	Head	Personal Locations	11	In/Outdoors
Furnari *et al.* [40]	Video	1	Head	Personal Locations	9	In/Outdoors
Stanford ECM [101]	Video/Acc/HR	10	Chest	Activities	24	In/Outdoors
UT-EGO [82]	Video	4	Head	Summarization	-	In/Outdoors
ADL [110]	Video	20	Chest	Activities	32	Indoors
Poleg *et al.* [111]	Video	13	Head	Activities/G	7	Outdoors
GTEA [35]	Video	4	Head	Actions/HM	71	Kitchens
GTEA Gaze [34]	Video/G	14	Head	Actions	94	Kitchens
GTEA Gaze+ [34]	Video/G/Audio	10	Head	Activities/Actions	7/25	Kitchens
EGTEA Gaze+ [85]	Video/G	32	Head	Actions/HM	106	Kitchens
EPIC-Kitchens [24]	Video/Audio	32	Head	Actions/Objects	2513/352	Kitchens

2.3.1 Activities of Daily Living

In Chapters 3 and 6 we make use of the Activities of Daily Living (ADL) dataset [110]. It consists of 20 videos captured from a first-person perspective of people performing activities occurring indoors. The camera was mounted on the participants' chests. Each video is a record of the subject's choice of activities from a predefined set, performed in an unscripted manner. In every video, the person is different and operates in their own house, thus providing considerable variations in locations and activities throughout the dataset. In total, there are approximately 10 hours of egocentric videos, equivalent to more than one million frames. The videos are annotated with 32 activity labels with start/end times, 48 object classes with bounding box instances, object tracks and human-object interactions. Example frames from the ADL dataset are shown in Fig. 2.2.

2.3.2 EPIC-Kitchens

The EPIC-Kitchens dataset [24] comprises a set of 432 egocentric videos recorded by 32 participants in their kitchens at 60fps with a front-facing head-mounted camera. There is no guiding script for the participants who freely perform activities in kitchens related to cooking, food preparation, or washing up. Example frames are showcased in Fig. 2.3. Each video is split into short action segments (mean duration is 3.7s) with specific start and end times and a verb and noun annotation describing the action, (*e.g.*, 'open fridge'). The verb classes are 125 and the noun classes 352,

Fig. 2.2.: Example frames (©2012 IEEE) from the ADL dataset. Notice the position of the objects and the hands with respect to the chest view of the scene.

the valid combinations of which are 2,521 actions. The dataset is divided into one train and two test splits. For both test splits, the verb and noun annotations are not openly available, hence we focus our work on the fully annotated train set. It consists of 272 videos by 28 participants (28,470 action segments). We experiment with EPIC-Kitchens in Chapters 4, 5, and 6.

2.3.3 EGTEA Gaze+

EGTEA Gaze+ [85] consists of 86 videos of 32 people in seven scenarios of food preparation activities in kitchens. The videos are cropped into 10,321 clips based on action segment annotations. The dataset comes with three predefined train and test splits, with the first comprising 8,299 and 2,022 clips, respectively. Each clip is labeled from 19 verbs and 53 nouns and their 106 valid action combinations in the dataset. In addition, the dataset is complemented with a gaze annotation for every video frame, which consists of an (x, y) coordinate and its type (fixation, saccade, or unknown). Example frames from the EGTEA Gaze+ dataset are shown in Fig. 2.4. We utilize EGTEA Gaze+ in our experiments in Chapters 5 and 6.

Fig. 2.3.: Frames from EPIC-Kitchens. The images comprise almost a top view with respect to the areas of interest where the actions take place. (Reprinted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature, Scaling Egocentric Vision: The Dataset by Dima Damen, Hazel Doughty, Giovanni Maria Farinella *et al.* ©2018)

Fig. 2.4.: Frames from EGTEA Gaze+. (Reprinted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature, In the Eye of Beholder: Joint Learning of Gaze and Actions in First-Person Video by Yin Li, Miao Liu, James M. Rehg *et al.* ©2018)

3 Object-based Location and Activity Classification

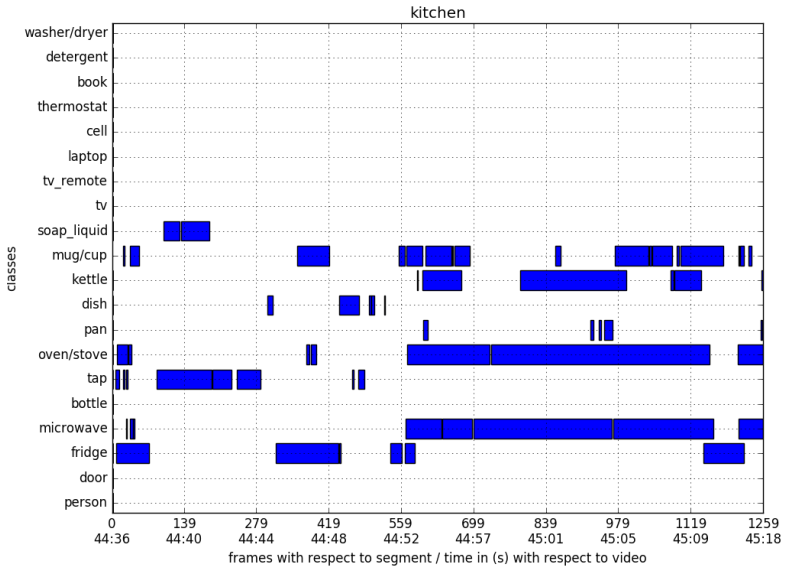
3.1 Introduction

In this chapter, we focus on indoor location and activity detection from egocentric videos, with typical applications in Ambient Assisted Living (AAL) [104]. An example can be non-intrusive status updates to healthcare professionals about the locations and actions of people suffering from limited vision or dementia. Activities of daily living are also of interest in the case of patient rehabilitation after serious illness. Normally, this process would take place in a protected environment, far from the person's home. The possibility of continuous and real-time monitoring offered by egocentric cameras allows for non-invasive and personalized care. Reusability of the equipment by other patients at the end of the recovery period is an additional incentive towards adoption by nursing homes. Moreover, enhancement with intelligent detection mechanisms will promote privacy, since only information relevant to the rehabilitation will need to be communicated to third parties and not the actual video stream. An example use-case is that of dementia patients who require constant monitoring and professional care [71]. Egocentric vision is able to provide the indoor location [70], the duration of physical exercises [154] or any performed activity, upon request or in a continuous mode.

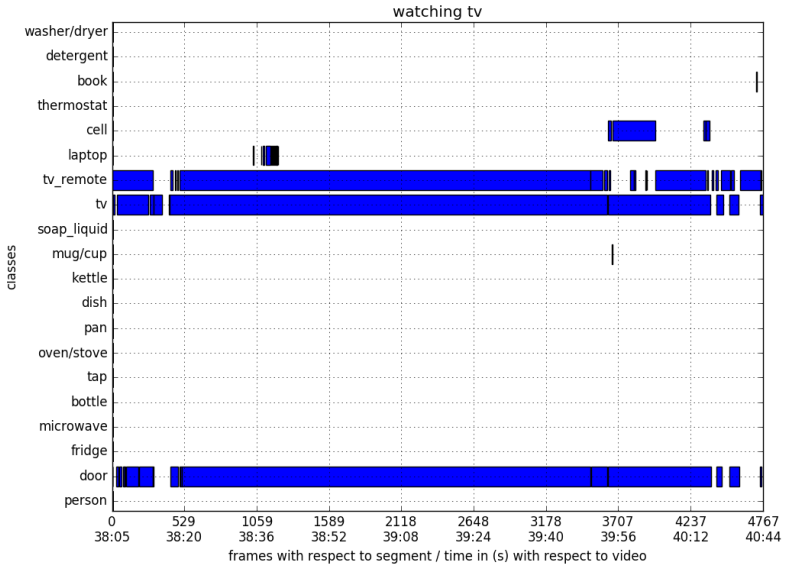
To produce an inference on an image or video frame, one could calculate image-descriptive features [23, 92] stack them in vectors and classify, using machine learning models in a supervised fashion. In recent years, feature extraction and classification tend to merge into end-to-end deep networks, providing promising results. In this chapter, we take a step back and consider a different type of input.

Published as:
G. Kapidis, R. W. Poppe, E. A. van Dam, R. C. Veltkamp, and L. P. J. J. Noldus. "Where Am I? Comparing CNN and LSTM for Location Classification in Egocentric Videos". IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). 2018, pp. 878–883
G. Kapidis, R. Poppe, E. van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. "Object Detection-Based Location and Activity Classification from Egocentric Videos:A Systematic Analysis". Smart Assisted Living. Springer International Publishing, 2020, pp. 119–145

We abstract away the detection of low-level visual features and consider high-level information to be the input to our location and activity classification pipelines.



(a) Location 'kitchen'. The oven and the microwave dominate the scene.



(b) For activity 'watching TV' the television and the remote controller are indicative of the performed activity.

Fig. 3.1.: Detected objects for video segments from the ADL [110] dataset.

Our key idea is to use the detected objects in a video frame as cues to recognize the indoor location or an ongoing activity [60]. Initially, we build on the idea that rooms can be characterized by the presence of specific, distinctive objects. Contrary to [60], we use a small set of objects, closely affiliated with the indoor videos of

ADL [110]. This consistency can be translated into associations between objects and locations. For example, consider Fig. 3.1a which shows the detected objects of an egocentric video segment from a kitchen. If we categorize the objects based on their mobility we may group them into a) those that can be thought of as movable, but bear meaning for understanding the scene, such as the soap, the mug, and the dish and b) those that are unmovable, but (i) distinctive to this particular location, such as the stove, the microwave, or the fridge and (ii) those that can be found in multiple locations, for example the tap, which could also appear in a bathroom. Similarly, we claim that the activity of the egocentric protagonist can be inferred from the detected objects, considering Fig. 3.1b which shows a TV and a remote controller for most of the duration of a video segment for activity 'watching TV'.

These observations motivate us to perform an analysis on the videos of the Activities of Daily Living (ADL) dataset [110] to discover associations between objects and locations and objects and activities. For the object-location associations, we train classifiers with Artificial Neural Networks (ANN) and Long Short-Term Memory networks (LSTM) [51] to experiment with per-frame classification and utilization of the temporal structure of the data, respectively. Conceptually, an individual frame of a scene might include only partial information about the objects, as not all that are detectable may fit in view. Furthermore, there might be missing or false-positive detections on a single frame. However, using LSTM we can encode a more complete view of the room by combining the detections from multiple frames over time. This can potentially enhance the knowledge about objects and alleviate the effects of noisy detections. Eventually, we compare the performance of classifiers from both types of models, trained either on object labels or detections, from detectors trained on object categories from different datasets.

For the object-activity associations we rely on object detections enhanced with information about the appearance of the objects. Apart from the presence of objects (binary), we measure the bounding box sizes and their positions in the frames. We aim to investigate if this additional information modifies the status of an object as participating in the ongoing activity. For example, a pan observed from a distance (smaller) or at the edge of the view would indicate that a kitchen related activity other than frying is performed, such as eating or cleaning. Fig. 3.2 outlines our approaches. The contributions of this chapter can be summarized as:

- the development of a method to analyze object associations towards 1) locations and 2) activities in egocentric videos,
- the binary object presence feature, which despite its simplicity, demonstrates acceptable performance,

- the description of location classification results for diverse object sets and detection thresholds with and without temporal information,
- the demonstration that laborious object annotations are not required for location classification, given that our system performs equally well using only automatic detections, and
- the analysis of object-activity associations in the context of daily living and the effect that object sizes and positions have to the activity recognition results.

In Sec. 3.2 we describe the object detection pipeline and the methodology for the location classification and activity recognition tasks. In Sec. 3.3 we present our results and we provide a discussion in Sec. 3.4.

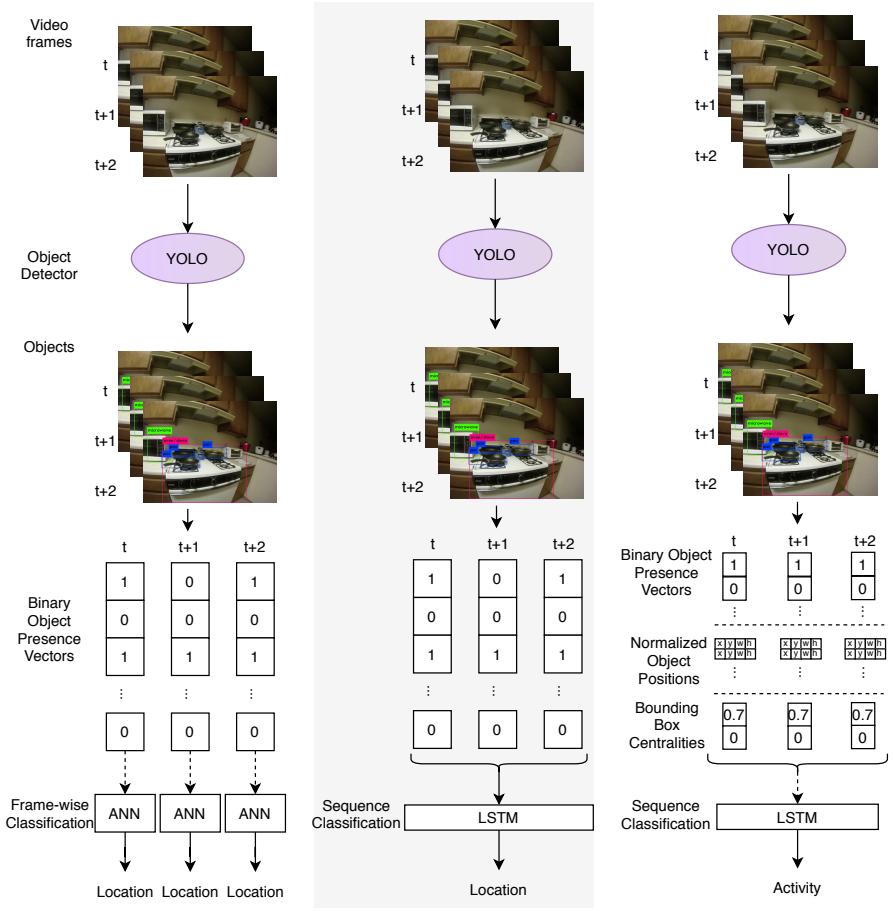


Fig. 3.2.: Three pipelines for location and activity recognition. We extract objects from images using an object detector (YOLO [119]). The detections for a frame are turned into binary object presence vectors (BPV). This representation is the input to ANNs (left) which produce one location per frame. BPV sequences are the input to an LSTM (middle) which generates one location for the entire sequence. For activity classification (right) we consider two additional features that describe the bounding box sizes and positions and classify them with an LSTM.

3.2 Methodology

In this section, we analyze in depth the Activities of Daily Living (ADL) egocentric video dataset in terms of objects, locations, and activities (Sec. 3.2.1). We also select the object detection framework to perform our tests on (Sec. 3.2.2). Finally, we study the parameters of the location (Sec. 3.2.3) and activity (Sec. 3.2.4) classification tasks.

3.2.1 Activities of Daily Living Dataset

A description of the ADL dataset [110] is given in Sec. 2.3.1. The dataset contains annotations for 48 object classes. For the object detectors we either use the whole set of 48 classes or a subset of 20. We elaborate on object detection in Sec. 3.2.2. A list of the object classes together with their occurrences in the ADL dataset appears in Table 3.1. Train and test splits are provided by the authors; videos 1 through 6 are considered training data and the remaining 14 comprise the test set. For our experiments we use the same splits.

Tab. 3.1.: The 48 object classes of the ADL dataset and the number of occurrences per class. In the third column the instances in the train set. In bold, the classes of the ADL20 subset.

Class	Total	Train	**Class**	Total	Train	**Class**	Total	Train
person	4650	2424	food/snack	3876	741	shoes	3248	735
door	7903	2019	**kettle**	1239	464	tea bag	359	177
fridge	1999	301	**mug/cup**	11050	2766	**laptop**	7027	2183
microwave	2369	527	**soap liquid**	8375	2658	cell phone	653	271
bottle	10310	1705	pills	394	148	**cell**	571	238
tap	7826	3252	basket	1588	35	**thermostat**	332	137
oven/stove	3196	1007	towel	4480	1961	**book**	4770	445
pan	3156	1026	tooth brush	1795	819	dental floss	547	385
trash can	2075	486	tooth paste	1746	492	vacuum	519	116
dish	8216	2274	electric keys	1570	417	electric keys 2	118	118
cloth	3077	78	**TV**	5600	2033	pitcher	1208	277
knife/spoon/fork	4843	1893	**remote**	2813	1253	**detergent**	1105	297
bed	783	228	container	5685	3821	**washer/dryer**	3362	954
large container	558	6	shoe	694	300	milk/juice	366	0
monitor	316	287	blanket	85	31	mop	403	0
keyboard	107	102	comb	307	51	perfume	550	0

In Sec. 3.2.3 we are interested in the analysis of locations and we extend the dataset with location annotations from [159]. For every 30 video frames, one out of eight possible locations are annotated, namely, kitchen, bedroom, bathroom, living room, laundry room, corridor, outdoor, and undefined (Table 3.2). Class 'undefined' occurs in blurred frames or non-identifiable locations. We do not use these frames for

training or testing the location classification models. Hence, the location classes are seven in our experiments.

Tab. 3.2.: Sampled frames per location. Class 'undefined' is not used for training and testing.

Location	Kitchen	Bedroom	Bathroom	Living room	Laundry room	Corridor	Outdoor	Undefined
Train	3414	1821	2307	2606	815	45	143	492
Test	6850	3966	2285	5045	1097	133	906	737
Total	10264	2285	4592	7651	1912	178	1049	1229

In Sec. 3.2.4 we focus on activity classification and make use of the existing activity annotations in the ADL dataset. We transform the labels from describing video segments with specific start and end times, to one activity per frame. The activities are shown in Table 3.3. In [110], only 18 activities are considered due to scarcity of samples, whereas the dataset contains labels for 32. We consider all 32 activities plus a background class in our experiments.

Tab. 3.3.: The 32 activity classes in the ADL dataset, plus the background class (35906, 85801). In parentheses the number of train and test frames per class, respectively.

1:combing hair (3539, 6267)	2:wearing make up (8363, 3926)	3:brushing teeth (22729, 26117)	4:using dental floss (8543, 2127)
5:washing hands/face (15050, 17270)	6:drying hands/face (4014, 6743)	7:entering/leaving room (0,0)	8:adjusting thermostat (1110, 2459)
9:doing laundry (28812, 46101)	10:washing dishes (21249, 45807)	11:moving dishes (9984, 0)	12:making tea (15679, 27265)
13:making coffee (6774, 18974)	14:drinking water/bottle (6565, 12328)	15:drinking water/tap (0, 540)	16:making hot food (8872, 38619)
17:making cold food/snack (14268, 11546)	18:eating food/snack (6686, 32180)	19:mopping in kitchen (1020, 8933)	20:vacuuming (3657, 9864)
21:taking pills (3237, 4409)	22:watching TV (37769, 78086)	23:using computer (20445, 57125)	24:using cell (5817, 10435)
25:making bed (0, 6055)	26:cleaning house (11360, 12655)	27:reading book (20350, 18016)	28:using mouth wash (420, 570)
29:writing (0, 3628)	30:putting on shoes (5668, 450)	31:drinking coffee/tea (15226, 33778)	32:grabbing tap water (599, 1170)

3.2.2 Object Detection

We use Darknet[1] for object detection. The detector is YOLOv2 (short for You Only Look Once) [118, 119]. It is a real-time object detection system that operates on input images of various sizes. YOLOv2 is based on the Darknet-19 architecture [119] and consists of 19 convolutional and 5 max-pooling layers. It is pretrained on ImageNet [28] for 1,000 classes, for 160 epochs. From this pretrained model, we develop three separate detectors, one for every object dataset we consider.

[1] https://pjreddie.com/darknet/

Our first YOLOv2-based detector is finetuned on the 80 classes of the MS COCO dataset [89] and the weights are provided by the authors of [119]. We call this detector 'COCO' for short. We train two additional models with this architecture for the object classes of the ADL dataset: (1) 'ADL48', on all the classes in Table 3.1 and (2) 'ADL20', on the 20 in bold. The selection of classes for 'ADL20' follows [159], where they select only classes for which their detector achieves more than 5% Average Precision (AP).

The reason for the diversification of detectors is that MS COCO and ADL consist of different sets of classes. ADL comprises objects found in homes (Table 3.1), whereas MS COCO is more generic in its categories (Table 3.4). The split between 'ADL20' and 'ADL48' is an attempt to produce a detector focused on classes with more samples in the training dataset, thus excluding harder to detect classes. We expect this to improve performance of the bounding box classifier for the ADL20 subset due to the increased available capacity of the network.

For both 'ADL20' and 'ADL48' we finetune the ImageNet weights for 35k iterations, (*i.e.*, batches). During training, the input dimensions of the detectors change together with the image sizes. This allows the network to learn the object features in various sizes. The training hyperparameters are the same as in [119]. The 'ADL20' detector achieves 29.84% mAP (mean Average Precision) and the 'ADL48' 11.15%. In Table 3.5, we report the average precision per class for our detectors. In total, the mAP for 'ADL20' is 29.80% and for [159] is 23.35%. Furthermore, the YOLOv2-based 'ADL20' is a more successful detector for the majority of the considered object classes of the ADL dataset than Fast R-CNN [44] from [159].

Tab. 3.4.: The 80 object classes of MS COCO [89].

person	bicycle	car	motorcycle	airplane	bus
train	truck	boat	traffic light	fire hydrant	stop sign
parking meter	bench	bird	cat	dog	horse
sheep	cow	elephant	bear	zebra	giraffe
backpack	umbrella	handbag	tie	suitcase	frisbee
skis	snowboard	sports ball	kite	baseball bat	baseball glove
skateboard	surfboard	tennis racket	bottle	wine-glass	cup
fork	knife	spoon	bowl	banana	apple
sandwich	orange	broccoli	carrot	hot dog	pizza
doughnut	cake	chair	sofa	potted plant	bed
dinning table	toilet	TV	laptop	mouse	remote
keyboard	cell phone	microwave	oven	toaster	sink
refrigerator	book	clock	vase	scissors	teddy bear
hair dryer	toothbrush				

Tab. 3.5.: Per-class average precision (%) of 'ADL20' and 'ADL48' object detectors, trained with YOLOv2. Certain classes are particularly challenging. In bold, the classes that improve in the ADL20 subset. Comparison with [159] using Fast R-CNN for the 20-class subset of the ADL dataset. With italics the object classes that are better in [159].

Objects [110]	ADL20	ADL48	[159]	Objects (ctnd.)	ADL20	ADL48	[159]
person	**69.00**	59.49	25.74	container	-	5.25	
door	**23.20**	17.72	5.59	shoes	-	0.72	
fridge	**22.75**	12.85	24.95	tea bag	-	0.68	
microwave	**37.80**	24.81	32.35	laptop	**44.40**	41.04	37.46
bottle	**10.02**	4.59	*11.28*	cell phone	-	10.91	8.65
tap	**59.27**	51.18	39.55	cell	**0.89**	0.65	
oven/stove	**44.48**	28.15	43.02	thermostat	**24.88**	3.89	9.01
pan	**16.88**	12.46	10.99	book	16.39	18.04	12.83
trash can	-	9.61		dental floss	-	0.92	
dish	**14.21**	6.22	11.19	vacuum	-	0.66	
cloth	-	4.55		elec keys	-	0.00	
knife/spoon/fork	-	4.80		pitcher	-	3.13	
food/snack	-	9.65		detergent	**9.90**	9.76	9.13
kettle	**22.54**	8.68	*23.83*	washer/dryer	**37.96**	25.58	*38.86*
mug/cup	**15.92**	15.69	13.24	bed	-	0.21	
soap liquid	21.76	31.59	18.77	large container	-	0.00	
pills	-	0.14		monitor	-	0.00	
basket	-	0.00		keyboard	-	0.00	
towel	-	9.38		shoe	-	0.15	
tooth brush	-	9.78		blanket	-	0.00	
tooth paste	-	11.67		comb	-	0.10	
electric keys	-	0.73		perfume	-	0.00	
TV	**52.07**	49.57	*57.58*	milk/juice	-	0.00	
remote	**52.49**	30.36	32.88	mop	-	0.00	

3.2.3 Locations

We model the relationship between the objects in a frame or in a series of frames to recognize the location. Applying object detection on the videos of the ADL dataset leads to a binary presence vector (BPV) of zeros and ones for every video frame, with length equal to the number of output classes of a detector, *i.e.*, 80 for 'COCO', 48 for 'ADL48' and 20 for 'ADL20'. In BPV we only consider whether an object is present in a scene or not, regardless of the times it is found. We also experimented with keeping the counts of multiple detections of the same object in a frame using a multiple presence vector (MPV), but without consistent improvements. Location labels exist once every 30 frames (1 second) [159] and only these frames are used for classification, without augmentation for the ones in between.

We train two types of classifiers. The first is a fully-connected neural network (Artificial Neural Network, ANN) with input one BPV per used frame. The second is

a Long Short-Term Memory (LSTM) network which is used to examine the temporal structure of the data by being trained on BPV sequences. For both ANN and LSTM classifiers we parametrize our experiments with respect to the object datasets. They are categorized based on:

- the **dataset combinations** for training and evaluating the location classifier,
- the **object detector classes**, and
- the **object detection thresholds**.

We categorize as such after considering our objective, *i.e.*, to assess whether an object detector can be used as the first step in an indoor location recognition pipeline. In this context, we experiment with using either object annotations or detections to model the locations. At test time, we compare against object detections in order to examine the modeling capabilities of either train set. Hence, the dataset combinations comprise the scenarios that affect the composition of a location classifier's dataset. The first combination is 'Labels to Detections' (L2D) where we use the object annotations for training and the detections for testing. The second is 'Detections to Detections' (D2D) which contains only detections for both train and test splits. We also consider 'Labels to Labels' (L2L) which consists of the object annotations for both splits, *i.e.*, the object detections are *not* used. The latter is used for comparison purposes to measure the possible performance drops from using only detections, which can be expected to contain noise.

The object detector class variations were discussed in Sec. 3.2.2. Using this as a parameter means that we vary the object detector that produces the object dataset. As a result, different object classes are learned. In turn, this leads to generating object vectors (BPV) of different lengths. The available object classes appear to affect performance on certain object-depended locations as shown in Sec. 3.3.1.2.

Finally, the detection acceptance threshold creates a trade-off between the confidence and the number of detections. Higher thresholds lead to fewer but more accurate detections. Lower thresholds provide more objects, but with higher numbers of false-positives. In the D2D experiments we always use detections with the same acceptance threshold for both training and testing.

3.2.4 Activities

For activity detection we also rely on object detections from Darknet. We use the set of 20 object classes of 'ADL20' as described in Sec. 3.2.1 (Table 3.1), enhanced with

object-related information. Specifically, for every video frame, we extract objects along with their size and position in the frame. As features, we consider the BPVs as described in Sec. 3.2.3, along with the bounding box positions and the centrality.

The **bounding box position** (BB) constitutes a 4-vector per object containing the (x, y, width, height) parameters that characterize a bounding box. The values are normalized to the width and height of the frame to fall into the [0-1] range. For the 20 object classes the BB feature has length 80, *i.e.*, four values per object class. The **centrality feature** (CF) signifies that a larger object area or a bounding box which is closer to the center of the image is more important for the detection of an activity. It constitutes a 2D Gaussian ($\mu = 0.5$, $\sigma = 0.1$) (in terms of normalized image coordinates) to produce a weight distribution that focuses its importance on bounding boxes found closer to the center of the frame. As a result, bigger boxes gain importance because they aggregate values over a larger area. Our intuition is that significant objects for human activities will be detected near the center of the scene or due to their size, they will draw attention to themselves [7]. An illustration of the centrality feature's estimation is provided in Fig. 3.3.

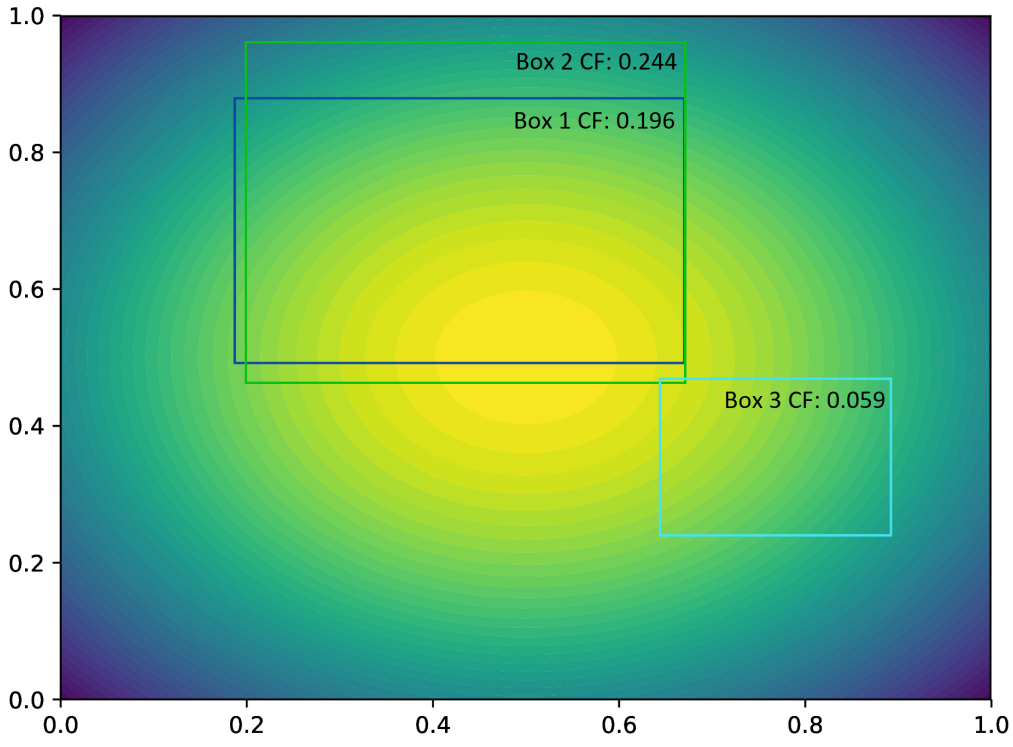


Fig. 3.3.: Estimation of the centrality feature for three boxes. Box 1: 0.196, Box 2: 0.244, Box 3: 0.059. As boxes become smaller or move away from the center the value of the centrality feature decreases.

3.3 Experiments

In this section, we delineate our experiments. Location classification is presented in Sec. 3.3.1 and activity recognition in Sec. 3.3.2.

3.3.1 Location Classification

We divide the experiments for location classification into ANN- and LSTM-based architectures in Sec. 3.3.1.1 and 3.3.1.3 respectively. In Sec. 3.3.1.2 we perform a per-class examination for certain ANN cases.

3.3.1.1. ANN Classification

Our Artificial Neural Network models consist of five fully connected layers, with Rectified Linear Unit (ReLU) activations for the neurons of the input and the three hidden layers that follow. The neurons per layer are 64, 256, 128, 64 and 7 (for the output), respectively. We do not apply dropout, following preliminary tests where we experience slightly worse performance. We use categorical cross entropy to calculate the loss and Stochastic Gradient Descent (SGD) for optimization. All models are trained for 150 epochs. We set the starting learning rate at 10^{-2} and divide by 10 every 50 epochs. The batch size is set to 64.

We implement experiments for the L2L, L2D and D2D cases of Sec. 3.2.3 with detection confidence threshold in the L2D and D2D cases ranging from 30% to 70%. The object sets vary between ADL20, ADL48, and MS COCO, with the latter only supporting the D2D case due to the lack of annotations for its object classes in the ADL dataset. The classifiers for each object set only differ on the input feature size which ranges between 20, 48 and 80 respectively. The models' trainable parameters with respect to the input feature size are 59,591, 61,383, and 63,431. In Tables 3.6 and 3.7 we present results in terms of overall Top1 accuracy and averaged F1-score over the seven locations in the test set, respectively.

When considering the *dataset combinations*, the highest classification accuracy is found in the L2L scenarios. They outperform the variants that depend on object detectors. This is expected since the object annotations do not contain detector-induced noise, so the train and test sets are clean with no objects out of place. When detectors are used, the D2D classifiers tend to outperform the L2D for the same object sets, even though they are trained on noisier samples. This observation

provides insights about the way the ANN classifier handles noise. It may be confused by unexpected objects at test time, but if the noisy detections are part of the training set it will deal with them more successfully at test time.

Tab. 3.6.: ANN Top1 accuracy (%) for location classification – averages of the best five models of each experiment. Comparisons between L2L, L2D, and D2D for the various detector cases and detection thresholds. L2L outperforms the variants that depend on object detectors. Decreasing the object detection threshold improves location classification accuracy for every object set.

L2L		Conf. (%)	L2D		D2D		
ADL20	ADL48		ADL20	ADL48	ADL20	ADL48	COCO
		30	59.684	54.802	62.951	56.469	64.356
		40	58.113	48.157	60.744	55.860	62.876
77.700	77.045	50	56.645	47.667	58.734	55.111	60.839
		60	55.001	39.368	55.766	52.056	57.801
		70	47.611	38.953	51.654	48.662	51.717

Tab. 3.7.: ANN F1-scores averaged over the seven location classes for the best performing model in Top1 accuracy. Comparison between L2L, L2D, and D2D for the various detector cases and detection thresholds. Certain locations (corridor, outdoor) are almost undetectable, negatively affecting the average score.

L2L		Conf. (%)	L2D		D2D		
ADL20	ADL48		ADL20	ADL48	ADL20	ADL48	COCO
		30	45.982	41.562	47.441	41.338	43.167
		40	42.633	39.211	45.181	40.247	39.484
58.474	57.738	50	42.875	37.010	42.608	38.696	36.785
		60	39.210	29.674	39.043	34.866	34.758
		70	34.151	22.252	34.261	31.882	30.261

Varying the *object detector* affects the classification results significantly. In Tables 3.6 and 3.7, ADL20 L2D and D2D outperform their ADL48 L2D and D2D counterparts and COCO D2D performs better than both. When comparing ADL20 L2D with ADL48 L2D it is important to consider the detection datasets. The test set of ADL20 consists of 67,906 ground truth boxes and that of ADL48 of 95,845 (TP+FN in Table 3.8). The additional 28k boxes of ADL48 belong to the harder to detect classes that are discarded from ADL20. The low average precision for these classes (Table 3.5) indicates that most of their instances are not detected. This suggests a harder task for 'ADL48' to produce the "Detections" datasets for any confidence threshold. For example, at 50% confidence it has less TP but more FP and FN (Table 3.8). These can be interpreted as increased noise (FP) and reduced detection quality (FN) when compared to 'ADL20'.

In terms of location classification accuracy, ADL20 L2D 50% is almost 9% better from ADL48 L2D 50% and ADL20 D2D 50% is 3.6% better from ADL48 D2D 50%.

Finally, 'COCO', due to the higher number of training samples for each object class (over 5k [89]) and despite consisting of 80 classes, seems more robust in its object detections and the resulting location classifiers perform better. Interestingly, in the L2L case the 'ADL48' variant is on par with 'ADL20', meaning that the additional object classes, when not burdened by noise, do not harm location classification. The fact that 'COCO' outperforms all other detector-based location classifiers adds to this, showing that quality detections without as many false-positives (resembling L2L as much as possible) even for classes from a more general context, are useful. Notably, MS COCO contains a number of classes that are irrelevant to indoor activities of daily living, such as elephant and airplane (Table 3.4). However, in spite of any false-positive detections of these, the detection-based locations classifiers based on 'COCO' are the best performing ones.

Finally, we vary the *detection threshold* from 30% to 70% with a step of 10%. Our results suggest that as it increases, location classification performance drops. Lower thresholds lead to more available true-positive object detections. This allows the location classifiers to identify uncertain locations easier, showing that they are resistant to noise. On the other hand, higher thresholds result in fewer detections with higher confidence on average and fewer false-positives, but they are not as adequate for inferring the location. All in all, decreasing the object detection threshold improves location classification accuracy for every object set. The significant variance in the number and quality of detections as a result of modifying the confidence threshold for 'ADL20' and 'ADL48' is shown in Table 3.8 where we report the object detection results on the ADL test videos.

Table 3.7 shows the average F1-score of the seven location classes for the best performing model in Top1 accuracy. The drop in performance when compared to the Top1 accuracy is attributed to the difficulty in detecting locations corridor and outdoor. These locations usually lack characterizing objects, *e.g.*, corridors usually consist of poorly illuminated walls. Finally, corridor and outdoor are the most sparsely annotated classes (Table 3.2), which naturally impairs performance.

3.3.1.2. Examination Per Class

In Fig. 3.4 we compare the per-class F1-scores for selected ANN classifiers to examine easier and harder to detect locations. No classifier is universally better, but superiority of certain classifiers can be observed for individual locations.

ADL20 outperforms ADL48 for all locations in both the L2D and the D2D cases. Similarly, the D2D cases outperform their L2D counterparts per class in most situations.

Tab. 3.8.: ADL20/48 object detector results. True-positives decrease along with the false-positives as the confidence threshold increases, complicating the classification task.

Detector	ADL20			ADL48		
Thresh. (%)	TP	FP	FN	TP	FP	FN
30	14,777	12,025	53,129	12,619	24,166	83,226
40	13,277	8,231	54,629	10,509	13,800	85,336
50	11,762	5,744	56,144	8,493	8,051	87,352
60	9,951	3,784	57,955	6,532	4,558	89,313
70	7,621	2,262	60,285	4,417	2,209	91,428

COCO D2D performs best for 'kitchen', 'bedroom', and 'living room' due to its ability to detect additional objects such as 'fork', 'sofa', 'chair', and 'bed'. However, it underperforms for 'laundry room' because it misses a location specific object class related to the 'washer/dryer' from ADL20/48. Locations 'outdoor' and 'corridor' generally suffer due to the scarcity of training samples and relevant objects (Table 3.2).

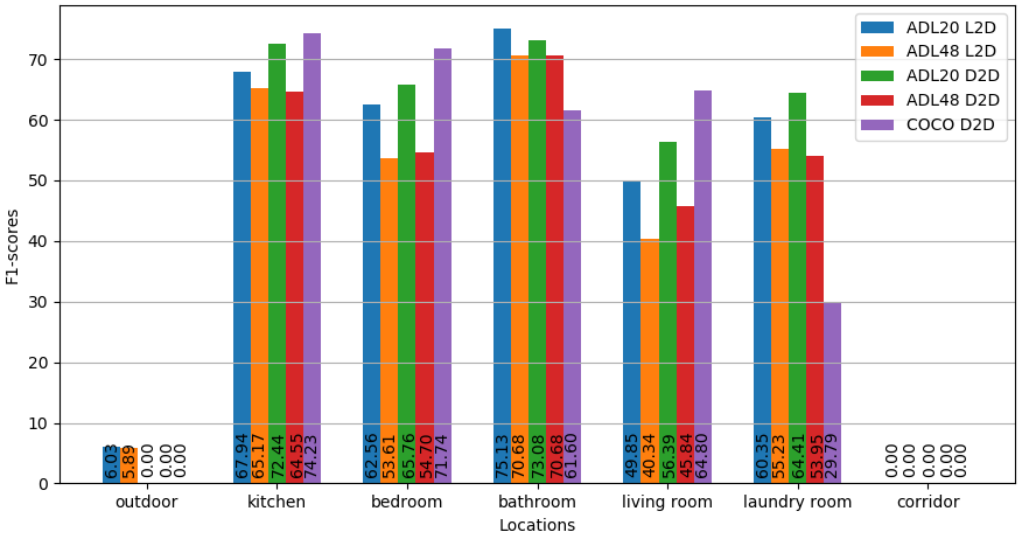


Fig. 3.4.: The per class F1-scores at detection threshold 0.3 for five ANN classifiers. (Better seen in color - the order of names in the legend identifies their order in the graph.)

3.3.1.3. LSTM Classification

In this section, we are interested in studying the objects in video segments instead of single frames. In Fig. 3.1a the 'kitchen' scene lasts for 1,260 consecutive frames (42 seconds) and the detected objects are not consistent throughout the segment. In certain views of the scene, the output of object detection are Binary Presence Vectors (BPVs) that cannot be associated with the ongoing location, for instance when no

objects have been detected. Classifying one such BPV with an ANN classifier, (*e.g.*, ADL20 D2D 30%) produces the mistaken prediction 'laundry room', in between correct predictions of 'kitchen' for surrounding frames that have sufficient detections. These frames include objects such as 'fridge' or 'oven', but the frame in question does not. This observation drives our LSTM experiments in order to investigate how the temporal coherence of the detected objects can improve location classification.

To test our hypothesis, we train an LSTM with the dataset of ADL20 D2D 30%. For training, we set the sequence size to 20 frames without augmenting the dataset with overlapping sequences, so each frame's detections are seen once per epoch as part of a single sequence. When testing the previously misclassified frame – now being part of a sequence – we find that the resulting location does not change from 'kitchen'. Another interesting remark from this example is the ability of the LSTM to revert the prediction back to 'kitchen' if it misclassifies certain frames of the sequence. Given a slice of three BPVs with only the 'tap' object and having from previous frames an ongoing location prediction of 'kitchen' with 52% probability, we classify the first BPV. It is classified as 'kitchen', but its probability drops to 49%. The following 'tap'-BPV modifies the prediction to 'bathroom' with probability 50% and 'kitchen' drops further to 47%. This pattern continues for the third 'tap'-BPV. However, given a vector that includes 'fridge', the 'kitchen' prediction returns with increasing confidence, demonstrating the ability of the LSTM to recover from false intermittent predictions. In order to test whether the LSTMs are also quantifiably better than ANNs we repeat the dataset parametrization experiments from Sec. 3.2.3. We expect higher Top1 accuracies, as well as to confirm the relative associations between L2L, L2D, D2D, the object detector datasets, and the detection thresholds.

For our experiments we use two stacked LSTM layers and a fully connected layer, attached at the last sequence step of the second layer to produce the output. We vary the feature size between 20, 48, and 80 following the BPV requirements. We set the number of hidden units per layer to double the feature size. The numbers of trainable parameters for each input size is 23,327, 131,239, and 362,087, respectively. Following the ANN training scheme, we use categorical cross entropy to calculate the loss and SGD for optimization. All models are trained for 150 epochs with 10^{-2} starting learning rate divided by 10 every 50 epochs. The sequence size is set to 20 which corresponds to a video duration of 20 seconds, (*i.e.*, 20 frames sampled at 1fps). The batch size is set to 16 sequences.

At training time, we use a single label to describe a sequence. To produce it, we perform majority voting on all location labels in the sequence and use that as the ground truth. Thus, the classifier is trained to produce one prediction per sequence.

At test time, we want to evaluate every frame in the test set. To that end, we clone the prediction to the sequence length and evaluate all labels one-by-one.

In Table 3.9 we report Top1 accuracies. For every task the LSTM model surpasses its ANN equivalent. Except for the L2L combinations where it is relatively close (2-4% difference), LSTM shows significant improvement, especially at the hardest cases, *e.g.*, ADL48 L2D 60% (+20.8%) and ADL48 L2D 70% (+16.5%). The same conclusion can be drawn from Table 3.10 where we present F1-scores. As expected, the absolute values are lower compared to Top1, due to the class imbalance.

Tab. 3.9.: LSTM Top1 location accuracy (%) – averages of the best five models of each experiment. In parentheses, the difference to the corresponding ANN result.

L2L		Conf.	L2D		D2D		
ADL20	ADL48	(%)	ADL20	ADL48	ADL20	ADL48	COCO
80.238 (+2.54)	80.632 (+3.59)	30	70.699 (+11.01)	63.815 (+9.01)	70.107 (+7.16)	65.072 (+8.60)	75.491 (+11.13)
		40	66.832 (+8.72)	63.834 (+15.68)	69.104 (+8.36)	62.655 (+6.79)	73.932 (+11.06)
		50	68.831 (+12.19)	61.484 (+13.82)	67.189 (+8.46)	59.752 (+4.64)	73.111 (+12.27)
		60	63.431 (+8.43)	60.165 (+20.80)	62.840 (+7.07)	59.315 (+7.26)	72.370 (+14.57)
		70	61.732 (+14.12)	55.445 (+16.49)	61.857 (+10.20)	56.731 (+8.03)	67.069 (+15.35)

Tab. 3.10.: LSTM F1-scores averaged over seven locations for the best performing model in Top1 accuracy. In parentheses, the difference to the corresponding ANN result.

L2L		Conf.	L2D		D2D		
ADL20	ADL48	(%)	ADL20	ADL48	ADL20	ADL48	COCO
63.793 (+5.32)	58.923 (+1.19)	30	54.099 (+8.12)	54.154 (+12.60)	52.607 (+5.17)	51.421 (+10.03)	53.543 (+10.38)
		40	52.259 (+9.67)	48.385 (+9.17)	53.782 (+8.60)	49.091 (+8.84)	51.45 (+11.97)
		50	52.587 (+9.71)	47.171 (+4.16)	49.434 (+6.83)	46.28 (+7.28)	50.864 (+14.08)
		60	42.147 (+2.94)	45.754 (+16.08)	46.914 (+7.87)	45.57 (+10.7)	50.065 (+15.31)
		70	46.012 (+11.86)	39.938 (+17.69)	47.002 (+12.74)	41.196 (+9.31)	45.776 (+15.52)

3.3.2 Activity Classification

In Sec. 3.3.2.1 we present the results of the activity classification scheme and in Sec. 3.3.2.2 an analysis of the class confusions. All our tests consider the **detections to detections** (D2D) dataset combination introduced in Sec. 3.2.3, where both train

and test splits are built from the output of an object detector. This provides a more realistic scenario for smart-home application development compared to the label to label (L2L) combination which assumes ideal object detections. Despite recent improvements in object detectors [48, 120, 121], perfect detections are not yet feasible and flawless annotations in unseen environments require significant human labeling effort.

3.3.2.1. LSTM Classification

For the activity classification experiments, we train an LSTM network for the sequences of each feature combination of Sec. 3.2.4 targeting the 33 activity classes of the ADL dataset (Table 3.3). We prefer LSTM over Artificial Neural Networks for their ability to incorporate temporal changes, compared to per-frame classification schemes that do not consider objects seen in the past. We train a single layer LSTM with 80 hidden units with a fully connected layer for the output. We set the sequence size to 150 frames and batch size to 64. We apply 15% dropout with 10^{-4} starting learning rate with polynomial decay down to 10^{-6} in 1000 training iterations. We finish training after 1500 iterations.

We report Top1 accuracies and F1-scores in Table 3.11. The best performing individual feature is the binary presence vector (BPV). Adding the bounding box coordinates hurts results but adding the centrality feature leads to the best performance overall. The high number of classes adds complexity to the classification task when compared to locations and leads to lower results overall.

Tab. 3.11.: LSTM Top1 accuracies (%) and averaged F1-scores for all 33 classes for the proposed feature combinations.

Feature	BPV	BB	CF	BPV+BB	BPV+CF	BB+CF	BPV+BB+CF
Top1	32.16	26.66	23.38	29.21	32.84	31.06	**33.97**
F1	10.51	9.07	6.69	11.11	10.89	10.83	**12.40**

3.3.2.2. Class Confusions

The confusion matrix in Fig. 3.5 shows a specific trend. It suggests strong preference to certain activities, including 0: 'background', 3: 'brushing teeth', 9: 'doing laundry', 10: 'washing dishes', 12: 'making tea', 17: 'making cold food/snack', 22: 'watching TV', 23: 'using computer', 27: 'reading book'. Beside their true-positives, these classes attract false-positives from conceptually relevant activities that rely on the

same objects for recognition, but have fewer instances associated with them at training time (Table 3.3).

Class 17: 'making cold/food snack' contains false assignments from classes 16: 'making hot food' and 18: 'eating food/snack'. These activities rely on similar kitchen objects such as 'dish', 'mug/cup' and 'tap', but the classifier assigns them to the class with the most instances during training. Similarly, instances from 29: 'writing' are assigned to 27: 'reading book' based on 'book' as the detected object. Further confusions that regard semantic relevance include classes 1: 'combing hair', 2: 'wearing make up', and 4: 'using dental floss' with 9: 'doing laundry'; class 28: 'using mouth wash' with 3: 'brushing teeth' and class 32: 'grabbing tap water' with 10: 'washing dishes'.

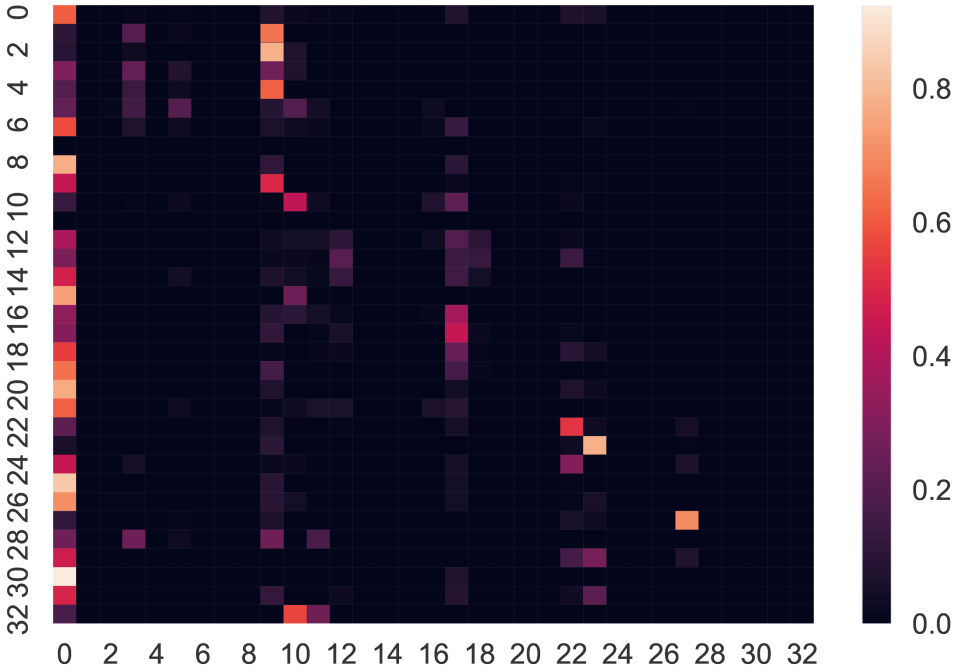


Fig. 3.5.: Confusion matrix for BPV + BB + CF. Some activities can be assigned to semantic super sets. Class 0 is the background class. The 32 activities are listed in Table 3.3.

3.4 Discussion

We envision a system that recognizes activities and locations based on detected objects in a real setting. We structure the task in a very simple way, *i.e.*, to solely rely on the presence of objects in the scene for inference. This is a source of confusion

even with the assumption of perfect detections, considering that objects are naturally found in multiple locations ('door') or are movable ('cup'). We work with these limitations and explore ways to address them by relying on the temporal associations of objects to learn an improved representation of a scene or an activity.

Using the L2L combination for the location classifiers is not a pragmatic approach mostly due to the difficulties in data collection that is the human effort in annotating customized home environments. To mitigate this, we make use of automatic object detectors, pretrained on specific sets of object classes. Initially, this leads us to evaluate the L2D scenario, where we train on generic room representations, *e.g.*, a common kitchen has a fridge, an oven, and a tap which are expected to be found in the test environment. However, the D2D classifiers perform better and show increased resilience to noisy detections at test time. Additionally, they are more convenient from an installation perspective, since they abolish the necessity for labeling locations with objects. Thus, having minimized the required human labelling effort, it becomes easier to learn new representations of existing places (for example, with a specialized detector that was previously unavailable), but also of unseen locations not included in the original categories.

Our purpose is to evaluate the applicability of the detectors in terms of object variability and acceptable accuracy in activity and location classification. The L2L experiments have the highest location classification performance and establish the idea that the presence of noise in the object detections (in this case, the lack thereof) can influence results. A second significant outcome is that the variability of available objects can enhance the ability of a classifier to detect a location accurately. This is observed from ADL48 L2L which outperforms ADL20 L2L in Top1 accuracy in the LSTM tests. However, when the objects contain noise, less is more, *i.e.*, in L2D and D2D, ADL48 does not outperform ADL20 in any (LSTM or ANN) experiment.

To enhance our original objective into activity classification, we consider a scenario with detections of house-related objects with additional object appearance features that are dynamic in time. We examine them with LSTM and find that this more complicated scenario cannot be sufficiently tackled with object-based features. While seven locations have been distinguished with up to 70% accuracy (LSTM ADL20 D2D 30%), the 33 activities reach 34%, with multiple intra-class similarities being observed and misclassified as such. The additional information about the object detections, *i.e.*, the bounding box location and its centrality contribute to improved performance. However, acquiring fine-grained associations between the object type and its position in the scene would require a much larger training set and a larger network that is better equipped to handle the additional information. The current

setup is incomplete in this effect and cannot fully address the elaborate task of analyzing semantically similar activities and counter the bias towards classes with higher prior probability.

To our knowledge there is no related work that tackles location classification in the ADL dataset. For the task of activity classification, related work does exist; however, the evaluation metrics and the regarded activity sets do not allow for a fair comparison. For example in [103] the authors report 26.3% average accuracy for the subset of 18 activities of the ADL dataset.

4 Hand- and Object-based Action Classification

4.1 Introduction

The prominent characteristic of egocentric vision is that it provides a first-person perspective of the scene by placing a forward-facing wearable camera on the chest or head of a human. This provides a unique view that is person-centric and optimally set to capture information that is arguably relevant to the camera wearer [65]. Naturally, this refers to the surrounding area and its contents, usually consisting of objects, hands, other people, and the scene background. Being able to examine a perspective of the scene that accumulates this information with clarity allows for improved inference of higher level cues such as the quantification of interactions between hands based on their proximity [103], object-activity relations from associated movements [7], and location identification from the presence of distinctive objects [68, 70] as shown in Chapter 3.

Egocentric vision is a strongly application-oriented field of computer vision. Research is not only focused on developing algorithms to tackle traditional computer vision tasks such as object detection and activity recognition but also considers them from the perspective of applicability [114]. Egocentric vision utilizes mobile devices for image and video capturing for storing, transmitting, or *in situ* data processing. These operations can suffer from restraints in processing power, energy usage, transmission bandwidth, or lack of a dedicated storage space. Such issues should be considered when designing an egocentric vision-based solution. In this chapter, we consider these limitations by reducing the amount of information used as input for human action recognition from videos.

Understanding of visual content in human intelligible terms remains challenging despite being facilitated from the egocentric perspective. That is, recognition of a specific area in view as the 'hand' or as a specific object class which may or may

Published as:
G. Kapidis, R. Poppe, E. Van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. "Egocentric Hand Track and Object-Based Human Action Recognition". 19th IEEE Conference on Ubiquitous Intelligence and Computing (UIC). Leicester, UK, 2019, pp. 922–929

Fig. 4.1.: Output from a single-frame hand detector on two consecutive frames of the EPIC-Kitchens dataset [24].

not be under manipulation, will always deteriorate due to inter-object and object-hand occlusions. In the video domain, the recognition task can become particularly challenging in the egocentric setup, due to motion introduced by rapid movements of the camera or the objects seen from this perspective. In Fig. 4.1 we illustrate an example where a slight motion of the left hand causes motion blur and a subsequent missed detection.

The intricacies of egocentric vision pose a challenge for algorithms developed in the context of third-person vision in terms of their applicability to first-person views. We argue that methods yielding cues towards human-like understanding of a scene from one domain can be compatible with egocentric vision given a certain amount of finetuning. The scope of this chapter is to assess up to what point existing object detection and tracking schemes can produce valuable information for egocentric action recognition. Our idea relates to methods that *reduce* RGB images to trajectories or poses of hands or objects in the scene and use this contextual information alone for human action recognition [5, 103] or a related task such as prediction [38]. Our objective is to investigate the information encoded in the movements of hands and objects in contrast to the currently predominant approach of using the visual information directly [17]. Are the motion sequences alone sufficient to model actions? We test the limits of object detection and tracking methods in their ability to produce usable data towards higher-level inference.

Initially, we focus on distinguishing the action-specific cues that can be acquired solely from hand movements. Instinctively, human hand movements are expected to carry much of the spirit of actions that are explicitly named after the actual motion itself, *e.g.*, 'put', 'take', 'stir', 'open', 'close' to name a few. We strive to exploit the clear view of the hands and their movements in egocentric videos and study them closely towards identifying associated actions, facilitated by detection and tracking of hand regions. We consider hand regions to be containing the hand and possibly the wrist and the lower arm, depending on the output of the hand detector. Furthermore, we

examine the structure of actions which are generally not only associated with the hand movements but also related to objects of interest arising from the context of the scene. For example, the action 'wash dishes' in Fig. 4.1 consists of the moving hands manipulating kitchenware and a sponge, while there is a sink and running water.

This chapter is directly associated with the production of hand trajectories. The prelude is that an object detector is applied on egocentic videos to accurately detect the hands, thus substituting the arduous task of manually labelling hand regions with an automated process. Subsequently, tracking is applied to temporally associate the detections into meaningful sequences. These are processed to clear overlapping misdetections and are then attributed to the left or the right hand. Finally, the hand trajectories are augmented with the detection of objects and used as input for action classification. We focus on action classification by incorporating the binary presence vector introduced in Chapter 3 to a hand tracking pipeline. Fig. 4.2 illustrates our approach.

The contributions of this chapter are threefold:

- A hand detection, tracking, and identification pipeline that extracts hand motions from egocentric videos, structured to provide the hand positions in every frame including their distinction into left and right hands.
- The assessment of the capability of hand tracks alone for egocentric action recognition and the effects of temporal sampling in the representative ability of hand motions.
- The inclusion of object presence and position to capitalize on the limited set of specific actions that an object can be associated with.

In Sec. 4.2 we describe our hand detection, tracking, and identification pipeline and in Sec. 4.3 the temporal classification problem for action recognition. In Sec. 4.4 we delineate our experiments and results. Finally, in Sec. 4.5 we discuss our findings and conclude this chapter.

4.2 Hand Track Dataset

In this section, we describe the process to produce a hand track dataset from the raw frames of EPIC-Kitchens [24].

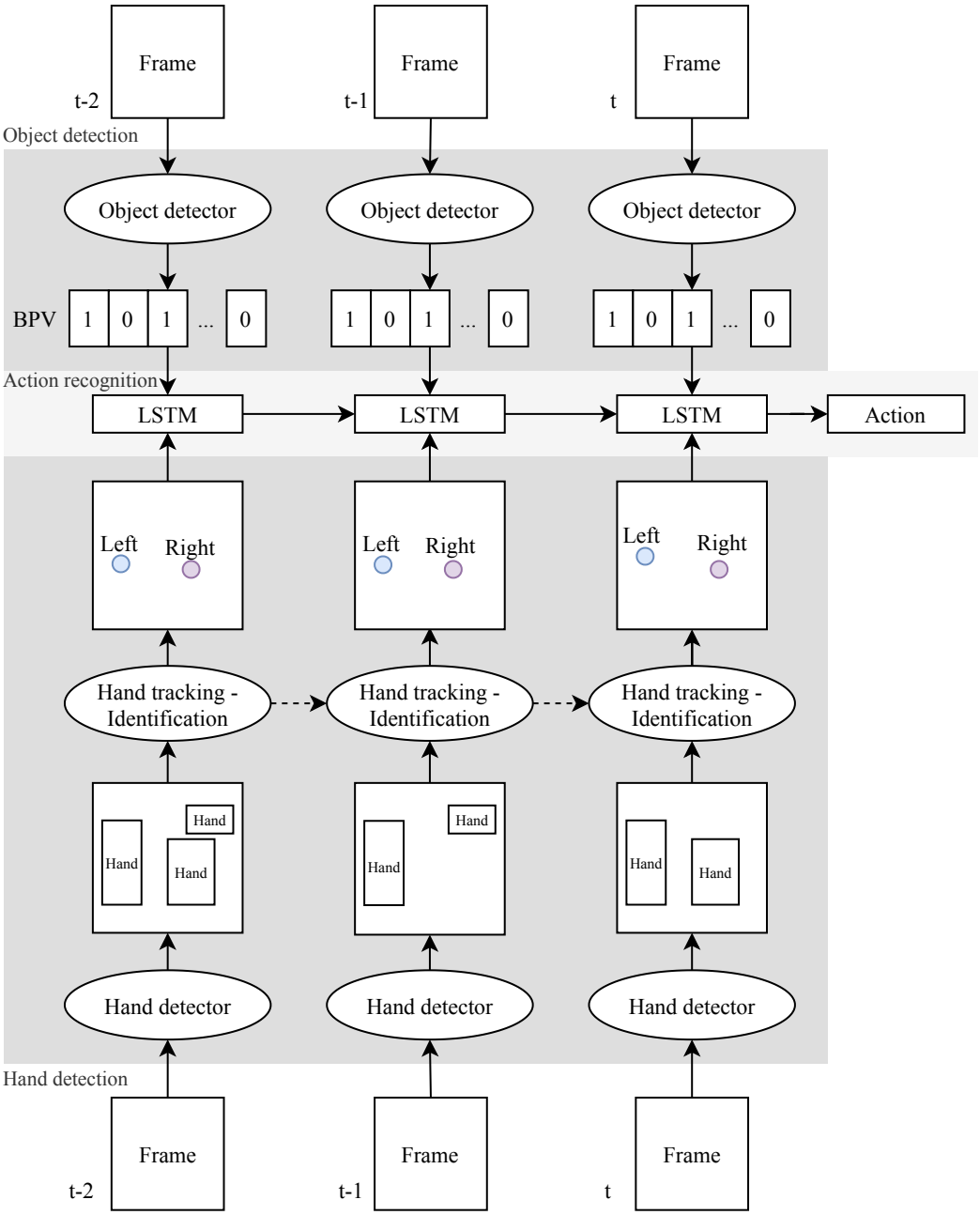


Fig. 4.2.: Our pipeline for action recognition. Hands are detected with YOLOv3 [120] in EPIC-Kitchens [24] and tracked by detection using SORT [11]. We remove track overlaps and identify left and right hands from their position in the frame. For the objects we also rely on YOLOv3 with a second model trained on the noun classes of EPIC-Kitchens. Binary presence vectors are propagating the object knowledge per frame. Finally, hand and object information is used as input to an LSTM to classify actions.

4.2.1 EPIC-Kitchens

We partition the train set of EPIC-Kitchens into custom train and validation splits based on the participant identities to avoid videos from the same person appearing in both splits. Videos from participants 1-8, 10, 12-17, 19-24 comprise the train set and videos from 25-31 the validation set. Videos from participants 9, 11, 18, and 32 are reserved for a hidden test set which can be evaluated online through the authors' website[1]. Additionally, almost 300k object bounding boxes are provided for the videos of the original train set which we utilize to train an object detector (Sec. 4.2.4).

The structure of the hand track dataset follows the action annotation format of EPIC-Kitchens. Based on the annotations, the videos are split into short video segments, each comprising an action with a specific verb and noun label, start and end frames and times. Our aim is to document the position of each of the two (at most) visible hands (those of the camera wearer) in view as an (x, y) coordinate pair for each frame. The coordinate pair signifies the center of the bounding box of a detected hand, without considering the size of the detection. In the remainder of this chapter, our subset of EPIC-Kitchens is referred to as EPIC-Kitchens, unless stated otherwise.

4.2.2 Hand Detection with YOLO

In order to acquire hand regions from EPIC-Kitchens we train a hand detector with YOLOv3 [120] on the combination of a collection of egocentric hand datasets.

4.2.2.1. Dataset Collection

We utilize hand annotations from existing egocentric datasets, namely EgoHands [3], EGTEA Gaze+ [85, 86], CMU EDSH [84], and THU-READ [151]. We start with the EgoHands [3] dataset. It consists of 48 videos with 100 labeled images for each, totalling 4,800 frames with 15,053 annotated hand masks. The dataset depicts dynamic interactions such as playing chess or cards, hence the high number of hand instances per image. Additionally, we consider EGTEA Gaze+ [85, 86] with the hand annotations consisting of 15,176 masks from 13,847 images. EGTEA Gaze+ is relevant to our task because it considers cooking activities, similar to EPIC-Kitchens. Further hand masks are acquired from the CMU EDSH dataset [84] which consists

[1] https://epic-kitchens.github.io/2020-55.html

Tab. 4.1.: Collection of hand annotations from egocentric datasets.

Dataset	Hand view	Images	Masks	Train	Test
EgoHands [3]	camera wearer/other	4,800	14,884	11,440	3,444
Egtea Gaze+ [85]	camera wearer	13,847	15,258	14,295	963
CMU EDSH [84]	camera wearer	743	1,394	1,186	208
THU-READ[151]	camera wearer	652	1,331	1,252	79
IEORD[123]	-	11,683	0	-	-
Combined		31,725	32,867	28,173	4,694

of 743 frame level masks in total, labeling one or two hands for each. The same annotation format is used in the THU-READ [151] dataset for 652 frames.

Since we are interested in detection and not segmentation we only keep the bounding rectangle of a hand mask and use it as the ground truth for a hand region. Next, we augment the dataset with negative samples, *i.e.*, frames that do not contain visible hands or annotations, in order to punish the objectness learning part of the network and produce fewer false proposals, ultimately reducing false-positive hand detections. We manually annotate 11,683 such frames from the Intel Egocentric Object Recognition Dataset (IEORD) [123]. The number of hand annotations and the size of the train and test splits per dataset are detailed in Table 4.1.

4.2.2.2. Training

We perform various experiments to train the hand detectors to determine the optimal dataset combination that supports generalization, since our eventual task is to apply the detector on an unseen dataset for extraction. We train a detector for each available hand dataset (except IEORD) and one for the combined train sets (including IEORD). All detectors are trained for a single target class '*hand*' without distinguishing between 'left' and 'right'. We set the batch size to 64, starting learning rate is 10^{-3}, momentum 0.9, and weight decay 5×10^{-4}. To initialize the hand detector we use weights that are pretrained on ImageNet [28] and then on MS COCO [89]. We do not recalculate box anchors for our datasets after preliminary tests suggesting minor performance decline. Training takes place for multiple input detector dimensions starting from 416×416 to enhance the detection of objects with different sizes. We evaluate every detector on every test set using Average Precision for 50% Intersection over Union-IoU (AP_{50}) and False Detection Rate[2] (FDR = 1-Precision). The acceptance threshold for detections in set to 25%.

[2]Since we plan to use the detections as a preamble for tracking, the frequency of false-positive detections is an important metric to consider.

Tab. 4.2.: Average Precision (%) with 50% IoU threshold. Row-wise the dataset used for training and column-wise the dataset used for testing. In parentheses the False Detection Rates for each test set. (For IEORD we only report the absolute number of false-positive detections.)

	IEORD [123]	EDSH [84]	EgoHands [3]	EGTEA+ [85]	THU-READ [151]	Combined Test
EDSH	71	100.00 (0.00)	17.00 (0.49)	72.50 (0.21)	86.17 (0.09)	31.98 (0.34)
EgoHands	15	26.09 (0.53)	90.58 (0.02)	43.42 (0.39)	17.15 (0.79)	77.29 (0.13)
EGTEA+	37	89.91 (0.04)	20.51 (0.51)	89.65 (0.05)	74.58 (0.07)	41.05 (0.32)
THU-READ	191	90.62 (0.01)	19.24 (0.50)	65.85 (0.19)	100.00 (0.00)	33.07 (0.38)
Comb. Train	18	100.00 (0.00)	90.53 (0.03)	89.38 (0.07)	90.05 (0.05)	**90.42 (0.04)**

In Table 4.2 we illustrate the best performing weights of each detector, based on the two metrics. In total, each detector is able to perform well on its individual test set. We notice performance drops when a detector is applied on a different test set. The detectors from EGTEA, EDSH, and THU-READ are somewhat more compatible with each other. The duality in the viewing perspectives of hands in EgoHands introduces a failing testing scenario for the detectors that have not seen EgoHands data in their training set. EgoHands appears to be the best single-dataset training set for a more general application. Finally, we show that in terms of both AP and FDR, the detector based on the dataset combination performs best on the combined test set and is on par with every single-dataset detector on its individual test set.

4.2.2.3. Detection on EPIC-Kitchens

We apply the combined hand detector on the EPIC-Kitchens dataset to extract hand instances using the same acceptance threshold. In Figs. 4.1 and 4.3 (column 1) we show that the detector generalizes in unseen images, but slight visual changes due to hand movements, strong ego-motion, changing illumination, or occlusions can cause missed detections. Finally, we address the issue of overlapping detections for the same hand region in Sec. 4.2.3.2.

4.2.3 Hand Tracking with SORT

The continuity of visual information in video streams enables the use of tracking methods to replace missing detections. The object detector operates per frame, whereas tracking by detection combines information from multiple frames. We utilize Simple Online and Real-time Tracking (SORT) [11] for this task. It associates detections over the course of a video with threshold-based tolerance to missed ones. Bounding boxes are associated through frames and identified as belonging to a track

with a certain identity. SORT uses the Kalman filter [63] to predict the coordinates of what would likely be the next bounding box of an existing track and the Hungarian algorithm [79] to assign the detections from subsequent frames to existing tracks or new ones. Tracking with SORT is controlled by three parameters:

- IoU_{min} is the minimum required overlap between a new detection and the predicted target from a track that leads to the detection's assignment to it,
- T_{LOST} defines the number of frames that a track can survive for before being finalized, without being assigned a detection,
- T_{min} is the minimum number of consecutive detections required to instantiate new tracks or recover them after encountering frames without assignment.

We set IoU_{min} to 10% to promote track continuity against strong ego or hand motions. In practice, frames that include quickly moving hands can be incomprehensible to the detector and detection will fail. Once the motion weakens, the hand may be detected in a slightly different location. If IoU_{min} is too strict it will not be possible to resume the track, leading to inconsistent single-frame tracks, as well as the simultaneous presence of multiple tracks for the detections of consecutive frames that could not be merged to a single track. T_{LOST} equals to 10 frames without detections to allow for sufficient time, (*i.e.*, 167 milliseconds in the 60fps videos of EPIC-Kitchens) to re-establish a track. Finally, T_{min} is set to one detection, in order to create or revive a track instantly. In column 2 of Fig. 4.3 we show the instantiated tracks as points in the centers of bounding boxes.

4.2.3.1. Track interpolation

We introduce the concept of *intermediate frames*. We define them as the video frames that are implicitly included in a track by means of previous and future frames that contain a detection for them. Intermediate frames do not hold a detection for the track. However, we assume that the hands do exist in these frames but are missed from the object detector. That is, due to the inherent continuity of information in sequential video frames and the short time span we allow for a track to be kept alive without a detection. To augment the tracks for the intermediate frames, we apply linear interpolation on the box centers starting from the last frame with an assigned value for the track until the latest one. The number of interpolated frames is bound to the value of T_{LOST}. In column 3 of Fig. 4.3 we showcase this for frames 2-4 being assigned with interpolated coordinates for the right hand.

4.2.3.2. Track elimination

In the videos of EPIC-Kitchens the participants undertake kitchen activities alone. This limits the maximum number of co-occurring hand tracks at any given moment to two, one for each hand. Particularly, we assign each track to the left or the right hand of the participant based on the location of the center of the first detection of an assumed track. Overlapping tracks for the same image region that have been associated with the same hand are removed and the longest track survives. An example is shown in Fig. 4.3 frame 9 with the elimination of a superfluous track for the right hand. Finally, for the frames with no available detection and track information we assume a hand position below the field of view of the camera. Potential downsides of our hand coordinate detection method are discussed in Sec. 4.5.

4.2.4 Noun Object Detector

To study the hand-object relationships we rely on information about object presence. EPIC-Kitchens includes object labels for the majority of its noun classes. These object annotations amount to $326,388$ bounding boxes. We utilize them to train an object detector using YOLOv3 with the same parameters as those in Sec. 4.2.2.2 with the exception of the base network dimensions which are increased to 608×608 and the introduction of Sparse Pyramid Pooling [50] in the model structure to enhance the detector's ability to find smaller objects. We train for 50k iterations (almost 11.5 epochs) after which the average loss is stabilized. We apply the detector on all the frames of EPIC-Kitchens and accept detections with confidence greater than 25%.

4.3 Motions to Actions

We aim to develop a frame-wise correspondence between every image of EPIC-Kitchens and the hand detection tracks in order to exchange the visual information with our representation. The pipeline of Sec. 4.2 contributes knowledge about the hand locations whether they have been detected in a given image or not. This continuous evolution of positions leads to a sequence of coordinates, which in the temporal dimension capture the motions of the hands. To gain knowledge about these motions we formulate the problem of hand track classification as a *sequence learning problem*. Long Short-term Memory (LSTM) networks [51] have shown ability to model long-term dependencies in sequences of arbitrary sizes of coordinate

Fig. 4.3.: Hand tracking and track augmentation on a 10-frame sequence (rows). Each column showcases a step. Column 1 visualizes the detections (rectangles). In column 2, SORT associates them to distinct tracks (points and lines). A track for each hand spans the frame sequence. Frames 2-4 and 7 are still not assigned with a coordinate for the right hand. For these frames we interpolate between the last available coordinate and the latest as shown in column 3. The right hand track is augmented in these frames and the missing detections are covered. In column 4 we show the final tracks for both hands. A case of removing a redundant right hand track is shown for frame 9.

[42, 103] or object presence [70] data and we employ them for their classification into actions.

4.4 Experiments

We construct a series of experiments to investigate the ability of the hand track and object presence information to substitute the visual information of raw RGB frames for detecting hand related actions from the egocentric perspective.

For our experiments we use LSTMs with a Fully Connected layer attached to the last hidden state of the final LSTM layer to obtain a class prediction for an action segment. To train the models we use cyclical learning rate (CLR) [136, 137] with a triangular policy that fluctuates between the base and the maximum learning rate in 20 epochs. Batch size is set to 128. We train our models for 1,000 epochs capitalizing on the ability of the triangular CLR policy to move weights out of local minima in search of better configurations. We use categorical cross entropy to calculate the loss and Stochastic Gradient Descent for optimization. Our learning scheme targets the 125 verb classes of the EPIC-Kitchens dataset. For the LSTM experiments we consider the following features:

- The **concatenated Left/Right hand coordinates** (LR) of the center of each hand, normalized to the image size. The values are in the [0-1] range when there is a hand present. Alternatively, the (x, y) coordinate is set to (0.25, 1.5) to declare that the left hand is out of view and correspondingly to (0.75, 1.5) for the right hand. Feature size is 4.
- The **Binary Presence Vector** (BPV) of objects [68, 70] from Chapter 3.2.3 consisting of zeros and ones with length equal to the number of noun classes of EPIC-Kitchens (352). The BPVs are concatenated to the hand coordinates for every frame and the feature size increases to 356 (352 + 4).
- The **tracked object coordinates** (Obj) instead of the objects' BPV in a video frame. This increases the feature length to 708 (352*2 + 4). In case of multiple instances of an object in a frame we only consider its longest running occurrence following the tracking scheme of Sec. 4.2.3. When an object is not present on a frame its coordinates are set to (0,0).

Our motivation for the x hand coordinate when the hand is not in view is based on the expected resting position of the hands on their respective side of the body. For the y value we consider that since no part of the hand is detected it must be resting

below the image frame. On the contrary, we cannot assume a location for every missing object. Thus, we provide a zero value to nullify the effect of the feature when training the action recognition model.

We report classification results for our validation set in Table 4.3 in terms of overall Top1 and Top5 accuracy. Following the evaluation scheme of EPIC-Kitchens we additionally report mean per-class precision and recall for verb classes with more than 100 samples during training, which in our splits are 24.

Tab. 4.3.: Verb classification results on EPIC-Kitchens using different feature sets, model sizes, and length of feature sequences for the LSTM models and frames for the vision models. We compare against vision-based 2D-CNNs using RGB and optical flow (TSN) [160] and 3D-CNNs [20].

	Model Parameters				Accuracy %		Average %	
#	Model	Feature	Hidden/ Layers	Seq. Length	Top-1	Top-5	Cls Precision	Cls Recall
1	LSTM	LR[a]	32/2	Full	31.10	74.12	11.02	10.46
2	LSTM	LR	16/2	32	31.01	73.15	10.38	8.46
3	LSTM	LR+BPV[b]	32/2	Full	**34.97**	**76.08**	15.08	12.00
4	LSTM	LR+BPV	16/2	32	34.05	75.36	17.64	10.83
5	LSTM	LR+Trc BPV	16/2	32	34.05	75.29	19.08	11.51
6	LSTM	LR +Obj[c]	16/2	32	32.81	73.70	12.84	10.41
7	TSN [160]	RGB stream	-	25	36.98	77.89	20.28	13.12
8	TSN [160]	Flow stream	-	25	37.99	76.45	23.14	14.06
9	MFNet [20]	RGB-3DConv	-	16	**44.31**	**79.10**	29.46	21.37

[a]Left/Right normalized hand coordinates (x, y)
[b]Binary Presence Vector of detected objects
[c]Normalized detected object coordinates (x, y)

4.4.1 LSTM Results

Initially, we test the ability of the LSTM to model the hand track sequences in full length using the LR feature. This is no straightforward task since the durations of action segments vary significantly from 0.5 seconds to 3.5 minutes which translates from 30 to as many as 12,000 frames. In our first experiment, we train using the full hand coordinate sequences. For the LSTM to support training in batches with variable sequence sizes we zero-pad the shorter sequences to the size of the longest one in the batch. For the shorter sequences we use the last hidden state before zero-padding as input to the Fully Connected layer and calculate the loss on this prediction. The Top1 accuracy is 31.10% and the Top5 74.12%.

In an effort to simplify and speed-up the learning task, instead of smoothing the coordinates as in [42], we downsample the action segments into shorter lengths. We are inspired by the concept used to train 3D-CNNs, which aim to capture the temporal structure of videos, but due to computational restrictions are unable to process the full frame sequences to represent them [155].

In the second experiment, we sample the coordinate sequences down to 32 steps and use these as input to LSTM. In Fig. 4.4 we visualize the difference between a full and a sampled sequence for the proposed sequence size. Furthermore, we reduce the number of hidden units per layer to avoid overfitting, since the input is reduced significantly. Top1 performance drops ~0.9% compared to the first experiment which can be attributed to the exclusion of temporal structure (we discuss this effect in Sec. 4.5).

(a) Original length **(b)** Sampled to 32 steps

Fig. 4.4.: Left (cyan) and right (purple) hand motion patterns extracted from a 2.7s sequence (161 frames) for action 'clean lid'. (a) The final view of the full sequence including all 161 steps. (b) The same sequence sampled to 32 steps. Zoom-in for best view.

For experiments #3-5, we enhance the LR feature vector with object BPVs by appending them to the hand coordinates for every sequence step. For the 'LR+BPV' experiments we incorporate the detected objects directly and for 'LR+Trc BPV' we track the objects following the interpolation scheme of Sec. 4.2.3.1, in order to gain object presence knowledge for as many frames as possible. In experiment #3 we use the complete motion sequences (following experiment one) and improve Top1 classification accuracy by 3.8% to 34.97%. In experiment #4 we repeat the feature setup of experiment three with sampled sequences. The addition of the BPV feature improves Top1 accuracy in the sampled sequences as well by 3% to 34.05% showing that the improvement from objects is consistent. Tracking the objects in experiment #5 again reaches 34.05% without introducing further improvements.

In the sixth experiment, we use the tracked object coordinates to enhance the LR feature (LR+Obj) instead of the BPV. Again, we notice an improvement over having no object presence (Table 4.3); however, it is not as strong as the BPV-

based experiments. We attribute the smaller increase to the added uncertainty from location information of objects that may be false-positives. The LSTM seems to be able to more adequately reduce the importance of a falsely detected BPV rather than that of a coordinate that is propagated to the whole sequence.

4.4.2 Comparison with Video-based Methods

We perform experiments #7-9 to compare against state-of-the-art video-based methods, Temporal Segment Networks [160] (TSN) and Multi-Fiber Networks [20] (MFNet) that utilize Convolutional Neural Networks as feature extractors for two [160] or three dimensional [20] inputs. In the 2D case, TSN extracts convolutional features from multiple stacks of either images (RGB stream - #7) or pairs of horizontal and vertical optical flow values (Flow stream - #8) that capture the perceived motion through series of images [171]. MFNet (#9) utilizes a set of 16 frames sampled from the sequence of video frames to represent the segment. Both networks utilize only the video (or flow) information without any contextual information about the scene.

In terms of overall Top1 accuracy on the test set, the results are highest (44.30%) when using 3D convolutions. Our methods remain close to TSN, but are still ~2% lower. In Table 4.4 we perform a class-wise comparison for the 10 most common verb classes in our train set, following the analysis in [24]. We see that recall and precision are comparable between our methods and both TSN streams (#1-4 and #7, #8) for classes 'put', 'take', 'wash', 'close', 'mix', 'pour', and 'turn-on'. Against MFNet we are close for actions 'take', 'wash', and 'pour'. The relatively small gap in performance indicates an expressive quality in our data that can lead to action comprehension comparative to more elaborate methods by using only object detection and tracking as the means to deliver the input.

Tab. 4.4.: Comparison for the 10 most frequent verb classes in our training split. Showing per-class Recall and Precision, results in %. The experiment identities (#) correspond to the experiment identities in Table 4.3.

#	put		take		wash		open		close		cut		mix		pour		move		turn-on	
	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P
1	42.36	33.29	48.55	28.16	63.09	36.78	11.85	26.50	12.90	19.13	24.35	45.16	13.07	30.30	33.82	18.85	0.00	0.00	0.00	0.00
2	64.53	28.89	40.57	29.10	52.89	39.60	2.87	55.56	0.00	0.00	14.78	47.89	27.45	25.61	0.00	0.00	0.00	0.00	0.00	0.00
3	42.05	34.78	56.61	31.57	68.32	43.62	18.93	39.92	16.72	27.40	28.26	34.03	43.14	38.60	8.82	13.64	0.00	0.00	2.20	50.00
4	34.53	34.48	68.58	28.66	62.55	44.42	17.02	37.87	12.90	36.67	20.87	47.06	28.10	55.13	1.47	4.55	0.00	0.00	1.10	12.50
7	57.24	29.89	43.19	29.80	58.93	60.3	43.21	51.36	12.61	39.45	54.35	67.57	30.72	51.09	10.29	26.92	0.00	0.00	2.20	20.00
8	31.87	38.95	81.81	29.90	37.05	56.33	58.51	58.40	24.34	62.88	23.91	60.44	41.18	70.00	32.35	28.57	0.00	0.00	0.00	0.00
9	52.62	39.55	60.88	37.47	62.01	63.20	55.64	53.49	26.39	54.88	60.43	64.35	53.59	61.19	26.47	36.73	0.00	0.00	24.18	31.43

4.5 Discussion

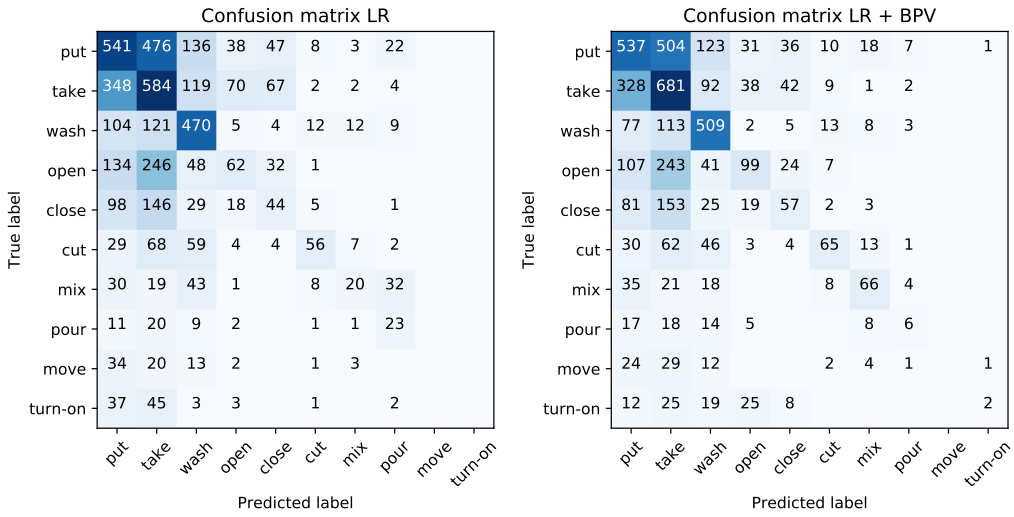
In this section, we developed a pipeline to capture the motions of regions of interest, in order to model the underlying human actions, from an egocentric perspective. This process is in close relation to the information it aims to comprehend and addresses specific issues that stem from detection and tracking in an egocentric setup. For example, in Fig. 4.1 we demonstrate a persistent complication with the hand detections that is successfully solved with tracking.

An issue that may be introducing inconsistency to the hand tracks is the detection of both hands as one region of interest (~2.5% of detections). This leads to the assignment of the detection in either the left or the right hand track and momentarily produces an outlier coordinate (since the center of the detection is abruptly found elsewhere - see Fig. 4.5). A way to suppress this could be to add a third 'dual hand' track, (*e.g.*, LR+D as a feature) that captures these sequences and incorporates them in the final model as such; a promising direction for future work.

The contextual information added from the detection of objects, other than hands, contributes to the knowledge about actions; however, we argue that there are potential improvements with more accurate object detections. Our findings in Chapter 3 about the effect of using object annotations over detections also support this claim.

We view the process of standardizing all sequences to a certain length in experiments 2,4,5,6 as a manipulation of their temporal structure. After sampling, the *temporal distance* between consecutive steps is not fixed to 16.7 milliseconds (see also Sec. 4.2.3) but becomes a function of the sequence length and the sampling rate, which in turn originates from the volume of samples in the learning phase and is not fixed for any two sequences. In essence, we sacrifice part of the information related to the precise duration of each tiny motion step. The trade-off to the reduced performance in these experiments is the significantly shorter training times per mini-batch (0.321s to 0.025s) and epoch (57.2s to 4.4s) in our setup with a 1080Ti GPU. An additional interesting direction for future work is the investigation of spatial

Fig. 4.5.: A double hand detection finds its way into the right hand track.



(a) LR **(b)** LR + BPV

Fig. 4.6.: Confusion matrices for experiments one and three for the 10 most frequent verb classes in our training split. The similarity of the motions leads to a high number of predictions being associated to a different class. The presence of objects alleviates this issue for most classes.

smoothing techniques, (*e.g.*, as in [42]) instead of temporal to simplify the motions, which could ultimately correlate coherent gestures to specific actions. A potential disadvantage of temporal subsampling is loss of prediction accuracy in an online classification setting. A solution to this would be reducing sampling on the input stream to match the subsampling step of the model.

Hand tracks are our primary means for distinguishing human actions. Due to the high representative ability of human hands and their multipurposeness, the same motion can be expected to be part of multiple actions. For example, 'pull' and 'take' are conceptually alike, hence the related hand motions are also expected to be similar. We showcase this confusion in Fig. 4.6a. This introduces an additional burden to our representation which we enhance using the objects in the scene (Fig. 4.6b). Investigating other sources of contextual information, such as the explicit duration of hand and object movements with an emphasis on hand-object interactions, together with improving existing sources through the removal of ego-motion from the hand tracks and the enhancement of object detection are additional directions for future work.

5 Multitask Learning to Recognize Egocentric Actions

5.1 Introduction

Human activity recognition from video is a growing field of computer vision that promises real-time and large-scale behavior recognition and automated analysis. Activity recognition applies to both the third- and first-person vision domains, incorporating the distinct visual characteristics of each case. Third-person videos tend to capture the full range of motions of the human body from a static point of view. The viewing angle in egocentric videos matches that of the human performing the activity, providing a unique, moving perspective of the scene [65]. At the same time egocentric videos usually offer a clear view of the camera wearer's hands [33], which in many cases are essential for the execution of an activity. An outlook of the objects manipulated by human hands promises additional cues about the performed activity, culminating to improved recognition performance [4, 33, 86].

In order to expand the feature space, a network can be augmented with additional data modalities such as optical flow [133], depth [151], or segmentation masks around interesting areas [157]. The additional information aims to guide the network towards learning more activity-specific features that it might otherwise have missed. In order to incorporate the supplementary inputs, networks comprise multiple streams and their results are combined at a later stage. The multi-stream approach is associated with the combination of the individual feature sets towards an extended and more expressive representation from which the structure of activities is inferred.

A related but fundamentally different concept that we employ in this chapter is that of Multitask Learning (MTL) [19]. The idea behind MTL is to train a neural network with multiple related objectives (tasks) while sharing as much as possible of a common network structure [19]. Branch diversification occurs only for the

Published as:
G. Kapidis, R. Poppe, E. van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. "Multitask Learning to Improve Egocentric Action Recognition". IEEE International Conference on Computer Vision Workshops (ICCVW). 2019, pp. 4396–4405

task-specific output layers and there are as many output layers (branches) as there are learnable tasks spawning from the main network block (Fig. 5.1).

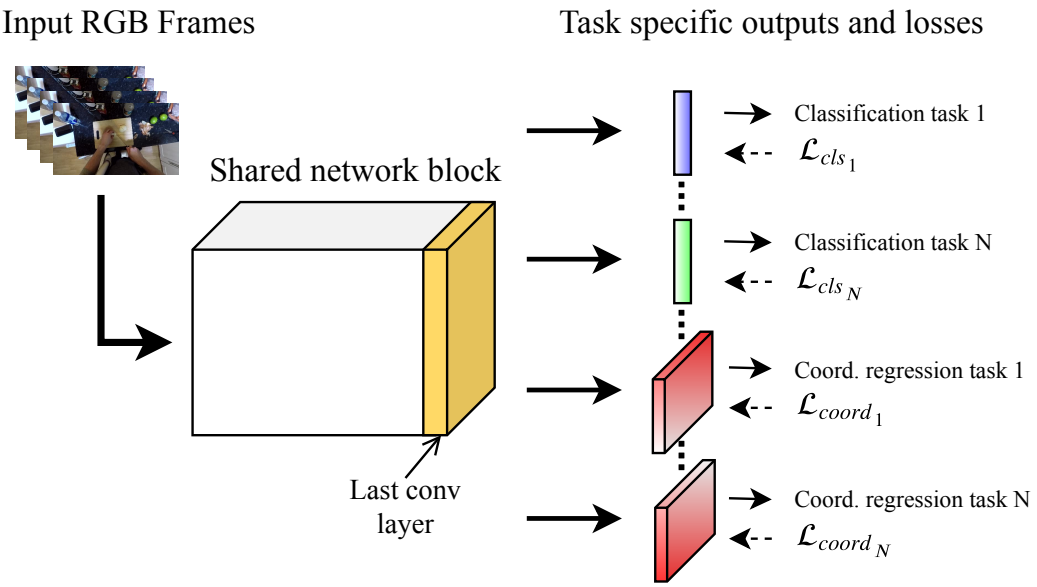
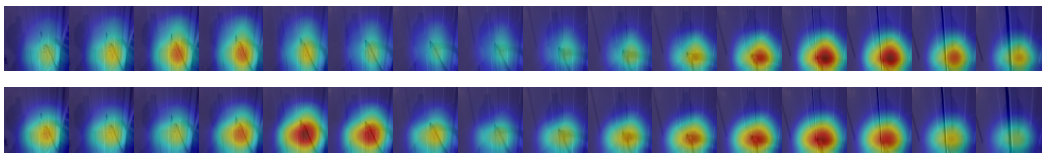


Fig. 5.1.: MTL Network Structure. The shared network block can be any convolutional network that extracts features from the input. Each task-specific output layer is plugged to the 'last conv layer'. They are independent from the others and their parameters are trained individually. However, all output layers use the feature representation produced by the shared part of the network as input.

MTL is conceptually the opposite from multi-stream approaches, since the additional information is not used as input, but is expressed as the output of the network and is only required for supervision. The significant merit of MTL over multi-stream methods is that the additional information is only needed during training and what would otherwise come from the additional input modalities is already incorporated in the network weights at test time. For example, in the video domain, the input of a network remains the same set of RGB images regardless of the number of tasks.

The premise of MTL is that by combining the objectives of related tasks in the same network, we can benefit from their structural commonalities. This is the case because the weights of the shared network block aim to jointly encapsulate each task's representation requirements. When these are complementary, they enhance the inputs of the task-specific output layers. Then, inference can be improved for all or some of them or just the one that we focus on the most, by using the best performing weights of the task [19].

In this chapter, we utilize MTL to improve action classification performance in egocentric videos. We are motivated from the idea that we developed in Chapter 4



(1) (2) (3) (4) (5) (6) (7) (8) (9) (10) (11) (12) (13) (14) (15) (16)

Fig. 5.2.: Visualizing the class activation maps [143] for an instance of class 'open' from EPIC-Kitchens [24]. Top: Multi-Fiber Network (MFNet) [20] trained end-to-end for the single task of classifying short clips into actions. Bottom: MFNet trained to additionally predict one (x,y) coordinate for each hand. Training with the hand coordinates as the extra task leads to a greater inclusion of the right hand area into the class activation map.

that hands are critical for the comprehension of egocentric actions, while remaining difficult for networks to capture this delicate motion information. In Fig. 5.2, we show that by having the network learn hand regions explicitly as an extra task in addition to the actions, we steer it to produce activation maps that cover the corresponding hand areas to a greater extent. Eventually, incorporating these areas also improves the action classification results.

We experiment with egocentric video datasets EPIC-Kitchens [24] and EGTEA Gaze+ [85] by explicitly utilizing the location of hands, gaze, and other signals towards actions. We leverage the motion and visual attention information that is present in the hand movements and the gaze of the camera wearer, respectively, which have proven descriptive for predicting egocentric actions on their own [33, 66, 86]. In addition, we show that when complementary classification tasks are added during training, performance improves further.

The contributions of this chapter are the following:

- We introduce a Multitask Learning scheme that extends 3D-CNNs [20] and functions with an arbitrary number of output tasks.
- We perform experiments demonstrating that MTL improves on egocentric action classification over singe-task learning (STL) baselines without requiring any additional information at test time, other than the input video.
- We demonstrate with experiments generalization of our MTL scheme to a number of related classification and coordinate estimation tasks that improve egocentric action classification.

In Sec. 5.2 we develop our MTL pipeline for an arbitrary number of tasks. In Sec. 5.3 we document our experiments on two egocentric video datasets. Finally, in Sec. 5.4 we discuss our findings from this chapter.

5.2 Methodology

In Sec. 5.2.1, we detail the process of adapting a network from single to multitask and in Sec. 5.2.2 we describe the output layers with their individual loss functions for the tasks we consider. In Sec. 5.2.3, we discuss the details of the coordinate prediction layer and its application in 3D-CNNs to model the progression of movements through time.

5.2.1 Multitask Network Structure

In Fig. 5.1 we visualize our network architecture for multitask learning. The backbone of the network is a feature extractor (the shared network block), upon which the task-specific output layers are attached. We represent the feature extracting network block with $g(x;\theta)$, where x is an input data point from input space $\mathcal{X}$ and θ are the parameters of g. For each task t, we define an output function $f_t(g(x);\theta_t)$, where θ_t are the parameters of the task-specific layer and $t \in \mathcal{T}$, with $\mathcal{T}$ being the set of tasks. For this work, function g is approximated with a 3D Convolutional Neural Network and the input space $\mathcal{X}$ is defined as a set of RGB images sampled from a video clip.

5.2.2 Task-specific Output Layers

In order to train the network, we formulate the loss function based on the number and types of tasks it encapsulates. We perform MTL with two types of tasks: classification and coordinate regression. For a classification task i, $\mathcal{L}_{cls_i}$ is the categorical cross-entropy loss. For a coordinate regression task j, $\mathcal{L}_{coord_j}$ is defined as the Differentiable Spatial to Numerical Transform (DSNT) loss from [105], explained in Sec. 5.2.3. The full loss function $\mathcal{L}$ is defined as

$$\mathcal{L} = \sum_i \mathcal{L}_{cls_i} + \sum_j \mathcal{L}_{coord_j}. \tag{5.1}$$

During training the value of $\mathcal{L}$ is not propagated through each task-specific layer, but each task layer t produces gradients with respect to its individual loss, hence its parameters θ_t are not affected by the remaining tasks. Finally, the gradients from all the output layers are summed and backpropagated through g. This assumes a

relationship between the tasks and the network input, in the sense that it is possible to acquire an output value from every output layer given the same input signal.

5.2.3 Coordinate prediction

Our approach for predicting coordinates stems from the numerical coordinate regression layer introduced in [105]. It enables a 2D-CNN to output an (x, y) coordinate without using a fully connected layer, thus ensuring spatial invariance in the predicted coordinate [105]. Instead, it relies on an additional convolutional layer that predicts a heatmap Z of shape $m \times n$. The softmax activation is applied on Z, such that $\hat{Z} = \sigma(Z)$, to create a 2D probability distribution, which is passed through the Differentiable Spatial to Numerical Transform (DSNT) layer to become a coordinate.

In DSNT, $\hat{Z}$ is discretized by calculating its Frobenius inner product for each dimension, against two uniformly distributed vectors with values in [-1, 1], shaped $m \times 1$ and $1 \times n$ respectively and copied over their singular dimension (n and m times) to become matrices the shape of $\hat{Z}$. The output value of the Frobenious inner product for each matrix is the respective coordinate value with sub-pixel precision in the range [-1, 1]. This process preserves differentiability through the layer and allows gradient flow from a loss function directly associated with the error in coordinate space instead of the error in heatmap space.

The coordinate loss $\mathcal{L}_{coord}$ is the Euclidean distance between the predicted (c_p) and the expected (c_{gt}) coordinate with an added regularization factor $\lambda = 0.5$ to smooth the gradients around the prediction, *i.e.*,

$$\mathcal{L}_{coord} = \lambda \mathcal{L}_{euc}(c_p, c_{gt}) + (1 - \lambda)\mathcal{L}_{reg}(\hat{Z}), \tag{5.2}$$

where the Euclidean loss is

$$\mathcal{L}_{euc} = \|(c_p, c_{gt})\|_2 \tag{5.3}$$

with $c_p = DSNT(\hat{Z})$ and the regularization loss is

$$\mathcal{L}_{reg}(\hat{Z}) = \mathcal{L}_{JS}(\hat{Z}, c_{gt}) = JS(\hat{Z} \parallel \mathcal{N}(c_{gt}, \sigma^2)) \tag{5.4}$$

based on the Jensen-Shannon divergence.

In order to successfully apply this coordinate regression layer in our setup, we need to account for the output dimensions of the last convolutional layer of the 3D-CNN. In the 2D case that is $B \times C \times m \times n$ where B is the batch size and C the channel

dimension. In the 3D case the output shape extends to $B \times C \times l \times m \times n$ with l the added temporal dimension due to the 3-dimensional input. This leads to having l heatmaps Z as well as l coordinate losses (instead of 1) for an RGB clip. These losses are averaged over the temporal dimension to prevent propagating huge gradients to the rest of the network and introducing bias from the coordinate regression tasks.

5.3 Experiments

We use EPIC-Kitchens [24] and EGTEA Gaze+ [85] for our experiments. Similar to our work in Sec. 4.2.1 we partition the fully annotated training split into custom training and validation splits. Videos from participants 1-29 are used for training (26,375 clips) and 30-31 for validation (2,095 clips). For EGTEA Gaze+ we use the first split as defined by the authors [85] to train and evaluate at the clip level. Further details about the datasets are given in Sec. 2.3.2 and 2.3.3.

Hand locations One of the tasks we consider for MTL is hand coordinate prediction. Similar to using gaze annotations for the gaze estimation task, this task requires a supervision signal for the hand locations on each frame. To accommodate our experiments we use the egocentric hand detection, tracking, and identification algorithm from Chapter 4 to produce hand location information for every video frame. It uses [120] to detect hand bounding boxes, [11] to track them through time, and hand-crafted priors to remove false-positives and identify between left and right hands. We further modify that workflow to track the top right area of the left hand bounding box and the top left area of the right hand to acquire coordinates that more accurately pinpoint the hands instead of the forearms.

5.3.1 Training and Evaluation

For the shared network block of Fig. 5.1 we employ the Multi-Fiber Network (MFNet) [20]. Its architecture contains 3D convolutional layers to capture spatio-temporal information from frame sequences, while using a relatively low number of parameters (10M) and computational resources (11.1 GFLOPS), leading to an efficient training scheme for large video datasets. For all our experiments we use weights pretrained on Kinetics-400 [17] and retrain the full network structure end-to-end on the respective dataset.

We train with a triangular cyclical learning rate (CLR) [136] policy that shifts learning rate from 5×10^{-4} to 5×10^{-3} and back in 20 epochs. For optimization we use Stochastic Gradient Descent with Nesterov momentum (0.9) and weight decay (5×10^{-4}). We input a sequence of 16 frames, randomly scaled to 256×256 and cropped to 224×224. The frames are uniformly sampled from a 32-frame window that starts at a random point of an action video segment and does not exceed its last frame. The batch size is 32 for our setup with two Nvidia 1080Ti GPUs, training lasts for 60 epochs and results are reported for the best performing epoch for the main task (early stopping). To evaluate, we sample uniformly 16 frames from a window of 32, centered around the temporal center of an action segment. We resize them to 256×256 and use the 224×224 center crop as the network input.

5.3.2 Results on EPIC-Kitchens

Our results on EPIC-Kitchens are summarized in Table 5.1. Initially, we train the single-task learning (STL) baseline for the verb task and later combine verbs with nouns and hands as separate tasks. Training for verbs together with hands (V+H) increases Top1 accuracy on verbs to 49.31% (+0.75%) compared to training for verbs alone. Adding nouns (V+N+H) harms verb Top1 accuracy by 1.1% but produces our best performing noun classifier.

In the EPIC-Kitchens literature [24] verb and noun predictions are combined following their individual inference stages and are later synthesized into an action prediction. In our MTL scheme we train for the action task explicitly, *i.e.*, the 2,521 valid verb and noun combinations. Having actions, verbs, and hands for supervision (A+V+H) leads to 49.05% for verbs, improving on the verb STL baseline by 0.48% and additionally using the nouns (A+V+N+H) still improves from verb STL (+0.33%). However, both cases do not improve as much as with only the hands, implying a conflict between the extra tasks. On the other hand, if we consider actions as the main task, the addition of verb, noun, and hand learning tasks will always improve on the action STL baseline reaching 19.29% (+0.81% from A and +0.71% from A+V+H).

5.3.3 Results on EGTEA Gaze+

In Table 5.2 we delineate our results as the network moves from one to multiple tasks in the EGTEA Gaze+ dataset. We establish the action STL baseline (A) at 63.75% Top1 accuracy. Next, we train using additional supervision from verbs and

Tab. 5.1.: Multitask learning results on EPIC-Kitchens. The first column shows the trained tasks for a model: Actions (A), Verbs (V), Nouns (N), and Hands (H). We report Top1 and Top5 accuracy on our validation set. Average class precision and recall are reported for many-hot verbs, nouns, and actions. Many-hot verbs and nouns are the ones having more than 100 instances in our training set. Many-hot actions are the valid combinations of many-hot verbs and nouns with at least one instance in the training set, following [24].

	Top1 Acc. (%)			Top5 Acc. (%)			Avg class Prec. (%)			Avg class Rec. (%)		
Tasks	Actions	Verbs	Nouns	Actions	Verbs	Nouns	Actions	Verbs	Nouns	Actions	Verbs	Nouns
V	-	48.57	-	-	78.32	-	-	34.39	-	-	25.29	-
V + H	-	**49.31**	-	-	**78.80**	-	-	29.85	-	-	25.68	-
V + N + H	-	47.47	**27.60**	-	78.37	**51.19**	-	27.80	21.43	-	23.61	18.80
A	18.48	-	-	36.20	-	-	2.76	-	-	2.67	-	-
A + V + H	18.58	49.05	-	**38.82**	78.75	-	2.89	28.43	-	2.87	23.23	-
A + V + N + H	**19.29**	48.90	27.27	35.91	78.18	47.85	3.25	29.31	22.68	3.04	24.03	17.84

nouns (A+V) and (A+V+N) and reach 67.80% (+4.05%) and 68.00% (+4.25%), respectively. For further experiments we utilize coordinate regression layers to train on gaze points and hand tracks. We see that with either task we improve in both Top1 and mean class accuracy over STL; A+G is 66.59% (+2.84%) and A+H is 67.46% (+3.71%). Further improvements stem from training for all classification tasks together with gaze estimation or hand prediction. A+V+N+G reaches Top1 68.74% (+4.99%) and A+V+N+H is 68.20% (+4.45%). The attempt to combine gaze and hand coordinate regression tasks only with actions shows that the two coordinate tasks are competing to influence the shared representation and have the smallest improvement over the STL baseline with A+G+H Top1 at 66.12% (+2.37%). However, when all the classification and coordinate prediction tasks are present in one model (A+V+N+G+H), we achieve our best Top1 accuracy at 68.99% (+5.24%) and our best mean class accuracy at 61.40% (+6.05% from STL at 55.35%).

Tab. 5.2.: Multitask learning results on EGTEA Gaze+. The first column shows the names of the supervised tasks: Actions (A), Verbs (V), Nouns (N), Gaze (G), and Hands (H). We report Top1, Top5 and mean class accuracy on the first split of the EGTEA Gaze+ test set.

	Top1 Acc. (%)			Top5 Acc. (%)			Mean Cls Acc. (%)		
Tasks	Actions	Verbs	Nouns	Actions	Verbs	Nouns	Actions	Verbs	Nouns
A	63.75	-	-	91.05	-	-	55.35	-	-
A + V	67.80	79.03	-	91.89	99.41	-	59.15	**79.44**	-
A + V + N	68.00	78.98	78.93	91.94	99.31	96.24	59.67	78.24	72.06
A + G	66.59	-	-	91.54	-	-	59.44	-	-
A + H	67.46	-	-	91.99	-	-	59.78	-	-
A + G + H	66.12	-	-	90.54	-	-	58.91	-	-
A + V + N + G	68.74	78.14	**79.13**	91.59	99.41	**96.54**	60.34	79.29	72.03
A + V + N + H	68.20	**79.18**	77.94	**92.24**	**99.51**	96.34	60.13	79.34	71.1
A + V + N + H + G	**68.99**	79.08	79.03	91.74	99.26	96.39	**61.40**	77.40	**72.49**

5.3.4 State-of-the-art Comparison

In Tables 5.3 and 5.4 we compare with the state-of-the-art in action recognition for EPIC-Kitchens and EGTEA Gaze+, respectively. For EPIC, we demonstrate slightly lower but comparable performance to the top methods for the seen (s1) and unseen (s2) test splits, by requiring only a fraction of the input. For example [39] requires RGB and flow at test time and [164] utilizes knowledge from past video segments, in effect having a larger temporal view of the action. However, we still outperform the attention mechanism of [149]. For EGTEA, we test against several methods, for different metrics. Top1 recognition accuracy for the first split at the clip level is reported in [57] (55.63%) and in [150] (62.17%) where we improve by 13.36% and 6.82% respectively. Li *et al.* [85] report 47.71% mean class accuracy on the first split at the clip level (and 53.30% on the video level[1]). Our method depending on the task combination reaches 58.91% up to 61.40% (+11.20% to +13.69% respectively).

For a more elaborate comparison on EGTEA Gaze+, we train the A+V+N+G+H model for splits 2 and 3 and average the Top1 accuracy over all splits. We achieve 65.70% Top1 accuracy which is the highest among the reported values by a margin of 3.84%. For future reference we also report the mean class accuracy averaged over the three splits (57.60%).

Tab. 5.3.: Comparison on action recognition against state-of-the-art methods on EPIC-Kitchens. Our method is consistently close to the best performing, while requiring less information at test time.

	Top1 Acc. (%)			Top5 Acc. (%)		
Method	Actions	Verbs	Nouns	Actions	Verbs	Nouns
Test s1 (Seen kitchens)						
TSN [24]	20.54	48.23	36.71	39.79	84.09	62.32
LSTA [149]	30.33	59.55	38.35	49.97	85.77	61.49
Ours (all tasks)	29.73	56.00	40.15	50.95	87.06	64.07
RU [39]	**33.06**	56.93	43.05	**55.32**	85.68	67.12
LFB [164]	32.70	**60.00**	**45.00**	55.30	**88.40**	**71.80**
Test s2 (Unseen kitchens)						
TSN [24]	10.89	39.40	22.70	25.26	74.29	45.72
LSTA [149]	16.63	47.32	22.16	30.39	77.02	43.15
Ours (all tasks)	17.86	45.99	26.25	35.68	**77.98**	50.19
RU [39]	19.49	43.67	26.77	37.15	73.30	48.28
LFB [164]	**21.20**	**50.90**	**31.50**	**39.40**	77.60	**57.80**

[1]For video level validation the authors split each action segment into equally sized clips and average the action scores for the final prediction.

Tab. 5.4.: Action recognition comparison on EGTEA Gaze+. We compare against the available metrics from each work.

	Split 1		Avg. Splits 1-3	
Method	Top1	Mean Cls	Top1	Mean Cls
Li *et al.* [85]	-	47.71	-	-
MCN [57]	55.63	-	-	-
RU [39]	-	-	60.20	-
EGO-RNN [150]	62.17	-	60.76	-
LSTA [149]	-	-	61.86	-
Ours (all tasks)	**68.99**	**61.40**	**65.70**	**57.60**

An additional interesting scope from EGTEA is gaze estimation. Since a number of our models are able to predict gaze on the input frames, we proceed to evaluate the task with two standard metrics in the literature: Average Angle Error in degrees (AAE) and Area Under the Curve (AUC) [124] following [57]. For evaluation we use only the frames from the clips of the first test split for which after resizing and cropping to 224×224 there is a valid ground truth gaze point in this area, regardless of the gaze type. This leads to the evaluation of 177,292/206,649 (85.79%) frames from 2,022 clips (the remaining frames are not considered). The results are shown in Table 5.5. We discover that gaze estimation techniques which are explicitly designed to model gaze through elaborate attention mechanisms such as [57] achieve lower angular error (-3.11°) although our model (A+G+H) improves over [55] and is very close to [85]. Furthermore, considering AUC, our model is second best to [57] with a -0.06 margin (with the following two methods also being close). The two metrics imply that our method is able to produce gaze predictions that lie in the vicinity of the ground truth (high AUC) but with an angular offset with respect to the exact ground truth gaze position. In Fig. 5.3 we show and qualitatively assess gaze and hand predictions. The images show both the predicted saliency in heatmap form as well as its transformation into a single point per frame for the gaze and each hand coordinate.

Tab. 5.5.: Gaze estimation comparison on EGTEA Gaze+ split 1. AAE lower is better, AUC higher is better. SALICON [55], Li *et al.* [85], DFG [175], and Huang *et al.* [56] are reported from [57].

Method	AAE	AUC
SALICON [55]	11.17	0.881
Ours (A+G+H)	*8.90*	*0.926*
Li *et al.* [85]	8.58	0.87
DFG [175]	6.30	0.923
Huang *et al.* [56]	6.25	0.925
MCN [57]	**5.79**	**0.932**

5.4 Discussion

Our initial objective with MTL is to drive the focus of the network's activation maps towards hand regions and their movements. By training for the hand coordinate task we imply greater importance to these regions and introduce this as a training requirement to the weights of the shared network block via gradient descent. An example of the expected behavior of the network is in Fig. 5.2, where the class activation maps after the last convolutional layer cover a larger area of the visible hands.

The task of gaze prediction is similar to hand detection in that it expects the network to focus on specific regions of the input frames. The difference is that these regions do not necessarily contain the well-structured form of hands, but the salient areas of a scene, which are not predetermined. This limits the ability of region-specific features to become significant making it a dataset- and class-specific quality.

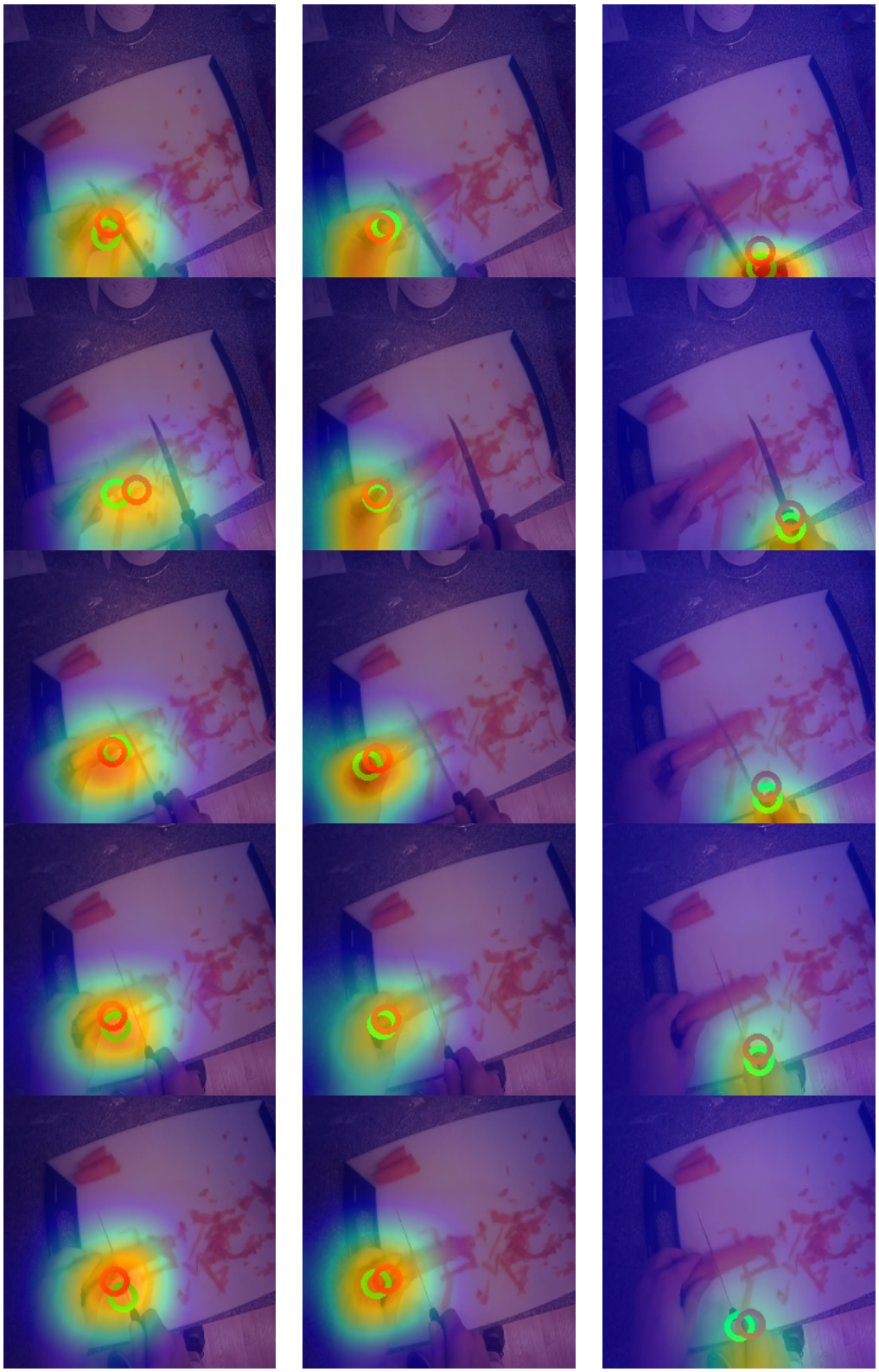
In both datasets, we observe almost consistent improvements over STL with the introduction of hands and other tasks in the training scheme. However, the choice of tasks involves a significant amount of intuition as well as weighing their importance in the loss function. In this work, we use a naive weighing mechanism and consider all tasks equal regardless of the loss they incur. When training multitask models for EPIC-Kitchens we notice high values of loss in the classification tasks, which stem from the class imbalance and the large number of action, verb, and noun classes. These losses initially affect their individual layers, but the backpropagated gradients to the shared weights are also higher, affecting the representation in an unbalanced way. In the EPIC-Kitchens results we see that by adding classification tasks with more classes (such as N at V+N+H, or A at A+V+N+H), we get a worse verb classifier. This is caused by the high losses incurred from the added tasks. In certain cases, they act as regularization but when losses are too high they can increase the training time and even prevent convergence. We believe further research is needed in MTL for video recognition to establish weighing mechanisms such as [127] to obtain a more optimized shared parameter space. For example, our work might benefit from a scheme to balance the losses of the classification tasks, minimizing the impact of the relatively larger losses incurred by the action task.

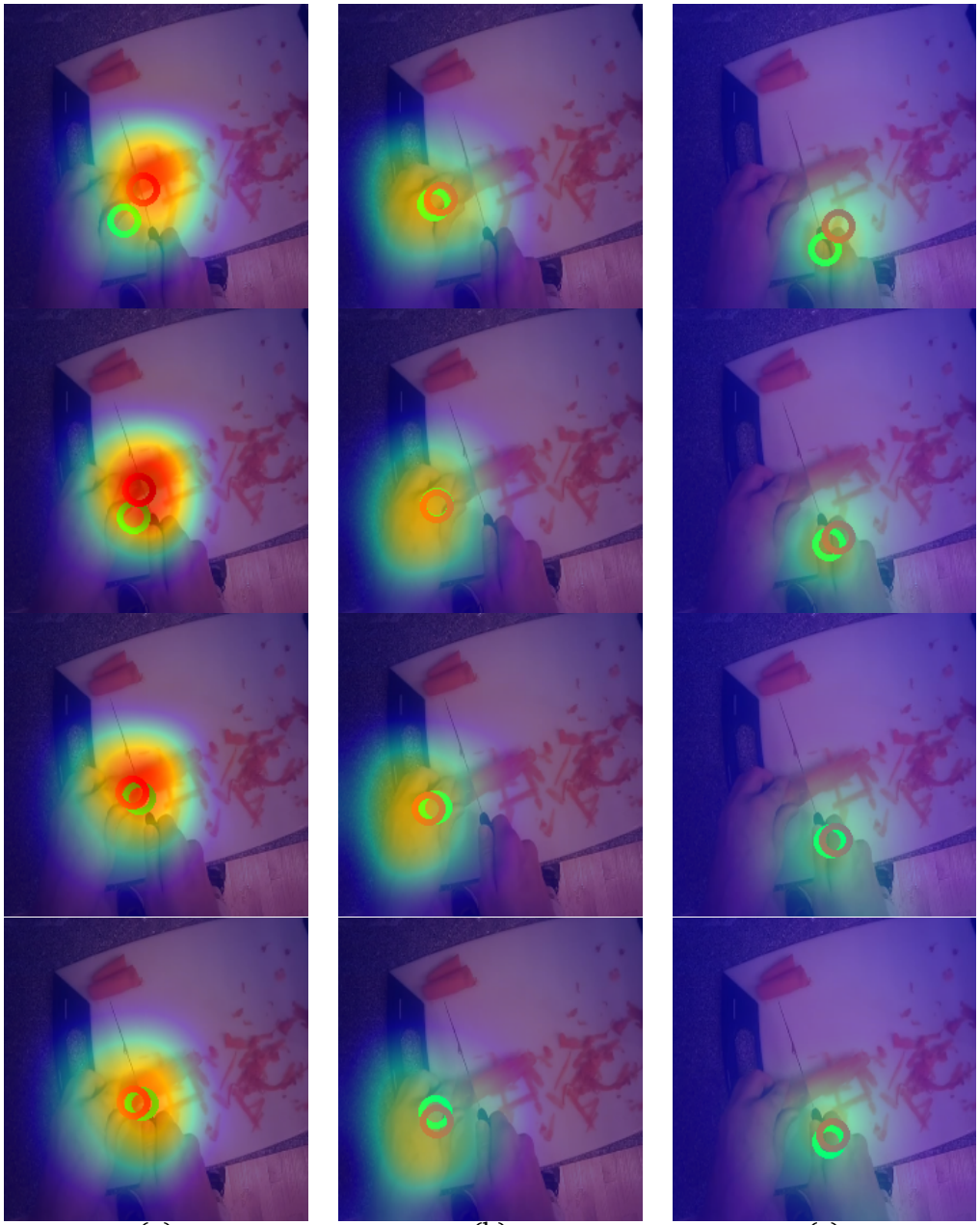
On the other hand, on EGTEA Gaze+, MTL consistently outperforms STL for every task combination. This shows that carefully designing the classification tasks, (*e.g.*, fewer classes, balanced dataset) can be mutually beneficial to all and specifically to the action task we are most interested in. Incorporating hands in training confirms our initial intuition that motions create a fitting side-task to actions, improving

performance. A possible reason for the larger improvement due to hands on EGTEA compared to EPIC is the presence of hand annotations from the former in the training set of the hand detector of Chapter 4. This would result in a more accurate synthetic hand dataset for EGTEA and higher quality hand supervision. Finally, improvements due to gaze validate the connection between actions and gaze [57, 85] also from the perspective of MTL.

A possible pitfall of MTL is the competition among tasks which leads to negative effects on performance. This is possible if certain tasks are incompatible or if the network is not large enough to create a representation that engulfs the different aspects of information required for each one. The former regards a (lack of) conceptual relevance, for example verbs and nouns on EPIC, or structural, for example classification layers operating differently from coordinate regression ones and possibly requiring a distinct representation in earlier layers. The concept of task compatibility has been studied for other domains in [19] concluding that the degree of assistance from an extra task in learning another cannot be fully clear a priori without experimentation. However, the hypotheses of what might help are usually straightforward and the selection of trainable tasks can be treated as an additional hyperparameter.

An example of task incompatibility with respect to actions is when both gaze and hands are used for action recognition (A+G+H) but lead to worse performance than training individually (A+G, A+H). Adding a task may not improve as much as another combination, but it can also reduce the expected baseline performance. In this case, the trade-off is that actions and hands contribute towards an optimal gaze detector (Fig. 5.3). Most importantly, MTL is a straightforward way to enhance the reusability of a model for multiple tasks.





(a) (b) (c)

Fig. 5.3.: (a) Gaze, (b) left hand, (c) right hand coordinates from the A+G+H model. Green circles represent ground truth coordinates and red circles the predicted coordinates. The underlying heatmap shows the 2D probability distribution $\hat{Z}$.

6 Multi-dataset Multitask Learning to Recognize Egocentric Actions

6.1 Introduction

Classification models for egocentric vision tasks such as action recognition are predominantly trained using supervised learning schemes. While action recognition from first-person and third-person videos can be assumed to have a comparable complexity, labeled datasets for the third-person perspective, (*e.g.*, [14, 72, 73, 78, 132, 141]) are typically orders of magnitude larger than egocentric datasets, (*e.g.*, [3, 6, 24, 59, 82, 83, 84, 85, 101, 110, 122, 131, 146, 151]).

While more general egocentric video datasets exist, (*e.g.*, [101, 112, 131, 151]), they focus on longer-term activities such as walking [101], socializing [3, 82, 146], or doing sports [6]. Activities and actions differ in their duration and complexity. An activity's duration is typically in the range of tens of seconds or even minutes and it considers a complex scenario that comprises several actions. The distinction between activities and actions becomes straightforward with an example from the ADL dataset [110] (Table 3.3). Activity 'making coffee' may consist of a sequence of actions such as 'pick-up cup', 'move cup', 'pick-up kettle', 'pour coffee' etc. Recognizing this activity is the combination of detecting the relevant objects, the hand-object interactions and the order they occurred over the course of the video, *i.e.*, a high-level analysis is mandatory, besides the visual features that characterize the scene. Actions on the other hand can be much shorter (as mentioned in Sec. 2.3.2 the mean duration of actions in EPIC-Kitchens is 3.7s, and their recognition requires a more granular analysis over shorter-term video fragments. Egocentric video datasets that address such action recognition tasks are homogeneous in terms of the action domain, the recording environment, and the recorded actors. While there is a steady

Published as:
G. Kapidis, R. Poppe, and R. C. Veltkamp. "Multi-Dataset, Multitask Learning of Egocentric Vision Tasks". IEEE Transactions on Pattern Analysis and Machine Intelligence, Special Issue on Egocentric Perception (2021)

progression in the variation within the datasets that have been introduced over the years, each dataset has a focus on a specific task or application.

ADL was one of the first egocentric video datasets that focused on human activities in indoor environments. Participants performed daily activities such as cooking and cleaning in their homes with annotations of the temporal range of activities, objects used, and the locations in the house [159]. To increase granularity and specialization in cooking activity recognition, the EGTEA datasets were introduced [34, 85] where participants followed narrated recipes for meal preparation in their kitchens. To scale up the dataset size and remove the use of scripts, the EPIC-Kitchens dataset [24] introduced a culturally diverse set of videos with a large variety of actions and interactions with cooking ingredients and kitchen-related objects. Additional modalities such as object presence are predominantly used both during training and at test time. While the performance of some detection tasks such as object detection is impressive, the requirement of additional inputs for testing is a limiting factor.

Obtaining egocentric videos with relevant labels for various tasks is labor-intensive, and there is the need for learning schemes that can reduce overfitting of trained models without requiring more annotated data. In this chapter, we introduce such a scheme that uses annotations from both related tasks and related datasets. We extend the multitask learning (MTL) scheme introduced in Chapter 5 to exploit annotations of related tasks during training, while only video data are required at test time. We base our work on ideas developed in [67], where joint training with related video recognition tasks such as object, hand, and gaze detection have been shown to improve action recognition performance. We investigate the concept of task relatedness [19]. Our premise is that common actions in different datasets such as 'cut' and 'open' are associated by the network and the same neural pathways are reused, producing efficient and robust multi-purpose models. This provides an effective and efficient way to utilize additional training data from diverse sources. We allow for video data from other datasets to be used in the training process and treat the issue of different label sets as extra tasks. Our novel learning scheme is termed Multi-Dataset, Multitask Learning (MD-MTL).

To demonstrate the benefits of our approach, we adapt a 3D-convolutional neural network [20] to include additional task-specific output layers [67] for the tasks of other datasets. In MD-MTL, each epoch consists of the data of the combined training sets, while each batch comprises data randomly chosen out of all datasets, to allow for batch loss calculation that represents the full spectre of available domains. We also experiment with other batch division strategies.

We experiment with combinations of data from EPIC-Kitchens [24], EGTEA Gaze+ [85], and ADL [110] to demonstrate the effectiveness of our multi-dataset, multitask training scheme for egocentric action recognition. Specifically, regarding ADL we investigate the potential improvements on longer term activity recognition performance by utilizing the short-term actions from EPIC and EGTEA. Lastly, we use Charades-EGO [131] to investigate the benefits from associated third-person videos in egocentric action recognition.

The contributions of this chapter are the following:

- We extend Multitask Learning (MTL) to include training data from multiple datasets (MD-MTL) with a simple but effective network modification.
- We introduce a batch formation scheme for on-the-fly association of dataset-specific samples to dataset-specific tasks.
- We demonstrate the improvements of MD-MTL in classification performance for the main action recognition tasks. We also highlight the reuse of the same pathways for related classes across datasets.

In Sec. 6.2 we introduce MD-MTL. In Sec. 6.3 we describe our experiments and in Sec. 6.4 we provide an extended analysis of the task outputs for each dataset. Finally, in Sec. 6.5 we delve into a discussion on the findings of this chapter.

6.2 Methodology

In this section, we describe the extension of a single task network to multitask (MTL) (Sec. 6.2.1), and subsequently describe our process to adapt it to multiple datasets (MD-MTL) (Sec. 6.2.2).

6.2.1 Multitask Network Structure

We adopt the multitask network with task-specific output layers (MTL) from Chapter 5. It comprises a 3D-CNN backbone feature extractor [20] that receives a short video clip and outputs spatio-temporal features after the last convolutional layer. We prefer 3D-CNNs because they can handle motion information from the temporal structure of the video without requiring an additional optical flow input. Recent approaches to capture motion from RGB, *e.g.*, [36, 167], are promising developments

to further acquire temporal motion features but these are out of scope for this work. Fig. 6.1a shows the MTL network with task-specific layers.

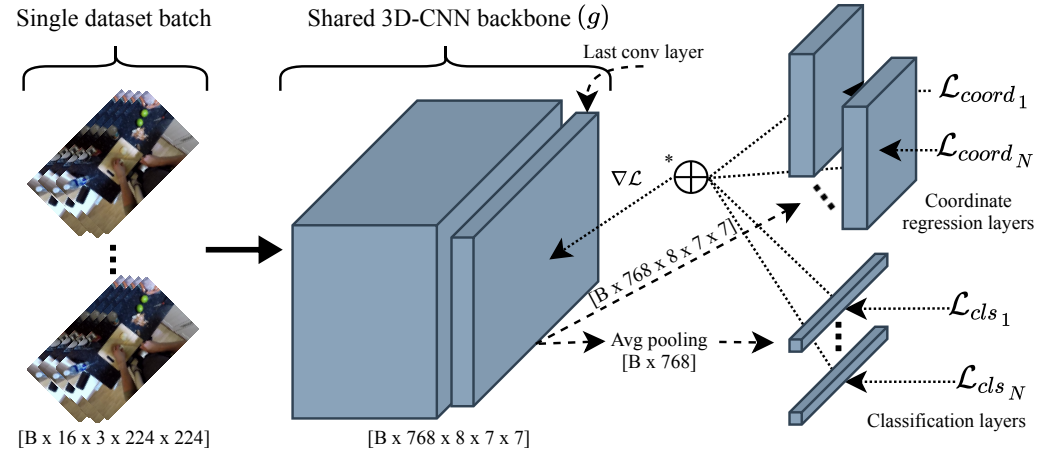
Following Chapter 5, in our MTL setting we define a set of tasks $\mathcal{T}$ with a distinct task-specific output layer for their respective results. Formally, for each task $t \in \mathcal{T}$ we define an output function $f_t(g(x); \theta_t)$, with $g(x; \theta_s)$ the shared block, θ_t the task-specific parameters, θ_s the shared parameters from g and x the network input. Each task-specific layer comprises a distinct loss function designed to accommodate the type of task it represents. In line with Chapter 5, We use classification and coordinate regression tasks. Classification tasks are modeled with a fully connected layer. Their inputs are the activations of $g(x)$, followed by an average pooling operation to reduce the temporal dimension, and their outputs are the per-class probabilities. To train classification tasks we use the categorical cross-entropy loss.

We use coordinate regression tasks in our experiments (see Sec. 6.3) to find egocentric hand positions and gaze estimates. These are implemented with the numerical coordinate regression layer, introduced in [105], to predict a coordinate for every two input frames and extended in Chapter 5 to handle 3D feature volumes as input. The coordinate regression layer begins with a 3D convolution. The 3D output is split along the temporal dimension with each slice Z being passed to a Differential Spatial to Numerical Transform (DSNT) layer [105] to produce a coordinate for each. In the DSNT layer, each slice is passed through a softmax activation to produce a 2D probability distribution $\hat{Z}$ that represents the abstract location. The final (x, y) coordinate is taken as the probability distribution's expectation for each dimension. Following Chapter 5, to train the coordinate regression layer we utilize the DSNT loss which is defined as the Euclidean distance between the predicted (c_p) and the ground truth (c_{gt}) coordinate regularized with the Jensen-Shannon divergence to smooth the gradients around the prediction with a factor $\lambda = 0.5$. Analytically, the DSNT loss function is given in Equation 6.1:

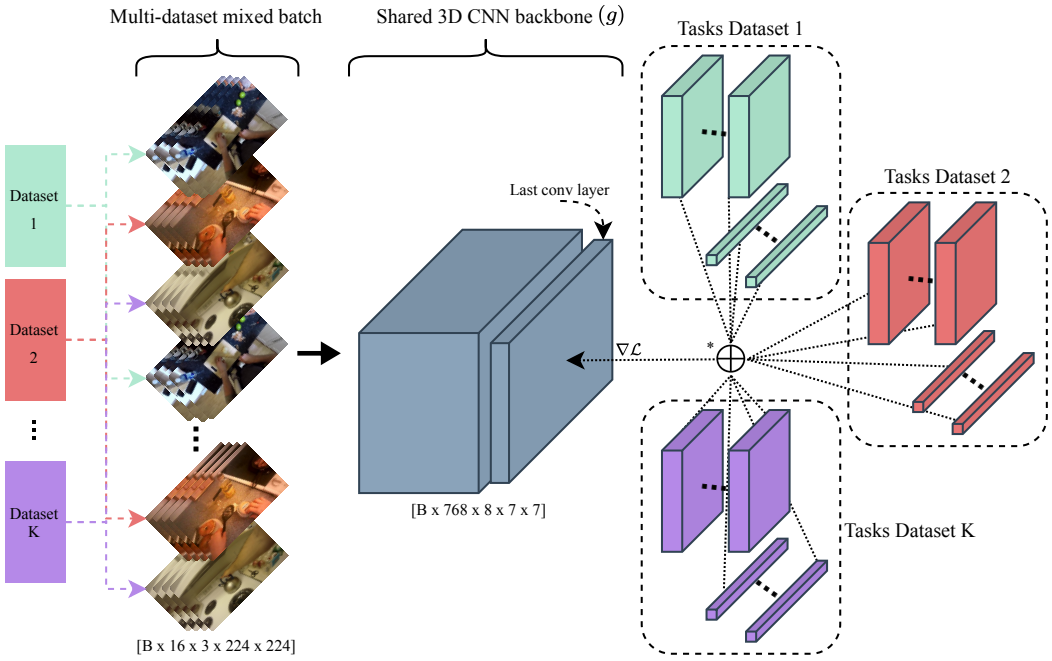
$$\mathcal{L}_{coord} = \lambda \mathcal{L}_{euc}(c_p, c_{gt}) + (1 - \lambda) JS(\hat{Z} \parallel \mathcal{N}(c_{gt}, \sigma^2)). \tag{6.1}$$

6.2.2 Multi-dataset Network Adaptation

Our extension from single- to multi-dataset training (MD-MTL) requires two modifications. The first is to append task-specific layers for the tasks of the additional datasets and the second is to adapt the training process to accommodate for the induced variation in the mixed training batches.



(a) Multitask network with task-specific output layers (* accumulating gradients from all tasks).



(b) Multitask network adapted for multiple datasets (* accumulating gradients from all tasks across datasets within a single batch.

Fig. 6.1.: We adapt the MTL network structure from Chapter 5 to accommodate the tasks from a range of datasets within a single network. In (a) the network combines task-specific output layers by aggregating the gradients from each output. In (b) we extend the structure by further attaching task-specific layers for the additional tasks in the new datasets.

We handle the additional tasks in the way we would treat any added task from the initial dataset, *i.e.*, we add task-specific layers to the shared network block that produce a distinct output, independent of the other tasks. Similar to the single-dataset MTL network, each output layer in MD-MTL utilizes its own loss function for training. A visualization of this extension is given in Fig. 6.1b.

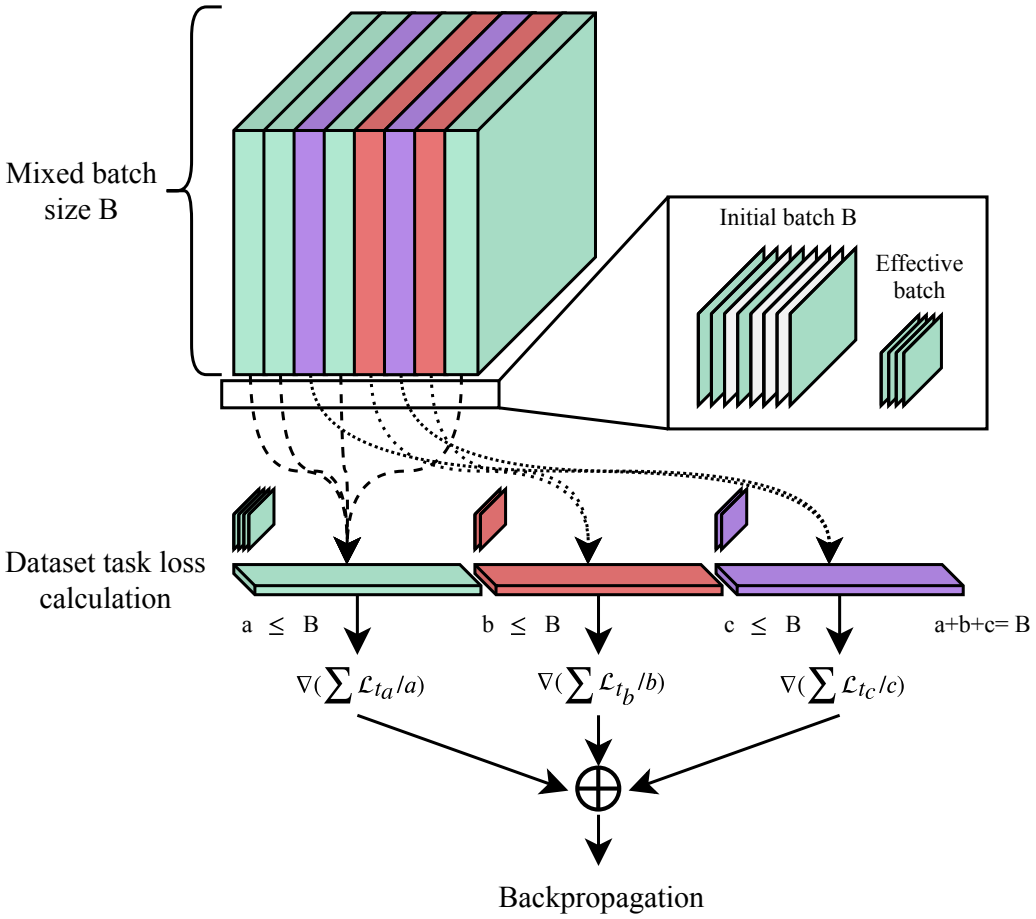


Fig. 6.2.: Mixed-batch loss approximation. A batch is subsampled for each task. The loss from each task layer is averaged over its dataset's samples. Task-specific losses are passed through their respective layer.

We need to accommodate for the fact that no samples within a mixed batch have labels associated with all the available tasks, since each subset corresponds to a distinct dataset. Our strategy is to leverage the process of averaging the loss across a batch, which is commonly employed when training neural networks with mini-batches.

The premise is that for a batch of size B the loss is calculated B times and averaged to provide an approximation of a B-sized mini-batch. Loss averaging is not possible when batches assimilate different datasets and tasks. In this case, we subsample

each batch based on its origin dataset i and produce an effective batch per dataset of size b_i. Then, we calculate each task's loss for the appropriate samples only, zeroing out those that were forwarded through a task-specific layer for which there is no available label. The losses are then averaged over the size of the effective batch and gradients for each task-specific layer are calculated with respect to the dataset tasks' losses. Once the per-task gradient approximation is handled, they are accumulated before being backpropagated through g. Consequently, all tasks are contributing into training the shared network block regardless of the number of samples that were taken from each dataset. We visualize this process in Fig. 6.2.

Multi-dataset training with mixed batches (instead of interleaved batches or alternating datasets sequentially) allows the network to gather gradients from samples representing the full range of available datasets in a single training step. Hence, the direction of the gradient will not be representative of one dataset as in single dataset training. Instead, it will be biased by all datasets in a ratio defined by the sampling process during batch formulation. We permute all datasets and allow the imbalance to be induced in the network. Due to the similarities of datasets in our experiments, we expect mixed batches to contain complementary information and prevent divergence in training. Indeed, we see in Sec. 6.3.1 evidence of improved performance when the datasets are related and deterioration when they represent a different domain. In Sec. 6.4.1 we visualize and discuss the progression of the training losses.

6.3 Experiments

First, we discuss the datasets we experimented with and the training and evaluation settings. In Sec. 6.3.1 we analyze the experiments with egocentric datasets and in Sec. 6.3.3 we delve into a task mapping scheme to capitalize on the semantic class relationships. In Sec. 6.3.2 and 6.3.4 we analyze the mechanics of MD-MTL models to demonstrate the correlations across tasks from different datasets and in Sec. 6.3.5 we focus on the extension for datasets between first- and third-person vision. In Sec. 6.3.6 we experiment with alternative batch formation strategies. Finally, in Sec. 6.3.7 we provide a comparison with the state-of-the-art on EPIC-Kitchens and EGTEA Gaze+.

Datasets We design multi-dataset experiments on egocentric video datasets EPIC-Kitchens [24], EGTEA Gaze+ [85], and ADL [110], all of which capture actions or activities performed in homes from the first-person perspective. EGTEA Gaze+

consists of scripted meal preparation activities, whereas EPIC promotes action variability by encouraging participants to behave consistently to their routines. Videos from both datasets take place in kitchens, ensuring homogeneous locations and both consist of specialized and related sets of short duration actions such as 'open', 'close', and 'cut'. ADL is less specific in terms of environments and actions, capturing a predefined set of daily living activities occurring throughout the participants' homes, performed in an unscripted manner. These annotations represent temporally longer activities such as 'washing dishes' or 'watching tv', which makes it harder to represent the whole activity in the short video segments that are used as input to the network. Hence, content-wise, EPIC and EGTEA are suitable candidates for our task- and dataset-relatedness experiments in order to estimate the possible benefits of joint training. On the other hand, the more varied context of ADL allows us to investigate whether our multi-dataset training approach can adapt to a more diverse domain within a single model.

Furthermore, we perform experiments on the Charades-EGO [131] dataset. It comprises a joint collection of first- and third-person videos. For each third-person video there is an associated egocentric one, recorded by the same participant for the same activities and environment. This allows researchers to model the association between the two video perspectives. Our aim is not to capture the inter-video associations but to examine if a model trained on contrasting perspectives can be efficiently applied to both, simultaneously. Table 6.1 lists the datasets and their characteristics.

Tab. 6.1.: List of datasets and their characteristics. We emphasize on the sizes of the classification tasks (fpv: first-person videos, 3rd: third-person videos).

Name	ADL	EGTEA Gaze+	EPIC-Kitchens	Charades-EGO
Videos	fpv	fpv	fpv	fpv/3rd
#Participants	20	32	32	112
Scripted	partially	yes	no	yes
Labels				
#Actions	18	106	2513	157
#Verbs	-	19	125	33
#Nouns	-	53	352	38
#Locations	8 [159]	kitchen	kitchen	16
Other	objects	gaze, recipes, hand segmentations	objects, narrations	narrations

Following Chapter 5 we also leverage hand location predictions. They have been found to improve classification performance when included as additional tasks in a multitask setting, due to the implicit focus on the salient regions. For the annotations, we synthesize the left and right hand location coordinates for each frame of

ADL, EGTEA, and EPIC-Kitchens using the hand detection algorithm presented in Chapter 4.

Training and evaluation settings For all experiments we use a Multi-Fiber Network (MFNet) [20] pretrained on Kinetics-400 [73] as the backbone feature extractor. It acts as the initial structure upon which task-specific layers are attached. Our choice is justified by the fact that it comprises a 3D-CNN structure, able to capture spatio-temporal information without the need for an optical flow stream, with a significantly lower number of parameters ($\sim$8M), for depth similar to a 3D ResNet-50 ($\sim$47M). We train all models with triangular cyclical learning rate (CLR) [136] oscillating from 0.0005 to 0.005 and back within 20 epochs. Our training cycle is repeated three times, (*i.e.*, 60 epochs) unless otherwise stated. We use stochastic gradient descent for optimization, with Nesterov momentum (0.9) and weight decay (0.0005). The input for training is a sequence of 16 frames uniformly sampled from a 32-frame window randomly chosen to represent the action segment for an epoch. The selected frames are scaled to 256×256 and randomly cropped to 224×224. Additionally, we perform color augmentations and flip the sequence horizontally with a 50% chance. Even though it is counter-intuitive to train a hand detector that identifies left and right hands with random video flipping, early experiments showed that it does not affect hand estimation. Lastly, we use a batch size of 32 for both single and multi-dataset experiments, for comparison purposes.

To evaluate an action segment, we select 16 frames from a 32-frame window around the clip's temporal center. We resize to 256×256 and use the 224×224 center crop. The indicated performances are derived from the best performing weights for the action task, acquired with early stopping.

6.3.1 Multi-dataset Experiments on EPIC, EGTEA, and ADL

Single dataset baselines In the single dataset (SD) setup in Chapter 5 the trainable tasks for EPIC are action, verb, and noun classification and left/right hand location prediction (E_{ALL}). For EGTEA, gaze estimation is added to the set of trainable tasks (G_{ALL}). ADL annotations describe long-term activities with the addition of indoor locations from [159] (A_{ALL}). For EPIC, training and validation are performed on the custom train/val splits from Chapter 5, namely 26,375 action segments from participants 1-29 are used for training and the remaining 2,095 for validation, with the exception of videos withheld by the dataset authors for testing. The latter denote scenarios on seen (S1) and unseen (S2) kitchens. S1 consists of videos from participants that also have a number of videos in the training set, whereas in S2 all

participant videos are excluded from the training set. S1 and S2 are evaluated on the EPIC-Kitchens server. On EGTEA we use the first split provided by the dataset authors which consists of 8,299 training and 2,022 validation segments and for ADL we train on the videos of participants 1-6 (111 clips) and validate for participants 7-20 (198 clips). We report the SD baselines in Table 6.2.

Tab. 6.2.: SD-MTL Top1/Top5 accuracy (%) for actions (A), verbs (V) and nouns (N) for EPIC and EGTEA (reported from Chapter 5) and for activities (A) and locations (L) for ADL for the best performing weights on (A).

Model	Top1 (A-V-N/A-L)			Top5 (A-V-N/A-L)		
E_{ALL} (EPIC)	19.29	48.90	27.27	35.39	78.18	47.85
G_{ALL} (EGTEA)	68.99	79.08	79.03	91.74	99.26	96.39
A_{ALL} (ADL)	64.65	72.22		88.38	96.97	

6.3.1.1. EPIC-Kitchens Analysis

We now turn to multi-dataset learning (MD-MTL). We incrementally add new datasets and their tasks to be trained alongside EPIC. The multi-dataset (MD) experiments are named after the included tasks, so E_{ALL}+G_{ALL} contains all tasks of the SD EPIC experiment (E_{ALL}) and all tasks from the SD EGTEA experiment (G_{ALL}). We also perform an MD experiment only on the action tasks for the two datasets (E_A+G_A) to show the effect of the missing classification and coordinate regression tasks in the MD-MTL setting. In Table 6.3 we compare models containing EPIC-Kitchens in the training set.

Tab. 6.3.: EPIC-Kitchens, EGTEA Gaze+, and ADL task combinations. For EPIC, We report Top1/Top5 (%) action classification performance on the validation set and Top1 on the S1 and S2 test sets.

Tasks	Top1	Top5	Top1 S1	Top1 S2
E_{ALL}	19.29	35.91	**29.73**	**17.86**
E_A+G_A	18.15	35.93	24.35	17.04
E_{ALL}+G_{ALL}	**19.69**	**36.68**	26.69	17.17
E_{ALL}+G_{ALL}+A_{ALL}	18.29	34.15	24.17	15.84

$\mathbf{E_A + G_A}$ For this experiment we trained only on the 2,513 and 125 action classes of EPIC and EGTEA, respectively. We achieve a similar level of overfit on the validation set (18.15%) but results on both test sets are below the SD baselines (-5.38% and -0.82%), especially for S1. This highlights the importance of the additional tasks to regularize training and enhance the information acquired by the network when they are present, verifying our findings from Chapter 5 about the usefulness of MTL, also in the MD setting.

$\mathbf{E_{ALL}} + \mathbf{G_{ALL}}$ We proceed to integrate actions, verbs, nouns, and hands from EPIC and actions, verbs, nouns, hands, and gaze from EGTEA. The additional tasks offer a noticeable improvement on action classification for EPIC over the SD baseline on the validation set (+0.40%). This shows that the network is able to fit both training sets simultaneously and that there is potential benefit from our approach if applied on a larger scale. However, we also observe a decline in test set S1 performance (-3.04%). We highlight that performance on S2 is not as affected as in S1 (-0.69%). The reason is that the additional tasks from EGTEA prohibit the network from overfitting on EPIC, resulting in a larger performance drop on the seen kitchens. The model's generalization capability to unseen data is less affected, manifesting relatively robust results on S2.

$\mathbf{E_{ALL}} + \mathbf{G_{ALL}} + \mathbf{A_{ALL}}$ With the addition of the ADL action and location tasks we reach the limit of the learning capability of our model. The domain shift that occurs from the long unstructured activity videos prohibits convergence to the same minimum for EPIC (-1.00%). Thus, test performance also drops.

6.3.1.2. EGTEA Gaze+ Analysis

We now evaluate the EGTEA tasks of the previous models. Table 6.4 summarizes the results on the action task.

Tab. 6.4.: EPIC-Kitchens, EGTEA Gaze+, and ADL task combinations. For EGTEA, we report Top1/Top5 (%) and mean class accuracy (%) for tha action classification task on test split 1.

Tasks	Top1	Top5	Mean cls acc.
G_{ALL}	68.99	91.74	61.40
E_A+G_A	69.78	**93.37**	62.31
E_{ALL}+G_{ALL}	**70.38**	93.08	**62.61**
E_{ALL}+G_{ALL}+A_{ALL}	69.34	92.63	60.87

$\mathbf{E_A} + \mathbf{G_A}$ In this experiment we train only on the EGTEA and EPIC action tasks. Performance improves from the SD baseline (+0.79% Top1, +0.91% mean class accuracy). This already shows the benefit of using MD-MTL. We are improving on EGTEA without adding data specifically for it, but only train jointly with a task from a different dataset.

$\mathbf{E_{ALL}} + \mathbf{G_{ALL}}$ Similar to EPIC, using all available classification tasks, together with the coordinate regression layers further improves performance. It is +1.39% in Top1 and +1.21% in mean class accuracy up from the SD baseline and +0.60% and +0.29%, respectively, from E_A+G_A. This is another demonstration of the benefits

from using MTL to utilize not only the additional relevant data, but all the learnable tasks.

$\mathbf{E_{ALL} + G_{ALL} + A_{ALL}}$ Adding data and tasks from the ADL dataset worsens action classification performance on the EGTEA tasks. Since EGTEA has greater room for improvement, the decline due to ADL is not as strong as for the EPIC tasks and the SD baseline is still surpassed (+0.35% Top1). However, the effects of the domain shift are evident. The training loss for the action task is higher (Fig. 6.5b) illustrating the difficulty to assimilate actions from the highly variable locations of ADL with the kitchen environments of EGTEA.

6.3.1.3. ADL Analysis

To train on ADL (E_{ALL}+G_{ALL}+A_{ALL}) we add one more learning cycle to the model and train for 80 epochs, to accommodate for the diverse distribution of the ADL dataset. Results on ADL are presented in Table 6.5.

Tab. 6.5.: EPIC-Kitchens, EGTEA Gaze+, and ADL task combinations. For ADL, we report Top1/Top5 (%) and mean class accuracy for the activity classification performance on the validation set.

Tasks	Top1	Top5	Mean cls acc.
A_{ALL}	**64.65**	**88.38**	**56.10**
E_{ALL}+G_{ALL}+A_{ALL}	58.08	86.87	43.61

$\mathbf{E_{ALL} + G_{ALL} + A_{ALL}}$ Following the results on EPIC and EGTEA, the three-dataset model is unable to reach the single dataset baseline of ADL. This result verifies the previous conclusion that additional datasets without a related data distribution can hurt performance. Further analysis of the ADL results is provided in Sec. 6.4.4.

6.3.2 Weight Correlations

In this section, we analyze the learned classification weights in MD networks. We measure the correlations between weights for the task pairs of actions, verbs, and nouns. We find that positive correlations arise in the classification weights across tasks for classes with similar semantic interpretations. This is an important find that demonstrates the ability of the network to capitalize on the relationships of the data without additional supervision. We highlight some examples in Fig. 6.3. We show correlations for classes with the same name, *e.g.*, 'take' in EGTEA with 'take' in EPIC ($r = 0.52$), but also on classes with similar semantic meaning, *e.g.*, 'tomato' in EGTEA

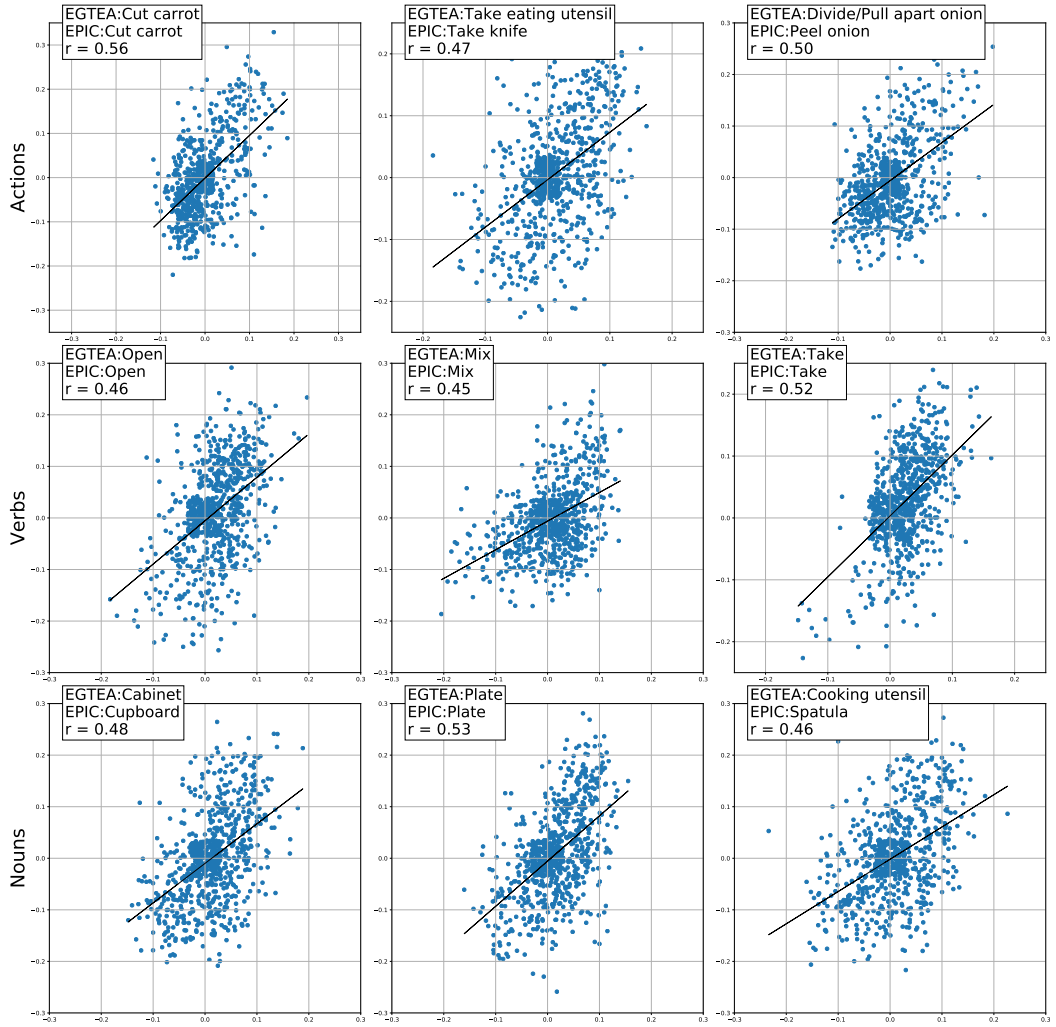


Fig. 6.3.: Correlations for classification weights across tasks in multi-dataset model E_{ALL}+G_{ALL}. Zoom-in for best view.

correlates with 'heart' in EPIC ($r = 0.43$) which refers to a tomato's interior, with second best the correlation with the actual 'tomato' class ($r = 0.38$). Correlation values are higher across action tasks, possibly due to their stricter nature in having to associate both the correct verb and the correct noun class. For example, the verb and noun constituents for 'divide/pull apart onion' of EGTEA correlate with 'peel' and 'onion' in EPIC with $r = 0.26$ and 0.37 respectively, whereas the correlation with action 'peel onion' is $r = 0.50$. This means that the model is more certain about the combination of features it requires when classifying a full action class instead of having to assess it as the union of a verb and a noun. In the following section, we investigate a way to further exploit the associative ability of the network by mapping these classes into the same task.

6.3.3 EPIC & EGTEA with Task Mapping

In many cases, the datasets have partly overlapping label sets for some tasks. In this experiment we reduce the output layers of the network by mapping similar tasks across datasets. We combine the verb and noun classification tasks of EPIC and EGTEA and the hand coordinate layers. We leave the action layers and the gaze unchanged. Our aim is to connect the verb and noun tasks as much as possible while training the action tasks independently. This effort resembles the merged labels technique in [108]. Our approach differs in that we manually map the semantically similar verb and noun classes of EGTEA to EPIC since the majority of its labels are identical or synonyms. There are rare cases where an EPIC label needs to be assigned to multiple EGTEA labels. For example, verb classes 'wash' and 'clean/wipe' are both assigned to EPIC's 'wash' and noun classes that represent containers such as 'tomato container' and 'bread container' are assigned to 'package'. This task combination scheme is less naive compared to our earlier MD approach. The downsides are that we are not able to properly evaluate the verb and noun tasks of EGTEA due to the many-to-one class assignments and that an almost direct mapping across tasks is not always feasible. The task mapping model is trained for 80 epochs (referred to as Verb-Noun Mapping - VN Mapping).

Verb-Noun Mapping results for EPIC are presented in Table 6.6. Action recognition performance is similar to the naive MD approach but with a significant increase in verb and noun classification as well as in Top1 on the EPIC test sets. In fact, with task mapping, the model is able to generalize as well as with the SD model on the S2 test set. This improvement shows that MD-MTL has an even greater potential when secondary tasks of the datasets can be combined explicitly.

Tab. 6.6.: Mapping EGTEA verb-noun tasks on EPIC. For EPIC, we report Top1 (%) action (A), verb (V), and noun (N) accuracy on the validation set and Top1 for actions on the S1-S2 test sets.

Model	A	V	N	S1 A	S2 A
E_{ALL}+G_{ALL}	**19.69**	45.99	25.65	26.69	17.17
Verb-Noun Mapping	19.68	**48.33**	**28.32**	**28.10**	**17.86**

Task mapping also proves beneficial for the action recognition task of EGTEA as shown in Table 6.7. Verb-Noun Mapping is +0.99% from the previous best (E_{ALL}+G_{ALL}: 70.38%) and +2.38% from the SD baseline (G_{ALL}: 68.99%).

Next, we present an additional experiment using transfer learning for SD EGTEA. We use the weights from the SD EPIC model E_{ALL} for the pretrained network state. It improves +1.09% from the SD model pretrained on Kinetics-400, but is still lower

than both naive MD (-0.30%) and MD with task mapping (-1.29%). This shows that MD-MTL networks can capitalize on the additional data advantageously over transfer learning, while being able to keep the tasks from the initial dataset and improve their performance.

Tab. 6.7.: Mapping EGTEA verb-noun tasks on EPIC. For EGTEA, we report Top1/Top5 and mean class accuracy (%) for actions on test split 1.

Model	Top1	Top5	Mean cls acc.
G_{ALL}	68.99	91.74	61.40
G_{ALL} pretrained on EPIC	70.08	92.63	**62.66**
E_{ALL}+G_{ALL}	70.38	**93.08**	62.61
Verb-Noun Mapping	**71.37**	92.78	62.23

6.3.4 Task Affinities

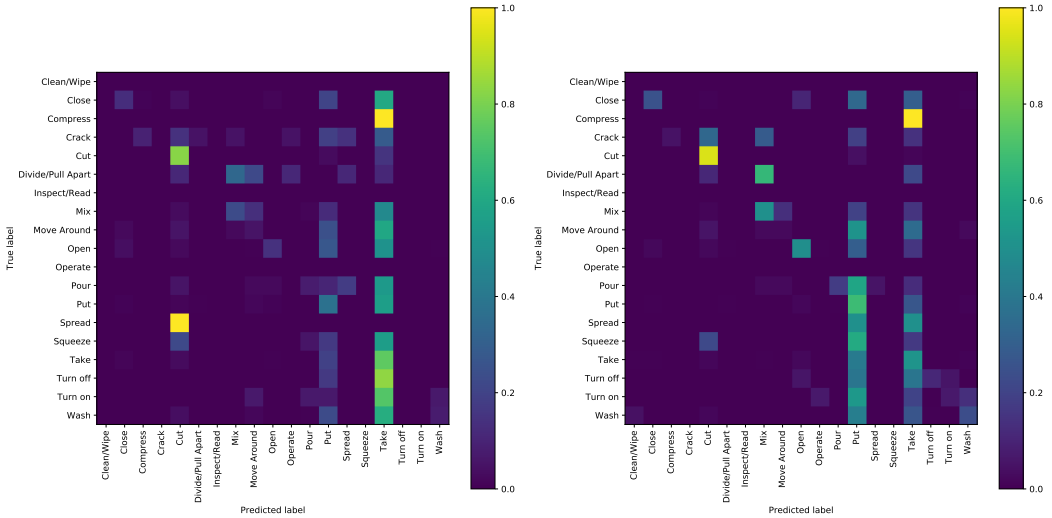
The task mapping approach from Sec. 6.3.3 enhances the correlations across actions, while fixing some inaccurate cases of the previous model. For example, correlation for the 'cut carrot' action increases from $r = 0.56$ to $r = 0.66$ and for 'peel onion' from $r = 0.50$ to $r = 0.54$. Notably, for the latter, the second best correlated action to 'divide/pull apart onion' from the first model is 'peel potato' with $r = 0.44$ which drops to $r = 0.37$. This suggests that the model is now better able to tell apart the two objects.

To further demonstrate the correlated outputs we compare the performance of the EGTEA SD model (G_{ALL}) against the EGTEA verb and noun tasks of the E_{ALL}+G_{ALL} MD model on the EPIC validation split for the samples that comprise mapped classes. This corresponds to 1,677 samples for verbs and 1,107 for nouns. Table 6.8 shows Top1 and mean class accuracy for the mapped verbs and nouns. The improvement of the MD model is consistent over SD, achieving +12.46% and +7.16% on the two metrics for verbs and +16.35% and +5.98% for nouns. This increase further establishes the generalization ability of MD-MTL for samples that do not belong in the data distribution for which the tasks are trained for.

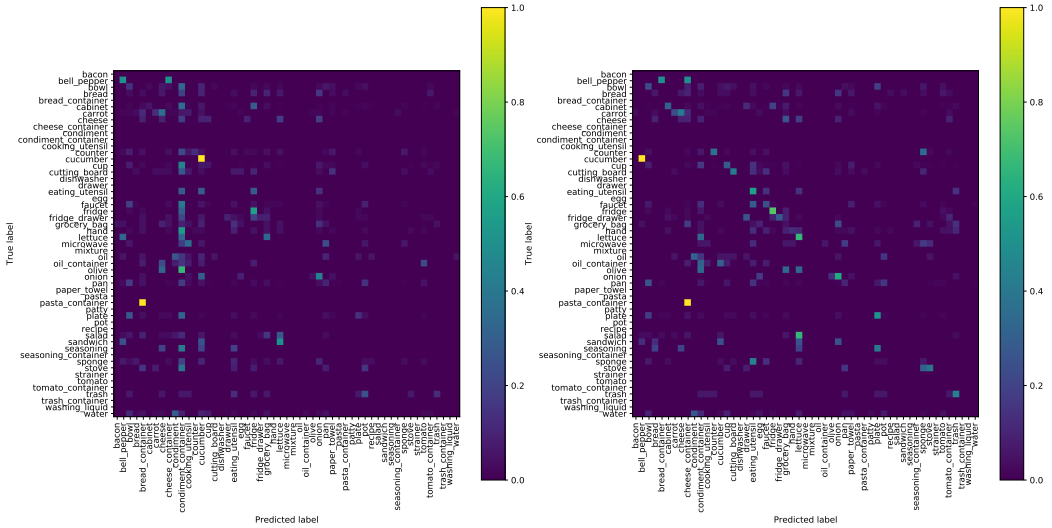
Finally, in Fig. 6.4 we visualize the normalized confusion matrices for these experiments. In Fig. 6.4a we observe fewer errors for verbs such as 'turn on' and 'turn off' and the performance of highly represented classes such as 'cut', 'open' and 'close' increases. Similarly for nouns, in Fig. 6.4b, we see that the SD model (left) tends to classify a number of samples as 'condiment container' which is largely fixed in the MD case (right). Generally, most noun classes have significant improvements.

Tab. 6.8.: Comparison between SD and MD-MTL on the mapped verbs and nouns. Evaluating the EGTEA tasks on the EPIC validation split.

	Mapped Verbs		**Mapped Nouns**	
Model	Top1 (%)	Mean cls acc. (%)	Top1 (%)	Mean cls acc. (%)
G_{ALL}	32.68	13.98	9.40	6.86
E_{ALL}+G_{ALL}	**45.14**	**21.14**	**25.75**	**12.84**



(a) Mapped verb confusion matrices; left EGTEA SD, right EGTEA MD.



(b) Mapped noun confusion matrices; left EGTEA SD, right EGTEA MD.

Fig. 6.4.: Confusion matrices for mapped verbs (a) and mapped nouns (b) from the EGTEA tasks on the EPIC validation split. Zoom-in for best view.

6.3.5 Multi-dataset Experiments on Charades-EGO

We perform experiments on Charades-EGO to explore the associative ability of tasks when applied to data from different viewing perspectives and the potential for performance improvements owing to the MD-MTL setting. We split the dataset into its first- and third-person constituents and treat them as two separate datasets. Consequently, we have two sub-datasets, CHAREGO1 and CHAREGO3, with the same classification tasks. We produce action segments from the video level annotations. This results in 33,099/9,148 action segments for CHAREGO1 and 34,269/9,386 for CHAREGO3 for training and validation, respectively. In Table 6.9 we report video level mean Average Precision (mAP) following [131] and Top1/Top5 accuracy for the action recognition task. We train three models in total. An SD model for CHAREGO1 for actions, verbs, and nouns ($C1_{ALL}$), an SD model for CHAREGO3 for the same tasks ($C3_{ALL}$), and the MD combination with both sets of tasks ($C1_{ALL}$+$C3_{ALL}$).

Tab. 6.9.: Action recognition performance on CHAREGO1 and CHAREGO3, the first- and third-person splits of Charades-EGO, respectively. SD models are trained on all tasks (actions, verbs, nouns) of their splits. The MD model is trained on the combination of the tasks of both splits. Results in %.

	Validation on charego1			**Validation on charego3**		
Model	Top1	Top5	mAP	Top1	Top5	mAP
$C1_{ALL}$ (SD)	**7.05**	24.21	21.90	3.55	14.70	12.30
$C3_{ALL}$ (SD)	3.61	15.40	14.70	**8.15**	**27.02**	**20.40**
$C1_{ALL}$ (MD)	7.01	**24.69**	**22.10**	6.79	22.85	18.20
$C3_{ALL}$ (MD)	5.81	21.75	20.10	8.12	26.04	20.00

Validation on CHAREGO1 shows that MD training provides a marginal improvement over the SD baseline on the video level mAP. This shows the benefit to the first-person tasks when using the third-person videos to train their distinct tasks in the MD setting. An interesting insight arises from evaluating on the first-person data using the respective $C3_{ALL}$ tasks of the MD model. Recognition performance is worse when compared to the egocentric tasks; however, it is significantly higher from the CHAREGO3 SD model. This shows that it learns to associate the internal representations of classes that co-exist in different tasks and reuses them across perspectives (confirming the findings of Sec. 6.3.2 and 6.3.4 also in this setting).

Similar insights can be inferred from the results of the third-person video split of Charades-EGO. In this experiment, the SD model exhibits marginally better mAP than the MD model, but the correlation property across tasks of different perspectives is still present. The first-person tasks of $C1_{ALL}$+$C3_{ALL}$ have +5.9% higher mAP from the SD $C1_{ALL}$ model when evaluated on CHAREGO3.

6.3.6 Batch Formation Strategies

The mixed batch (MB) formation strategy described in Sec. 6.2.2 is not the only way to present data to the MD-MTL network. To further demonstrate the ability of our batch formation strategy to allow optimal generalization across datasets, we compare against two alternative strategies: interleaved batches (IB) and interleaved datasets (ID). In interleaved batches, in every iteration a dataset is selected at random and the input to the network consists of data only from this dataset. In interleaved datasets, batch composition is the same, but each dataset's training set is fully processed before data from the remaining datasets are seen. In either case, the network sees the complete training set of every dataset per epoch. We experiment on the E_{ALL}+G_{ALL} tasks for every batch strategy, using the same training hyperparameters defined in Sec. 6.3. In Table 6.10 we summarize our results.

Tab. 6.10.: Comparison across batch formation strategies mixed (MB), interleaved batches (IB), and interleaved datasets (ID) on the task combinations of EPIC and EGTEA.

(a) Results on EPIC-Kitchens.

	Top1 (%)			Top5 (%)		
Strategy	Actions	Verbs	Nouns	Actions	Verbs	Nouns
MB	19.69	45.99	25.65	36.68	78.37	50.67
IB	**20.11**	47.76	**29.99**	**37.78**	**78.84**	**51.24**
ID	17.91	**48.42**	23.26	33.57	78.18	45.94

(b) Results on EGTEA Gaze+.

	Top1 (%)			Mean cls acc. (%)		
Strategy	Actions	Verbs	Nouns	Actions	Verbs	Nouns
MB	**70.38**	80.57	**79.03**	**62.61**	80.02	**73.55**
IB	65.43	79.72	74.83	55.31	77.21	65.95
ID	69.68	**80.86**	78.64	61.31	**81.59**	72.13

The three strategies have different effects on performance. Interleaved batches outperform mixed batches on EPIC, albeit with a strong performance drop for EGTEA. A possible reason is that the size difference of the datasets (the training set of EPIC is almost three times larger than that of EGTEA) does not allow the network to equally capture fine-grained features from EGTEA. When using the interleaved datasets strategy, we see a significant performance drop for EPIC, with EGTEA being more robust. This is the result of the order with which datasets are seen on every epoch. In our ID experiment, the training set of EPIC is always seen first and EGTEA follows in every epoch. Information that is acquired in the beginning of an epoch is partly "unlearned" when the second dataset is seen. Mixed batches (MB) appear to perform somewhat more consistently. However, the modest differences between

the strategies suggest that MD-MTL performs favorably over single-dataset MTL, independent of the choice of batch formation strategy.

6.3.7 State-of-the-art Comparison

EPIC-Kitchens In Table 6.11 we compare against the state-of-the-art on the S1 and S2 test sets of EPIC-Kitchens. Our method has competitive performance; however, a number of methods have improved accuracy. One reason is the additional input data that most of these methods employ. For example, the top performing approach [26] utilizes a much larger network (118M parameters) and is pretrained on a video dataset about 3k times larger than Kinetics-400 (IG-Kinetics-65M). Interestingly, with Kinetics-400 pretraining on a network eight times larger than ours (R(2+1)D-34, 64M parameters) they perform -1.30% lower on S1 and -1.06% on S2 Top1 actions. Furthermore, a number of methods include optical flow, object and audio input streams which tend to leverage separate networks for each modality. We highlight the work of Kazakos *et al.* [74] which outperforms us with their full model but when only the RGB stream is utilized we show a +7.75% improvement. The remaining approaches do not offer an ablation with only the RGB stream therefore we cannot compare directly. We note the Gate-Shift Networks introduced in [148] that only use RGB input and are +2.32% better on S2. They utilize feature gating to encode temporal information forward and backward in time on a 2D network backbone. Applying this to our 3D network would be an interesting direction for future work.

Tab. 6.11.: State-of-the-art comparison on EPIC-Kitchens. A = Actions, V = Verbs, N = Nouns, F = optical flow, AU = audio, O = objects/object features, TO = object features at various temporal locations, ED = Pretraining on very large scale external datasets, VN Mapping = Verb-Noun Mapping; values with '-' not reported by the authors.

			Test S1 (Seen kitchens)						**Test S2 (Unseen kitchens)**					
			Top1 (%)			Top5 (%)			Top1 (%)			Top5 (%)		
Method	Modalities	Params	A	V	N	A	V	N	A	V	N	A	V	N
TSN [24]	RGB+F	20.2M	20.54	48.23	36.71	39.79	84.09	62.32	10.89	39.40	22.70	25.26	74.29	45.72
EF [74]	RGB	11M	19.86	45.68	36.80	41.89	85.56	64.19	10.11	34.89	21.82	25.33	74.56	45.34
R(2+1)D-34 [26]	RGB	64M	26.80	59.10	38.00	46.10	87.40	62.70	16.80	48.40	26.60	31.20	77.20	50.40
LSTA [149]	RGB+F	82M	30.33	59.55	38.35	49.97	85.77	61.49	16.63	47.32	22.16	30.39	77.02	43.15
VN Mapping	RGB	10M	28.10	55.62	38.04	49.38	86.39	62.69	17.86	46.57	25.74	36.26	77.60	51.86
MTL [67]	RGB	10M	29.73	56.00	40.15	50.95	87.06	64.07	17.86	45.99	26.25	35.68	77.98	50.19
VFS [18]	RGB+F+AU	218M	29.13	44.64	30.64	49.71	76.41	59.39	18.40	38.37	15.23	35.64	75.15	39.84
RU [39]	RGB+F+O	52.6M	33.06	56.93	43.05	55.32	85.68	67.12	19.49	43.67	26.77	37.15	73.30	48.28
GSM [148]	RGB	13M	33.45	59.41	41.83	-	-	-	20.18	48.28	26.15	-	-	-
EF [74]	RGB+F+A	32.6M	**36.66**	**66.10**	47.89	**58.62**	**91.28**	**72.80**	20.97	54.46	30.39	39.40	81.23	55.69
LFB [164]	RGB+TO	201.2M	32.70	60.00	45.00	55.30	88.40	71.80	21.20	50.90	31.50	39.40	77.60	57.80
SAP [162]	RGB+O	198.6M	34.80	63.20	**48.30**	55.90	86.10	71.50	23.90	53.20	33.00	40.50	78.20	58.00
AV-SF [167]	RGB+SF+AU	38.5M	35.90	65.70	46.40	57.80	89.50	71.70	24.00	55.80	32.70	**43.20**	**81.70**	**58.90**
R(2+1)D [26]	RGB+ED	118M	34.50	65.20	45.10	53.80	87.40	67.80	**25.60**	**57.30**	**35.70**	42.70	81.10	58.70

EGTEA Gaze+ Only a number of the aforementioned works provide action recognition results on EGTEA Gaze+. We compare against methods that utilize RGB and optical flow in Table 6.12. Despite the absence of optical flow in our method, we are able to outperform all of them with significant margins. The previous state-of-the-art on split 1 of EGTEA Gaze+ [67] achieves 68.99% Top1 and 61.40% mean class accuracy, which we surpass by 1.39% and 1.21% with MD-MTL and by 2.38% and 0.83% with MD-MTL with task mapping, respectively. Furthermore, using MD-MTL the performance on the average of the three splits of EGTEA Gaze+ improves from [94] by 2.47% on Top1 and 2.88% on mean class accuracy.

Tab. 6.12.: Action recognition accuracy on EGTEA Gaze+. Refer to Table 6.11 for the used abbreviations.

		Split 1		**Avg. Splits 1-3**	
Method	Modalities	Top1	Mean cls	Top1	Mean cls
Li *et al.* [85]	RGB+F	-	47.71	-	-
MCN [57]	RGB+F	55.63	-	-	-
RU [39]	RGB+F+O	-	-	60.20	-
EGO-RNN [150]	RGB+F	62.17	-	60.76	-
LSTA [149]	RGB+F	-	-	61.86	-
SAP [162]	RGB+F+O	64.10	-	62.70	-
STAM [94]	RGB+F	68.60	60.54	65.97	57.02
MTL	RGB	68.99	61.40	65.70	57.60
MD-MTL	RGB	70.38	**62.61**	**68.44**	**59.90**
Verb-Noun Mapping	RGB	**71.37**	62.23	-	-

6.4 Supplementary analysis

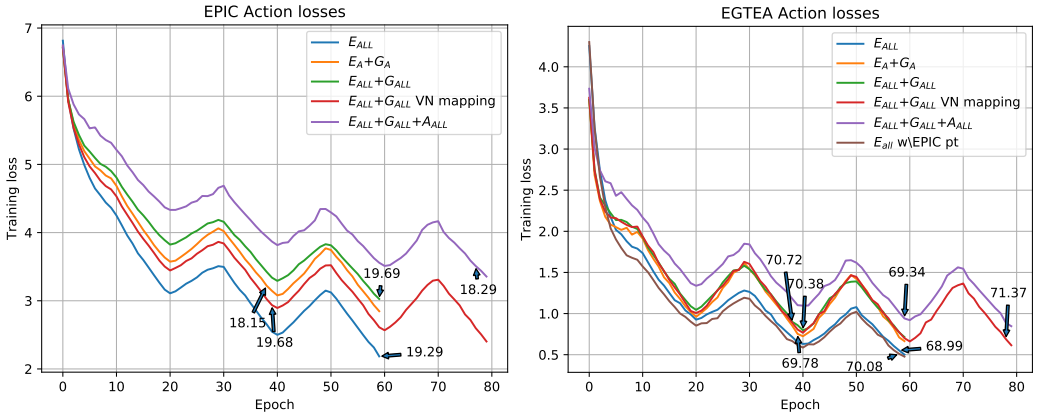
In this section we provide an analysis of the training process and its effects on results (Sec. 6.4.1). Furthermore, we delineate the complete results of the EPIC-Kitchens (Sec. 6.4.2), EGTEA Gaze+ (Sec. 6.4.3), ADL (Sec. 6.4.4), and Charades-EGO (Sec. 6.4.5) models. Finally, in Sec. 6.4.6 we provide a qualitative evaluation of the hand detection tasks for EPIC and EGTEA.

6.4.1 Analysis of Training Losses

In Fig. 6.5 we visualize the training losses for the action tasks of EPIC and EGTEA across a number of SD and MD models. Our aim is to show that the level of overfit on the training set is not necessarily an indicator of the performance on the validation

set. From Fig. 6.5a we observe that the training loss for the SD model E_{ALL} (blue line) is the lowest after the first few epochs and continues to be so after each learning rate cycle (20 epochs) [136]. This is expected, since the model is better able to fit the training set. Evidently, our best performance on EPIC is achieved on epoch 60 for the MD model $E_{ALL}+G_{ALL}$ with training loss almost $+1$ compared to E_{ALL} at the same epoch. This promotes the fact that the losses acquired from the extra tasks do not necessarily affect (negatively) the performance of the action task. On the other hand, when the number of tasks increases significantly and the dataset domains expand in $E_{ALL}+G_{ALL}+A_{ALL}$ (purple line) the model experiences underfitting, as witnessed by the higher training loss and the lower validation performance.

Similar observations can be made from Fig. 6.5b with regard to the training losses on the EGTEA action task. The SD models (blue and brown lines) have lower losses than the MD ones without consistent improvements in validation performance. On the contrary, when having three datasets in the training set, the model underfits for the EGTEA action task. In between the two scenarios, the MD $E_{ALL}+G_{ALL}$ models perform optimally. In fact, because our best performing weights are on epoch 78 (71.37%) we also highlight that the second best performance is found on epoch 38 (70.72%) which is still better than all the other models.



(a) Training losses for the EPIC action task **(b)** Training losses for the EGTEA action task

Fig. 6.5.: Training losses for the action tasks across models. We highlight the best action performance on the validation sets of each dataset. (Best seen in color.)

6.4.2 Extended Results on EPIC-Kitchens

A complete comparison for the action, verb, and noun tasks for EPIC-Kitchens appears in Table 6.13. Mean class precision and recall are provided for the classes

that have more than 100 samples in our training set. Besides the action task, we show that improvements from MD training are also achieved in the additional classification tasks, even without considering their best performing weights (we consider the best action weights in all scenarios). For example, Top1 noun performance increases by 1.05% in the task mapping experiment over the SD baseline and Top5 is improved for verbs and nouns for both two-dataset experiments.

Tab. 6.13.: Extended results on EPIC-Kitchens. For the verb and noun recognition tasks we report Top1/Top5 (%) accuracy, as well as mean class precision and recall for the classes with more than 100 samples in the training set. Model names follow Sec. 6.3.1.

Tasks	Top1 (%)			Top5 (%)			Top1 (%)	
	Actions	Verbs	Nouns	Actions	Verbs	Nouns	S1 Act.	S2 Act.
E_{ALL}	19.29	**48.90**	27.27	35.91	78.18	47.85	**29.73**	**17.86**
E_A+G_A	18.15	-	-	35.93	-	-	24.35	17.04
E_{ALL}+G_{ALL}	**19.69**	45.99	25.65	36.68	78.37	50.67	26.69	17.17
Verb-Noun Mapping	19.68	48.33	**28.32**	**38.68**	**78.99**	**53.44**	28.10	**17.86**
E_{ALL}+G_{ALL}+A_{ALL}	18.29	46.13	25.22	34.15	77.08	48.23	24.17	15.84

	Mean cls Precision / Recall (%)					
	Actions		Verbs		Nouns	
E_{ALL}	3.25	3.04	29.31	24.03	22.68	17.84
E_A+G_A	2.67	2.64	-	-	-	-
E_{ALL}+G_{ALL}	3.32	3.18	24.80	22.18	19.45	17.15
Verb-Noun Mapping	**3.44**	**3.35**	28.13	24.56	**24.50**	**20.05**
E_{ALL}+G_{ALL}+A_{ALL}	2.65	2.87	**32.28**	**24.69**	17.38	15.67

In Tables 6.17a and 6.17b we show per-class f-scores for verbs and nouns where either the SD or MD model is better. We highlight that the MD task mapping model shows improvements on few-shot classes, (*i.e.*, classes with fewer than 100 samples in the training set) without necessarily having additional data from the other dataset. This is an indication that this type of training also reduces overfitting on classes with more training samples. Perhaps, a mapping mechanism that relies on the word vector associations across sample narrations is able to further capitalize on the added data by introducing more robust class associations.

6.4.3 Extended Results on EGTEA Gaze+

Extended results for the verb and noun tasks of EGTEA Gaze+ are provided in Table 6.14. Interestingly, the MD experiments improve over the SD baseline and even the harder case of E_{ALL}+G_{ALL}+A_{ALL} also performs better in Top1 actions and verbs and Top5 actions, verbs, and nouns. In general, the highest Top1 performance is achieved with the task mapping model for all three classification tasks. However, the mean class accuracy for actions and verbs is best for the SD experiment with

EPIC-Kitchens pretraining. A simple explanation could be that due to the smaller size of the EGTEA tasks and the smaller training losses (Fig. 6.5b) the SD model may be able to better fit the classes with fewer training samples.

Tab. 6.14.: Extended results on EGTEA Gaze+. For the verb and noun recognition tasks we report Top1/Top5 (%) and mean class accuracy (%). Model names follow Sec. 6.3.1.

Tasks	Top1 (%) A-V-N			Top5 (%) A-V-N			Mean cls acc. (%) A-V-N		
G_{ALL}	68.99	79.08	79.03	91.74	99.26	96.39	61.40	77.40	72.49
G_{ALL} (pt. on EPIC)	70.08	80.91	79.03	91.99	99.36	96.24	**62.66**	**80.86**	72.97
E_A+G_A	69.78	-	-	**93.37**	-	-	62.31	-	-
E_{ALL}+G_{ALL}	70.38	80.57	79.03	93.08	**99.70**	**97.23**	62.61	80.02	**73.55**
Verb-Noun Mapping	**71.37**	**81.85**	**80.61**	92.78	99.46	97.08	62.23	-	-
E_{ALL}+G_{ALL}+A_{ALL}	69.34	80.42	78.19	92.63	99.41	96.79	60.88	80.19	71.18

6.4.4 Training Settings and Extended Results on ADL

The training and test sets of ADL consist of a relatively low number of action segments. The segments themselves consist of a large number of frames so a sampled short clip is not necessarily representative of the entire segment. A problem that stems from the small training set (111 clips) is that, with a batch size of 32, the network receives very few updates in an epoch ($111/32 = 3.47 \approx 4$), hence making it more difficult to train a network with SGD. Additionally, we found empirically that the cyclical learning rate schedule is more effective with more training steps in an epoch, because more intermediate learning rate values are used, enhancing regularization. To accommodate these issues, we enlarged the training set by copying it over 10 times, essentially prolonging each epoch 10-fold. Furthermore, to train E_{ALL}+G_{ALL}+A_{ALL} we added one more learning cycle to the model and trained for 80 epochs. Since we use random sampling to create a clip from a segment, the network still does not overfit to the training set. In Table 6.15 we present a full comparison of the models with the ADL tasks.

Tab. 6.15.: Extended results on ADL. We report Top1/Top5 (%) and mean class accuracy for the activity and location classification tasks for single- and multi-dataset models. Results in parentheses use the original (small) training set.

Tasks	Top1 (%) A-L		Top5 (%) A-L		Mean cls acc. (%)	
A_{ALL}	**64.65** (62.61)	72.72 **(77.27)**	88.38 **(89.39)**	**96.97** **(96.97)**	**56.10** (48.10)	**43.83** (40.47)
E_{ALL}+G_{ALL}+A_{ALL}	58.08 (54.55)	71.72 (72.73)	86.87 (84.34)	96.47 (96.46)	43.61 (45.90)	39.91 (40.96)

From both experiments we notice an improvement in the activity recognition task when using the enhanced training set, but reduced performance for the location task.

This is possible due to overfitting on the activity task, because of the larger losses incurred during training. As expected, the MD case does not improve performance on the ADL tasks. Notably, the location task is not significantly affected by MD training.

6.4.5 Extended Results on Charades-EGO

In Table 6.16 we present a complete performance comparison on Charades-EGO for the tasks of action, verb, and noun classification. We find that when evaluating on CHAREGO1 the mAP is higher in the MD model for verbs and nouns by 0.50% and 1.30%, respectively from SD $C1_{ALL}$. The $C3_{ALL}$ weights of the MD model outperform the $C3_{ALL}$ SD model, again showing the ability of MD-MTL models to effectively reuse the shared feature space. This indicates that egocentric action recognition can benefit from the use of third-person videos in an MD-MTL setting. Finally, from evaluating on CHAREGO3 we see that mAP is higher in the MD model for the verbs (+0.30%) but drops for actions (-0.40%) and nouns (-0.60%) compared to the SD model.

An interesting insight arises from the results of the verb task. On CHAREGO1, the $C3_{ALL}$ SD model is -1.93% from the best Top1 accuracy, but this decrease (11.14%) is much smaller compared to the decrease in actions (48.79%) and nouns (38.43%). Based on this and from the fact that on CHAREGO3 the $C3_{ALL}$ MD model is consistently improved on the verb task we can conclude that the learned features that define verbs are closely related regardless of the viewing perspective.

Tab. 6.16.: Extended results on Charades-EGO.

Tasks	Top1 (%) A-V-N			Top5 (%) A-V-N			mAP (%) A-V-N		
Validation on charego1									
$C1_{ALL}$ (SD)	**7.05**	16.82	20.07	24.21	61.04	52.53	21.90	36.50	34.80
$C3_{ALL}$ (SD)	3.61	15.70	13.09	15.40	55.67	41.16	14.70	30.60	26.30
$C1_{ALL}$ (MD)	7.01	**17.67**	**21.26**	**24.69**	**61.64**	**53.47**	**22.10**	**37.00**	**36.10**
$C3_{ALL}$ (MD)	5.81	17.30	18.38	21.75	59.85	50.92	20.10	35.50	33.10
Validation on charego3									
$C1_{ALL}$ (SD)	3.55	15.40	11.48	14.70	53.58	38.99	12.30	28.20	23.20
$C3_{ALL}$ (SD)	**8.15**	19.41	**20.15**	**27.02**	63.51	**53.39**	**20.40**	37.70	**35.10**
$C1_{ALL}$ (MD)	6.79	18.11	17.77	22.85	61.67	50.10	18.20	35.00	32.30
$C3_{ALL}$ (MD)	8.12	**20.04**	19.82	26.04	**64.03**	53.11	20.00	**38.00**	34.50

6.4.6 Hand Prediction

In Fig. 6.6 we visualize a series of hand predictions to showcase the effect of multi-dataset training on hand location prediction. We visualize the output heatmaps and the resulting coordinates from the hand coordinate layers. Figs. 6.6a and 6.6b represent the SD EPIC (E_{ALL}) and SD EGTEA (G_{ALL}) model outputs. Figs. 6.6c and 6.6d show the dataset-specific hand task outputs from the E_{ALL}+G_{ALL} model and Fig. 6.6e shows the output from the combined hand prediction task in the task mapping model. The first row of each model contains a segment from test split 1 of EGTEA Gaze+ and the second row a segment from our validation split on EPIC. This diversification is important since the coordinate regression layers experience overfitting on each dataset's visual variations, such as camera position and distance, size and visibility of hands. The ground truth coordinate is given with a green circle and the predicted coordinate with a red circle. We only provide an empirical assessment on hand detection because the original ground truths are synthesized and contain inaccuracies, which would make evaluation inconsistent.

In Figs. 6.6a and 6.6b we see that each model only creates accurate predictions for the sample that originates from its own dataset. For example, in Fig. 6.6a the prediction for both hands is lost in the third column of the EGTEA sample, whereas for the EPIC sample the predictions are consistently in the vicinity of the ground truth. Similarly in Fig. 6.6b, the hands in the EGTEA sample are more accurately localized, but the left hand in the EPIC sample is somewhat missed. The same can be inferred for the MD model with two hand coordinate outputs per hand. We only notice small visual improvements when the hand task of one dataset is applied on the other. Finally, in Fig. 6.6e, mapping all hand outputs together in two coordinate layers (one per hand) has the desired effect of obtaining accurate hand coordinates regardless of the originating dataset, using a single model.

6.5 Discussion

In this chapter, we introduced an effective batch scheme that comprises samples from multiple datasets and associates them with their respective tasks during training. This approach manifests a trade-off between acquiring the optimal estimation of the gradient direction from a batch from a single data distribution and the need to accommodate the presence of samples from multiple datasets in every training iteration. Essentially, we expect the network to find a minimum along a variety of

manifolds which can be costly for optimization, and even not possible if the dataset distributions are incompatible.

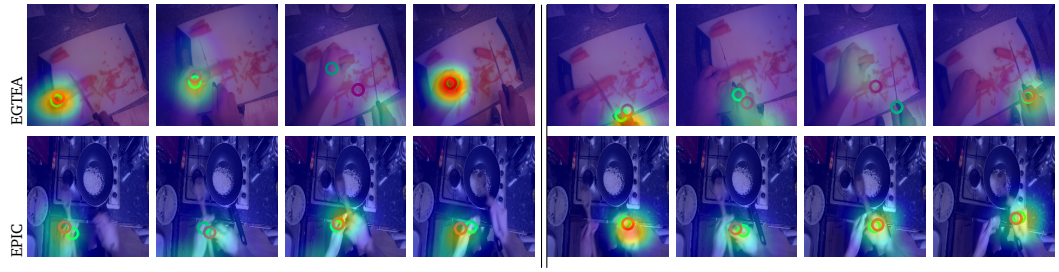
We found that EPIC and EGTEA had improved results in their validation sets which indicates that multi-dataset training is potentially beneficial when semantically related datasets are combined. At the same time, MD-MTL has the practical benefit of producing outputs that reflect tasks from multiple domains within a single model, without necessarily sacrificing accuracy. However, the inclusion of ADL showcases the possible pitfalls of adding a dissimilar dataset. Mainly, the fact that the activities of ADL are annotated with less granularity than the actions of EPIC and EGTEA and the variety of locations, compared to purely kitchens, introduces significant scene variations that complicate the learning task. On the contrary, we observed performance improvements when applying the multi-dataset training scheme on a combination of first- and third-person videos on Charades-EGO. This shows that a difference in video perspectives does not prohibit the network from learning a shared representation when other aspects of the datasets such as the environment and the performed actions are related.

In Sec. 6.3.3 we trained an SD model on EGTEA where we used weights pretrained on EPIC for initialization. Even though EPIC-Kitchens is not as large as the third-person video datasets that are usually employed for pretraining video recognition models, (*e.g.*, [72, 73]) we expected that the similarity between the source and the target domain would prove beneficial, and it did. We also showed that our multi-dataset approach outperforms, in our case, pretraining, while retaining all tasks.

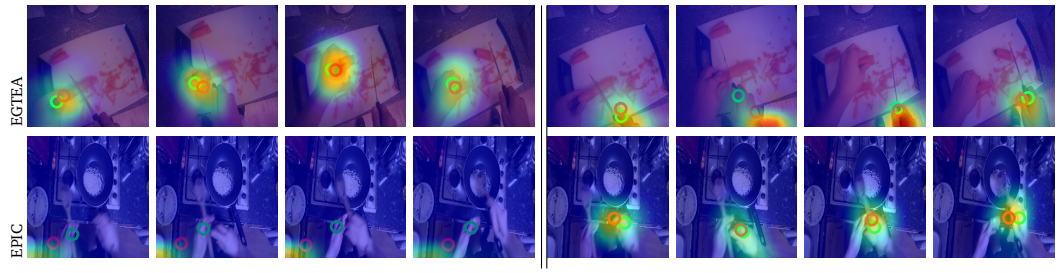
We showed in Sec. 6.3.2 and 6.3.4 that MD training drives classification layers to reuse feature sets for similar classes across tasks. This is an important element of these models, as it occurs without additional supervision, *i.e.*, we do not specify which labels across datasets are related. However, our experiments show that this is a regular phenomenon in MD-MTL. It also reinforces the basic concept of multitask learning that related tasks, even from varied sources, support each other by affecting the shared parameters.

However, class correlations are not so strong to suggest full reuse of features for the same classes. This leads to two distinct observations. First, the capacity of a network when trained for a single dataset is not fully utilized. We showed that the underlying weights can be adapted to accommodate additional information. Hence, whatever minimum is reached with SD training does not necessarily correspond to an optimal exploitation of the millions of parameters of modern neural network architectures. Instead, our experiments show that their capacity is larger than SD

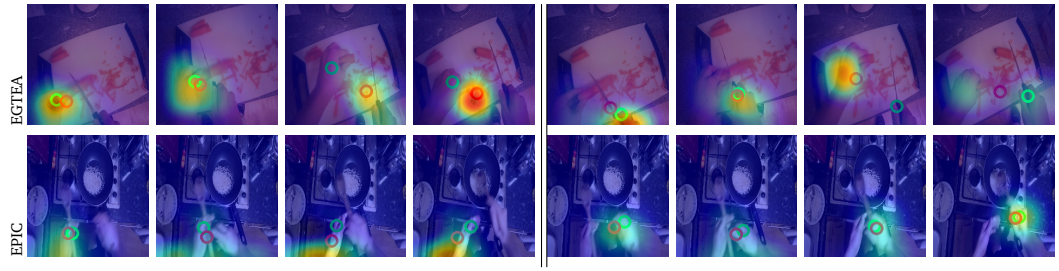
fitting initially suggests. Second, adaptive training mechanisms that substitute hard parameter sharing, such as explicit task-attention mechanisms [91, 97] or implicit weight assignment to tasks [145] are simulating larger network capacity not by inducing better associations among the shared weights, which MD-MTL seems to be achieving, but by establishing mechanisms to mask noisy features that otherwise find their way to the task-specific prediction layers. We believe a soft parameter sharing mechanism is a promising way forward for MD-MTL as the two concepts are complementary.



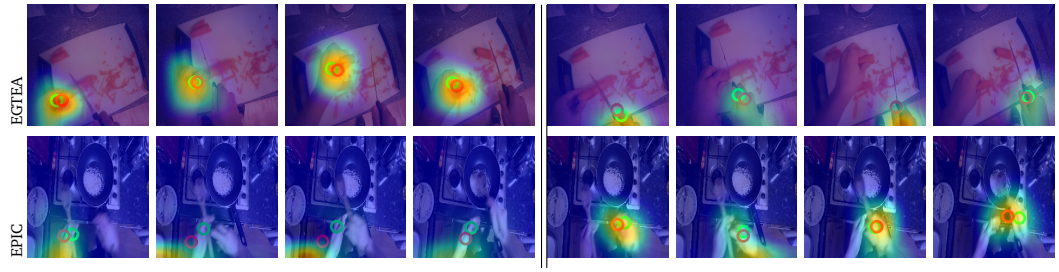
(a) Left - right hand estimation for single dataset EPIC model E_{ALL}.



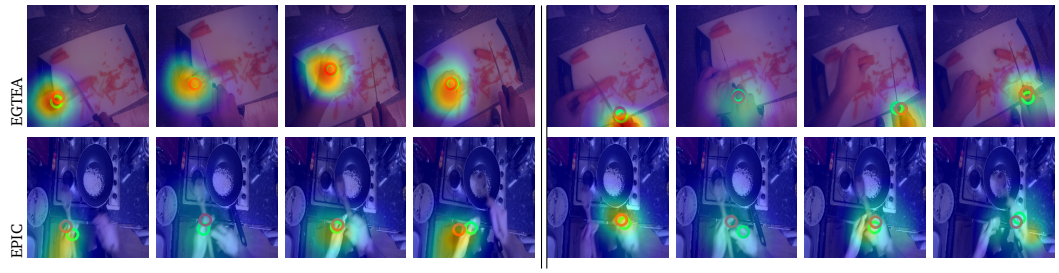
(b) Left - right hand estimation for single dataset EGTEA model G_{ALL}.



(c) Left - right hand estimation for the hand task of EPIC from the multi-dataset model $E_{ALL}+G_{ALL}$.



(d) Left - right hand estimation for the hand task of EGTEA from the multi-dataset model $E_{ALL}+G_{ALL}$.



(e) Left - right hand estimation for the combined hand task of the multi-dataset $E_{ALL}+G_{ALL}$ task mapping model.

Fig. 6.6.: Hand heatmaps and coordinates. We visualize the left and right hand predicitons on their respective sides in the figure.

Tab. 6.17.: Per-class F-scores on EPIC-Kitchens verbs and nouns. Bold for classes that have additional data from mapping. In the top blocks SD is better and in the bottom blocks task mapping is better (fs = few-shot: < 100 samples in training, VN = Verb-Noun mapping).

(a) Selection for EPIC-Kitchens verbs

Verb (cls id)	fs	E_{ALL}	E_{ALL}+G_{ALL}	VN
take (0)	no	24.52	22.30	23.76
put (1)	no	21.89	21.07	21.34
wash (4)	no	36.28	34.51	34.65
mix (6)	no	36.07	33.82	33.86
pour (7)	no	34.25	25.93	28.73
remove (10)	no	8.48	0.00	0.00
sweep (75)	yes	26.31	13.33	0.00
open (2)	no	30.93	30.00	31.66
close (3)	no	25.82	24.45	27.40
cut (5)	no	39.02	36.41	39.08
dry (11)	no	21.21	23.53	27.93
turn-on (12)	no	3.03	0.00	3.13
turn-off (15)	no	3.33	0.00	3.71
adjust (17)	no	17.39	9.52	26.92
fill (24)	yes	0.00	0.00	4.35
apply (36)	yes	0.00	0.00	33.33
set (41)	yes	0.00	0.00	22.22

(b) Selection for EPIC-Kitchens nouns

Noun (cls id)	fs	E_{ALL}	E_{ALL}+G_{ALL}	VN
tap (3)	no	25.00	24.06	22.06
spoon (7)	yes	15.43	7.95	11.98
cupboard (8)	no	30.85	29.35	26.77
drawer (9)	no	29.59	22.52	29.46
fridge (10)	no	29.91	28.68	29.01
water (17)	no	22.86	11.94	10.77
bag (20)	no	5.40	2.18	2.42
oil (24)	no	15.38	13.46	14.28
sink (33)	yes	22.22	10.00	10.00
food (37)	yes	3.49	2.44	2.17
kettle (38)	yes	23.08	11.43	18.92
box (39)	yes	20.19	6.89	11.11
cheese (51)	no	7.69	6.00	3.77
bread (52)	no	3.03	0.00	2.86
top (78)	no	21.80	23.17	20.99
paper (81)	yes	14.29	0.00	0.00
wash. mach. (82)	yes	13.64	0.00	7.69
floor (139)	yes	44.44	0.00	0.00
pan (1)	no	16.48	13.89	16.75
plate (4)	no	27.75	24.00	28.91
knife (5)	yes	11.60	12.42	13.28
bowl (6)	no	5.21	4.76	12.63
lid (11)	yes	9.17	5.68	9.68
hand (12)	no	14.77	19.19	21.87
onion (13)	no	16.67	9.68	19.78
fork (18)	no	9.09	0.00	10.64
chop. board (19)	no	19.30	23.46	21.15
sponge (21)	no	16.30	16.42	17.53
cup (23)	no	22.50	26.70	23.65
bin (25)	yes	8.33	9.75	18.52
bottle (28)	yes	0.00	10.35	3.57
carrot (40)	no	1.33	28.04	32.14
rice (44)	yes	4.45	0.00	4.77
garlic (45)	yes	28.81	20.00	32.65
hob (47)	no	16.67	23.34	23.81
salad (54)	no	7.50	15.38	14.58
coffee (58)	yes	0.00	4.00	4.17
jar (60)	no	0.00	0.00	6.25
skin (68)	yes	4.54	5.26	11.76
lettuce (73)	no	6.25	9.09	6.66
cutlery (75)	no	0.00	0.00	10.00
scissors (76)	yes	0.00	0.00	8.33
cucumber (94)	no	0.00	0.00	33.33
chilli (101)	yes	0.00	9.52	5.26
sugar (103)	yes	0.00	20.00	16.66
heat (106)	yes	31.82	12.50	32.14
rubbish (108)	no	0.00	8.82	3.03
stock (111)	yes	0.00	10.71	5.26

7 Conclusions

With this chapter we conclude the thesis. In Sec. 7.1, we provide a summary of our contributions to egocentric vision and multitask learning. Sec. 7.2 comprises a general discussion of our approach; its advantages, its limitations and our directions for future research in the field.

7.1 Summary

In the following, we delineate our contributions per chapter.

7.1.1 Chapter 3: Object-based Location and Activity Classification

Throughout this chapter, we explored the recognition of indoor locations and human activities in egocentric videos. We utilized a state-of-the-art object detection architecture which was trained on three separate object sets; ADL20 and ADL48 on annotations of the ADL dataset with objects relevant to indoor activities of daily living and one on MS COCO [89] which consists of a more generic object set. We applied object detection on egocentric videos to extract objects at various detection thresholds and classified these detections with Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM) networks to infer locations or activities.

We found that the selection of object set affects the relevance of the detections towards the location classification task and the detection threshold their number and quality. Using the Binary Presence Vector (BPV), we reached 75.50% Top1 accuracy using only the detected objects. Using the object annotations for training and testing increased Top1 location classification accuracy to 80.60%, leading to the conclusion that the lack of object detection noise was preferable. Experimenting with the detection threshold led to the discovery that if detection noise could not be avoided the true-positive/false-negative trade-off favored the abundance of true-positive detections even at the expense of having additional false-positives in the training set. The comparison between ANN and LSTM promoted the incorporation of temporal

structure in the BPVs (Tables 3.9, 3.10) in order to capitalize on the sequential nature of the data and minimize the effects of erroneous detections.

We also found that the more complicated task of activity classification is more difficult to tackle using object-based features alone. Our results showed that certain activities were harder to recognize than others, mostly due to their lower prior in the training set. We also found that activities which belong in semantic 'super' sets tend to be learned as belonging to the one representative activity that has the most instances in the training set.

7.1.2 Chapter 4: Hand- and Object-based Action Classification

In this chapter, we performed a study on the suitability of hand and object sequences for human action recognition from the egocentric perspective. We focused on actions performed in kitchen environments, utilizing the EPIC-Kitchens [24] egocentric video dataset. State-of-the-art methods in activity recognition utilize end-to-end video learning schemes with deep network structures. Alternatively, we explicitly modeled the sequences of hand coordinates detected in the scene. To that end we developed a hand detector better suited to handle egocentric videos. Our method comprises a detection and tracking scheme for the acquisition of hand motions from egocentric videos, which together with the detected objects in the scene are used to recognize egocentric actions as a sequence learning problem.

Our results highlighted the ability to infer a set of hand-based human actions with comparable accuracy to video-based methods, by only using a fraction of the input. In addition, we showed that the inclusion of the presence of relevant detected objects enhanced the feature set and improved performance.

This work is one of the few that comprehend that specialized hand movements can be interpreted as actions without the need to specifically rely on learned visual features for temporal modeling.

7.1.3 Chapter 5: Multitask Learning to Recognize Egocentric Actions

In this chapter, we developed a Multitask Learning (MTL) scheme for egocentric action recognition that supports a variable number of tasks. We trained for actions together with related classification tasks, such as verbs and nouns, and showed

that performance on one or all improves over the single-task baseline. We further combined classification with coordinate regression tasks to learn the egocentric left and right hand and gaze locations as coordinates. We predicted coordinate sequences for video segments by exploiting the temporal dimension of 3D-CNNs.

We highlight that having a network estimate coordinates allows it to focus more on areas with higher correspondence to the hands or other salient objects in the original image.

Our tests on EPIC-Kitchens showed improvements on action recognition performance over single-task learning. On EGTEA Gaze+ we achieved state-of-the-art performance in Top1 action recognition reaching 65.70% surpassing the previous best by more than 3.8%. Lastly, we showed that with our multitask learning setup we can produce accurate hand detectors and gaze estimation models with performance close to state-of-the-art.

7.1.4 Chapter 6: Multi-dataset Multitask Learning to Recognize Egocentric Actions

In this chapter, we introduced a multi-dataset multitask learning (MD-MTL) scheme that allows a single network to assimilate tasks from diverse datasets and tasks simultaneously. By combining samples across datasets within every batch, we effectively approximated having individual batches per dataset on every training iteration.

We applied our scheme in the context of egocentric action and activity classification, on EPIC-Kitchens, EGTEA Gaze+, and ADL datasets and the first- and third-person splits of Charades-EGO. Our results showed that our training scheme offers consistent improvements to classification tasks across datasets when the underlying data distributions are related. Furthermore, we demonstrated that networks acquire similar representations for semantically similar classification tasks without being instructed to do so.

Results on EPIC-Kitchens showed that our method is able to compete with the state-of-the-art. On EGTEA Gaze+ we outperformed more complex networks surpassing our previous best by 2.47%. We highlight that MD-MTL is an efficient technique to combine data from multiple sources without sacrificing the distinctive characteristics of one dataset in order to classify on another.

7.2 General Discussion

We now offer a general discussion about the applicability of our work in real-life scenarios (Sec. 7.2.1). Consequently, we examine the limitations of our work in Sec. 7.2.2. Finally, in Sec. 7.2.3 we present a series of proposals for future research in the domains of egocentric object, location, and action recognition and multitask learning.

7.2.1 Deployment

In this thesis, we embarked on a journey to develop lightweight methods to tackle egocentric vision-related problems. The latest trends in computer vision mostly involve deep network approaches that require large amounts of computational power to train, but also to deploy the resulting models in production environments. Especially in egocentric vision, an application oriented approach cannot depend on the user being equipped with devices that have the ability to carry out intense calculations, fast enough to be of use in realistic daily living scenarios.

Our work in Chapters 3 and 4 comprises uncomplicated feature-based approaches towards recognition that support inference mechanisms with minimal inputs, small amounts of computational requirements, and straightforward outputs. The bottleneck of our approach is the performance of the underlying object detection system, in terms of both speed and accuracy. We argue that most recent object detectors, such as YOLOv4 [12] and MobilenetsV3 [53] offer lightweight versions capable of operating on mobile phone CPUs, even with improved accuracy compared to the detectors used for this work. Our open-ended approach when utilizing an object detector to acquire specific features for hands and objects aimed at exactly this expectation; the development of more efficient devices and algorithms, better suited to provide the features for our location and activity inference mechanisms.

In Chapters 5 and 6 we developed a vision-based approach supported by state-of-the-art 3D Convolutional Neural Networks to ponder on the abilities of an end-to-end network to solve egocentric vision tasks. Our methods are still more efficient than the majority of action recognition models (see Table 6.11 for a comparison on model parameters) and the incorporation of multitask learning allows for complicated outputs to be acquired within a single application context. Arguably, computational speed may still be a limiting factor, but the use of multitask learning positively contributes to the trade-off of accuracy versus computational cost.

7.2.2 Limitations

In this thesis, we addressed the problems of hand, location, action, and activity recognition in egocentric videos. We considered handcrafted features in Chapters 3 and 4 and visual features acquired directly from RGB images using deep neural networks in Chapters 5 and 6.

Binary Presence Vector Our handcrafted feature-based classification approaches rely on the Binary Presence Vector (BPV), introduced in Chapter 3. Its size depends on the number of classes of the object detector and it captures the information regarding the found classes in a given image. Our considerations about BPV were its resilience to noise and its descriptive ability towards locations and activities that are relevant and would, by default, depend on similar inputs. We minimized the effects of noise with temporal models that considered the BPV in sequences, downplaying the significance of shortly occurring accidental detections and enhancing those found regularly. On the contrary, we found that a number of related activities were challenging to classify due to the reliance on similar objects and the lack of information about the detection frequency of objects, user engagement, or information associating the activity with the surrounding environment. Enhancing the BPV with the bounding box sizes and positions in the image slightly improved performance, but did not alleviate this problem because the training set was too small to provide enough samples to allow relying on such detailed features. A related feature we did not consider was the view of the detected objects. An object seen from its front might indicate that its use is imminent, but a side or rear view would point to it being ignored.

Hand-based action recognition In Chapter 4, we focused on the user's hands to capture the motion sequences that occurred during actions and the detected objects for contextual scene information. We followed our approach from Chapter 3 by modeling the objects as BPVs and the hands as coordinates, instead of using visual features directly. By reducing the visual information into this minimalistic representation, we created a lightweight system that classified egocentric actions with comparable performance to image-based deep network approaches. The lack of a more elaborate description of the scene that would be captured by a deep network is the most straightforward limitation of this approach. This means that our classifier does not have access to fine-grained motion descriptors. In this context, another limitation is not considering the ego-motions during acquisition of the hand motion sequences. Knowledge of ego-motion could reduce the negative effects from rapid

camera movements that are unrelated to hand activities. Furthermore, the lack of a direct connection between objects and hands would result in missing the interactions between them, which are detrimental for distinguishing actions. A limitation of our hand identification approach is the assumption that there at most two hands in view, thus restricting this step to single-person egocentric videos.

Optical flow In Chapters 5 and 6, we utilized deep networks for action recognition with RGB images directly as input. A relevant source of information that we did not consider was optical flow. Our approach with 3D-CNNs was able to capture features about the temporal changes of images throughout a video segment, but the use of flow has shown to be a more straightforward, albeit computationally expensive way, to reliably incorporate knowledge of motions in spatio-temporal networks.

Multitask Learning Regarding multitask learning, the lack of a direct way to measure the similarity of the output task distributions required extensive experimentation to find appropriate task combinations. Additionally, our MTL network design used a naive approach for the task combinations with the sole intention of implicitly affecting tasks through joint training of the shared feature extractor. More direct dependencies between tasks, such as using the gaze or hand posterior as input for the action recognition task could improve performance. The MTL experiments did not consider an object recognition task that would explicitly point the network to the area of interest. Instead, the noun component of the action classes was used as the indication for the object utilized for the action. Consequently, this led to the lack of a direct way of focusing on hand-object interactions.

Class imbalance A general limitation of this work is the difficulty in handling class imbalance for the tasks and datasets we considered. Especially in the earlier chapters which consisted of simpler features, this issue negatively affected the recognition of poorly represented classes in the datasets. Admittedly, class imbalance was a limiting factor to our deep network approaches as well. Despite the size of the datasets we experimented with, we found that only a small percentage of classes would have enough samples to be learned robustly. Instead, intra-class variance for classes with very few training instances was a debilitating factor which showed the ineffectiveness of our models in tackling rare events.

7.2.3 Future Research

We have developed a number of approaches to combine handcrafted and visual features for egocentric video understanding. In this section, we will discuss possible ways to further advance the field.

Handcrafted feature-based approaches have the advantage of low computational requirements, but they lack in descriptive quality. Enhancing the BPV with location priors would provide a clear cue for the distinction of similar activities that are characterized by the same objects. Another approach for both locations and activities is providing classification results from past segments into the current instance. This requires treating the input as a stream instead of fixed-sized segments, unconnected with each other. Handling the temporal classification problem in this fashion also makes sense from a practical perspective, as it is expected that an application is running without interruption and past information would be constantly available. In this direction we highlight the approach of Possas *et al.* [114] who utilized motion and visual inputs interchangeably for continuous activity recognition.

The output of our hand tracking method can be augmented further if we consider a number of features such as the distances between hands in a frame and the distance that has been traversed from hands in series of frames. Implicitly, this is what we expect the network to learn. However, a more direct way of introducing this information would be beneficial for the disambiguation of motions and subsequently the recognition of actions. In this spirit, we refer to the work of Cai *et al.* [15] who explicitly modeled the interactions between hand grasps and objects and associated them with the performed actions. Smoothing the hand tracks, as in [42], would further reduce the spatial noise in the hand input signal. Introducing hand poses using a hand pose detector, (*e.g.*, [75]), on top of our hand detector could add significant detail to the hand motions and enhance action recognition. An example of the benefit of hand poses of egocentric actions is in the work of Tekin *et al.* [153]. In recent years, interesting applications that depend on hand detection and tracking have emerged in the domain of Virtual Reality [32, 47].

Our approach in multitask learning has been a naive one and mostly focused on improving classification performance by choosing sensible additional tasks to expand the output space. Recent approaches based on visual inputs consider attention models with task-specific layers that adapt specifically to individual tasks. For example, in [97] a generic feature-extracting backbone is augmented by residual adapter blocks [117] that capture task-specific features and are utilized to disentangle task-specific information at test time, based on the single task output that is expected.

Further improvements can be made with advanced training mechanisms for loss and gradient balancing across tasks during training. We have identified this issue when discussing the size of the training loss per task in our models in Chapters 5 and 6. In practice, more challenging tasks produce higher losses that influence the network in their favor at the expense of the remaining tasks. Ways to mediate this have been proposed in [97, 135] to reduce the effect of task-specific gradients in training the shared parts of the network. They use an additional adversarial task [90] as part of the multitask network which purpose is to disambiguate the tasks by making their gradients statistically indistinguishable to the shared block. As a result, this minimizes the effects of unbalanced tasks.

In terms of augmenting the space of learned visual features for recognition, we address the works of [4] and [164]. Baradel *et al.* [4] use a distinct network branch for the acquisition of object masks and additional branches for inter-object associations and global visual features from the scene, which they model temporally with Recurrent Networks. This provides detailed information about the object relationships in the scene to helps distinguish them across actions. Another example comes from [164] who again use object features from the scene to enhance the action model, with the addition that these features stem not only from the current segment, but from all the frames leading to it. Their approach requires a full pass on the video to acquire the relevant features. However, we argue that incorporating at least the past information in a streaming manner is a feasible approach that can consistently improve egocentric action recognition.

Lastly, we can refer to some recent attempts to tackle few-shot and zero-shot learning in videos. We believe that this domain offers potential solutions to address the intra-class variance in classification tasks, especially when insufficient samples are available in the training set. Shen *et al.* [128], devised a multitask network that detects object and action regions with longer task-specific branches compared to our work. This allows each task to make independent inferences and produce accurate object-action pairs at test time that have not been part of the training set. In [102], a Generative Adversarial Network (GAN) is used to produce video clips of unseen actions. Even though the visual output is impressive we argue that the value from this work can expand by incorporating the synthetic clips into the training set. Generating clips for classes with few or no samples can potentially be a solution for their recognition in unbalanced datasets. A similar approach was introduced in [115] to learn zero-shot classes. The main difference was the use of augmentations at the annotation level of existing video clips instead of GANs to produce new ones. These label augmentations involved manipulation of the temporal structure of the video clip by reversing the order of frames. This led to the introduction of unseen action

instances in the training set which could effectively be used for data augmentation and zero-shot learning.

We have advanced the field of egocentric video recognition by highlighting the importance of egocentric objects and hands and by incorporating multitask learning in an analytical way that clearly demonstrates its advantages, its disadvantages, and its rooms for improvement. The modularity of our approaches allows for relatively straightforward incorporation of elements from the aforementioned related works. By building on top of our work, further great strides can be made to promote egocentric vision into the foreground of daily-living assistive technologies.

Bibliography

[1] G. Abebe and A. Cavallaro. “A Long Short-Term Memory Convolutional Neural Network for First-Person Vision Activity Recognition”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. Venice, Italy: IEEE, 2017, pp. 1339–1346 (cit. on p. 16).

[2] H. Altwaijry, M. Moghimi, and S. Belongie. “Recognizing Locations with Google Glass: A Case Study”. In: *IEEE Winter Conference on Applications of Computer Vision*. 2014, pp. 167–174 (cit. on p. 11).

[3] S. Bambach, S. Lee, D. J. Crandall, and C. Yu. “Lending A Hand: Detecting Hands and Recognizing Activities in Complex Egocentric Interactions”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1949–1957 (cit. on pp. 15, 18, 53–55, 79).

[4] F. Baradel, N. Neverova, C. Wolf, J. Mille, and G. Mori. “Object Level Visual Reasoning in Videos”. In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 106–122 (cit. on pp. 13, 15, 16, 19–21, 65, 116).

[5] F. Baradel, C. Wolf, and J. Mille. “Human Action Recognition: Pose-Based Attention Draws Focus to Hands”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. Venice: IEEE, 2017, pp. 604–613 (cit. on p. 50).

[6] G. Bertasius, A. Chan, and J. Shi. “Egocentric Basketball Motion Planning from a Single First-Person Image”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT, USA: IEEE, 2018, pp. 5889–5898 (cit. on p. 79).

[7] G. Bertasius, H. S. Park, S. X. Yu, and J. Shi. “First Person Action-Object Detection with EgoNet”. In: *Proceedings of Robotics: Science and Systems*. 2017 (cit. on pp. 15, 16, 19, 38, 49).

[8] A. Betancourt. “A Sequential Classifier for Hand Detection in the Framework of Egocentric Vision”. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2014, pp. 600–605 (cit. on p. 18).

[9] A. Betancourt, N. Díaz-Rodríguez, E. Barakova, et al. “Unsupervised Understanding of Location and Illumination Changes in Egocentric Videos”. In: *Pervasive and Mobile Computing* 40 (2017), pp. 414–429 (cit. on p. 11).

[10] A. Betancourt, P. Morerio, E. Barakova, et al. “Left/Right Hand Segmentation in Egocentric Videos”. In: *Computer Vision and Image Understanding* 154 (2016), pp. 73–81 (cit. on p. 18).

[11]A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. “Simple Online and Realtime Tracking”. In: *IEEE International Conference on Image Processing (ICIP)*. 2016, pp. 3464–3468 (cit. on pp. 52, 55, 70).

[12]A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao. “YOLOv4: Optimal Speed and Accuracy of Object Detection”. In: *arXiv:2004.10934 [cs, eess]* (2020). arXiv: 2004.10934 [cs, eess] (cit. on p. 112).

[13]M. Bolaños, Á. Peris, F. Casacuberta, S. Soler, and P. Radeva. “Egocentric Video Description Based on Temporally-Linked Sequences”. In: *Journal of Visual Communication and Image Representation, Special Issue on Egocentric Vision and Lifelogging Tools* 50 (2018), pp. 205–216 (cit. on p. 17).

[14]F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. “Activitynet: A Large-Scale Video Benchmark for Human Activity Understanding”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 961–970 (cit. on p. 79).

[15]M. Cai, K. Kitani, and Y. Sato. “Understanding Hand-Object Manipulation by Modeling the Contextual Relationship between Actions, Grasp Types and Object Attributes”. In: *arXiv:1807.08254 [cs]* (2018). arXiv: 1807.08254 [cs] (cit. on p. 115).

[16]M. Cai, K. Kitani, and Y. Sato. “Understanding Hand-Object Manipulation with Grasp Types and Object Attributes”. In: *Robotics: Science and Systems XII*. Robotics: Science and Systems Foundation, 2016 (cit. on p. 19).

[17]J. Carreira and A. Zisserman. “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4724–4733 (cit. on pp. 15, 50, 70).

[18]A. Cartas, J. Luque, P. Radeva, C. Segura, and M. Dimiccoli. “Seeing and Hearing Egocentric Actions: How Much Can We Learn?” In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2019, pp. 4470–4480 (cit. on pp. 16, 97).

[19]R. Caruana. “Multitask Learning”. In: *Machine Learning* 28 (1997), pp. 41–75 (cit. on pp. 19, 65, 66, 76, 80).

[20]Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng. “Multi-Fiber Networks for Video Recognition”. In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 364–380 (cit. on pp. 15, 60, 62, 67, 70, 80, 81, 87).

[21]E. Chong, N. Ruiz, Y. Wang, et al. “Connecting Gaze, Scene, and Attention: Generalized Attention Estimation via Joint Modeling of Gaze and Scene Saliency”. In: *European Conference on Computer Vision (ECCV)*. Vol. 11209. Springer International Publishing, 2018, pp. 397–412 (cit. on p. 22).

[22]H. Coskun, Z. Zia, B. Tekin, et al. “Domain-Specific Priors and Meta Learning for Low-Shot First-Person Action Recognition”. In: *arXiv:1907.09382 [cs]* (2019). arXiv: 1907.09382 [cs] (cit. on p. 22).

[23]N. Dalal and B. Triggs. “Histograms of Oriented Gradients for Human Detection”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. 2005, 886–893 vol. 1 (cit. on p. 29).

[24]D. Damen, H. Doughty, G. Maria Farinella, et al. “Scaling Egocentric Vision: The EPIC-KITCHENS Dataset”. In: *European Conference on Computer Vision (ECCV)*. 2018 (cit. on pp. 23, 50–52, 62, 67, 70–73, 79–81, 85, 97, 110).

[25]D. Damen, T. Leelasawassuk, and W. Mayol-Cuevas. “You-Do, I-Learn: Egocentric Unsupervised Discovery of Objects and Their Modes of Interaction towards Video-Based Guidance”. In: *Computer Vision and Image Understanding* 149 (2016), pp. 98–112 (cit. on pp. 11, 13).

[26]G. Deepti, D. Tran, and D. Mahajan. “Large-Scale Weakly-Supervised Pre-Training for Video Action Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, 2019, pp. 12038–12047 (cit. on pp. 21, 97).

[27]A. G. del Molino, C. Tan, J. Lim, and A. Tan. “Summarization of Egocentric Videos: A Comprehensive Survey”. In: *IEEE Transactions on Human-Machine Systems* 47.1 (2017), pp. 65–76 (cit. on p. 11).

[28]J. Deng, W. Dong, R. Socher, et al. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 248–255 (cit. on pp. 15, 34, 54).

[29]J. Donahue, L. A. Hendricks, M. Rohrbach, et al. “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.4 (2017), pp. 677–691 (cit. on p. 15).

[30]V. Dovgalecs, R. Mégret, and Y. Berthoumieu. “Multiple Feature Fusion Based on Co-Training Approach and Time Regularization for Place Classification in Wearable Video”. In: *Advances in Multimedia* 2013 (2013) (cit. on pp. 11, 12).

[31]L. Fa, Y. Song, and X. Shu. “Global and Local C3D Ensemble System for First Person Interactive Action Recognition”. In: *MultiMedia Modeling*. Lecture Notes in Computer Science. Springer International Publishing, 2018, pp. 153–164 (cit. on p. 16).

[32]Z. Fan, V. Bazarevsky, A. Vakunov, et al. “MediaPipe Hands: On-Device Real-Time Hand Tracking”. en. In: *arXiv:2006.10214 [cs]* (2020). arXiv: 2006.10214 [cs] (cit. on p. 115).

[33]A. Fathi, A. Farhadi, and J. M. Rehg. “Understanding Egocentric Activities”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2011, pp. 407–414 (cit. on pp. 13, 14, 18, 65, 67).

[34]A. Fathi, Y. Li, and J. M. Rehg. “Learning to Recognize Daily Actions Using Gaze”. In: *European Conference on Computer Vision (ECCV)*. Berlin, Heidelberg, 2012, pp. 314–327 (cit. on pp. 14, 22, 23, 80).

[35]A. Fathi, X. Ren, and J. M. Rehg. “Learning to Recognize Objects in Egocentric Activities”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011, pp. 3281–3288 (cit. on pp. 22, 23).

[36]C. Feichtenhofer, H. Fan, J. Malik, and K. He. “SlowFast Networks for Video Recognition”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 6201–6210 (cit. on pp. 15, 81).

[37]A. Furnari, S. Battiato, and G. M. Farinella. “Leveraging Uncertainty to Rethink Loss Functions and Evaluation Measures for Egocentric Action Anticipation”. In: *European Conference on Computer Vision Workshops (ECCVW)*. Vol. 11133. Springer International Publishing, 2019, pp. 389–405 (cit. on p. 16).

[38]A. Furnari, S. Battiato, K. Grauman, and G. M. Farinella. “Next-Active-Object Prediction from Egocentric Videos”. In: *Journal of Visual Communication and Image Representation* 49 (2017), pp. 401–411 (cit. on pp. 13, 18, 50).

[39]A. Furnari and G. M. Farinella. “What Would You Expect? Anticipating Egocentric Actions with Rolling-Unrolling LSTMs and Modality Attention”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2019 (cit. on pp. 17, 18, 73, 74, 97, 98).

[40]A. Furnari, G. M. Farinella, and S. Battiato. “Recognizing Personal Locations From Egocentric Videos”. In: *IEEE Transactions on Human-Machine Systems* 47.1 (2017), pp. 6–18 (cit. on pp. 12, 22, 23).

[41]A. Furnari, G. M. Farinella, and S. Battiato. “Temporal Segmentation of Egocentric Videos to Highlight Personal Locations of Interest”. In: *European Conference on Computer Vision Workshops (ECCVW)*. Springer International Publishing, 2016, pp. 474–489 (cit. on pp. 12, 22, 23).

[42]G. Garg, S. Hegde, R. Perla, et al. “DrawInAir: A Lightweight Gestural Interface Based on Fingertip Regression”. In: *European Conference on Computer Vision Workshops (ECCVW)*. Vol. 11134. Springer International Publishing, 2019, pp. 229–240 (cit. on pp. 18, 59, 61, 64, 115).

[43]R. Girdhar and D. Ramanan. “Attentional Pooling for Action Recognition”. In: *31st International Conference on Neural Information Processing Systems*. 2017, pp. 33–44 (cit. on p. 15).

[44]R. Girshick. “Fast R-CNN”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1440–1448 (cit. on p. 35).

[45]G. Gkioxari and J. Malik. “Finding Action Tubes”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 759–768 (cit. on p. 15).

[46]H. Guo, T. Tang, G. Luo, et al. “Multi-Domain Pose Network for Multi-Person Pose Estimation and Tracking”. In: *European Conference on Computer Vision Workshops (ECCVW)*. Vol. 11130. Springer International Publishing, 2019, pp. 209–216 (cit. on p. 22).

[47] S. Han, B. Liu, R. Cabezas, et al. “MEgATrack: Monochrome Egocentric Articulated Hand-Tracking for Virtual Reality”. en. In: *ACM Transactions on Graphics* 39.4 (2020) (cit. on p. 115).

[48] K. He, G. Gkioxari, P. Dollár, and R. Girshick. “Mask R-CNN”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017 (cit. on pp. 20, 45).

[49] K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778 (cit. on p. 15).

[50] K. He, X. Zhang, S. Ren, and J. Sun. “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), pp. 1904–1916 (cit. on p. 57).

[51] S. Hochreiter and J. Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (1997), pp. 1735–1780 (cit. on pp. 31, 57).

[52] S. Hosseini, M. A. Shabani, and Nam Ik Cho. “Distill-2MD-MTL: Data Distillation Based on Multi-Dataset Multi-Domain Multi-Task Frame Work to Solve Face Related Tasks, Multi Task Learning, Semi-Supervised Learning”. In: *arXiv:1907.03402 [cs]* (2019). arXiv: 1907.03402 [cs] (cit. on p. 22).

[53] A. Howard, M. Sandler, B. Chen, et al. “Searching for MobileNetV3”. In: *IEEE International Conference on Computer Vision (ICCV)*. Seoul, Korea: IEEE, 2019, pp. 1314–1324 (cit. on p. 112).

[54] S. Huang, W. Wang, S. He, and R. W. H. Lau. “Egocentric Temporal Action Proposals”. In: *IEEE Transactions on Image Processing* 27.2 (2018), pp. 764–777 (cit. on p. 17).

[55] X. Huang, C. Shen, X. Boix, and Q. Zhao. “SALICON: Reducing the Semantic Gap in Saliency Prediction by Adapting Deep Neural Networks”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 262–270 (cit. on p. 74).

[56] Y. Huang, M. Cai, Z. Li, and Y. Sato. “Predicting Gaze in Egocentric Video by Learning Task-Dependent Attention Transition”. In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 789–804 (cit. on p. 74).

[57] Y. Huang, Z. Li, M. Cai, and Y. Sato. “Mutual Context Network for Jointly Estimating Egocentric Gaze and Actions”. In: (2019). arXiv: 1901.01874 (cit. on pp. 17, 20, 21, 73, 74, 76, 98).

[58] S. Ioffe and C. Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *32nd International Conference on Machine Learning (ICML)*. ICML’15. 2015, pp. 448–456 (cit. on p. 21).

[59] A. Iwashita, Y. Takamine, R. Kurazume, and M. S. Ryoo. “First-Person Animal Activity Recognition from Egocentric Videos”. In: *22nd International Conference on Pattern Recognition (ICPR)*. 2014 (cit. on p. 79).

[60] M. Jain, J. C. van Gemert, and C. G. M. Snoek. “What Do 15,000 Object Categories Tell Us about Classifying and Localizing Actions?” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, 2015, pp. 46–55 (cit. on p. 30).

[61]H. Jiang, Y. Song, J. He, and X. Shu. “Cross Fusion for Egocentric Interactive Action Recognition”. In: *MultiMedia Modeling*. Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 714–726 (cit. on p. 16).

[62]L. Kaiser, A. N. Gomez, N. Shazeer, et al. “One Model To Learn Them All”. In: *arXiv:1706.05137 [cs, stat]* (2017). arXiv: 1706.05137 [cs, stat] (cit. on p. 21).

[63]Rudolph Emil Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: *Transactions of the ASME–Journal of Basic Engineering* 82.Series D (1960), pp. 35–45 (cit. on p. 56).

[64]V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid. “Joint Learning of Object and Action Detectors”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2001–2010 (cit. on p. 20).

[65]T. Kanade and M. Hebert. “First-Person Vision”. In: *Proceedings of the IEEE* 100.8 (2012), pp. 2442–2453 (cit. on pp. 13, 49, 65).

[66]G. Kapidis, R. Poppe, E. Van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. “Egocentric Hand Track and Object-Based Human Action Recognition”. In: *19th IEEE Conference on Ubiquitous Intelligence and Computing (UIC)*. Leicester, UK, 2019, pp. 922–929 (cit. on pp. 8, 16, 67).

[67]G. Kapidis, R. Poppe, E. van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. “Multitask Learning to Improve Egocentric Action Recognition”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2019, pp. 4396–4405 (cit. on pp. 8, 16, 80, 97, 98).

[68]G. Kapidis, R. Poppe, E. van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. “Object Detection-Based Location and Activity Classification from Egocentric Videos: A Systematic Analysis”. In: *Smart Assisted Living*. Springer International Publishing, 2020, pp. 119–145 (cit. on pp. 8, 16, 49, 59).

[69]G. Kapidis, R. Poppe, and R. C. Veltkamp. “Multi-Dataset, Multitask Learning of Egocentric Vision Tasks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence, Special Issue on Egocentric Perception* (2021) (cit. on p. 9).

[70]G. Kapidis, R. W. Poppe, E. A. van Dam, R. C. Veltkamp, and L. P. J. J. Noldus. “Where Am I? Comparing CNN and LSTM for Location Classification in Egocentric Videos”. In: *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 2018, pp. 878–883 (cit. on pp. 8, 29, 49, 59).

[71]S. Karaman, J. Benois-Pineau, R. Megret, et al. “Human Daily Activities Indexing in Videos from Wearable Cameras for Monitoring of Patients with Dementia Diseases”. In: *20th International Conference on Pattern Recognition*. 2010, pp. 4113–4116 (cit. on p. 29).

[72]A. Karpathy, G. Toderici, S. Shetty, et al. “Large-Scale Video Classification with Convolutional Neural Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 1725–1732 (cit. on pp. 15, 79, 104).

[73]W. Kay, J. Carreira, K. Simonyan, et al. “The Kinetics Human Action Video Dataset”. In: *arXiv:1705.06950 [cs]* (2017). arXiv: 1705.06950 [cs] (cit. on pp. 15, 79, 87, 104).

[74]E. Kazakos, A. Nagrani, A. Zisserman, and D. Damen. “EPIC-Fusion: Audio-Visual Temporal Binding for Egocentric Action Recognition”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 5491–5500 (cit. on pp. 15, 16, 97).

[75]A. Kendall, M. Grimes, and R. Cipolla. “PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization”. In: *IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE, 2015, pp. 2938–2946 (cit. on p. 115).

[76]I. Kokkinos. “UberNet: Training a ‘Universal’ Convolutional Neural Network for Low-, Mid-, and High-Level Vision Using Diverse Datasets and Limited Memory”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5454–5463 (cit. on pp. 20–22).

[77]K. S. Kretch, J. M. Franchak, and K. E. Adolph. “Crawling and Walking Infants See the World Differently”. In: *Child Development* 85.4 (2014), pp. 1503–1518 (cit. on p. 11).

[78]H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. “HMDB: A Large Video Database for Human Motion Recognition”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2011, pp. 2556–2563 (cit. on p. 79).

[79]H. W. Kuhn. “The Hungarian Method for the Assignment Problem”. In: *Naval Research Logistics Quarterly* 2.1-2 (1955), pp. 83–97 (cit. on p. 56).

[80]H. Kwon, Y. Kim, J. S. Lee, and M. Cho. “First Person Action Recognition via Two-Stream ConvNet with Long-Term Fusion Pooling”. In: *Pattern Recognition Letters* 112 (2018), pp. 161–167 (cit. on p. 17).

[81]N. Lee, C. Kim, W. Choi, M. Pyeon, and Y. Kim. “Development of Indoor Localization System Using a Mobile Data Acquisition Platform and BoW Image Matching”. In: *KSCE Journal of Civil Engineering* 21.1 (2017), pp. 418–430 (cit. on p. 11).

[82]Y. J. Lee, J. Ghosh, and K. Grauman. “Discovering Important People and Objects for Egocentric Video Summarization”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012, pp. 1346–1353 (cit. on pp. 22, 23, 79).

[83]Y. J. Lee and K. Grauman. “Predicting Important Objects for Egocentric Video Summarization”. In: *International Journal of Computer Vision* 114.1 (2015), pp. 38–55 (cit. on p. 79).

[84]C. Li and K. M. Kitani. “Pixel-Level Hand Detection in Ego-Centric Videos”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013, pp. 3570–3577 (cit. on pp. 18, 53–55, 79).

[85]Y. Li, M. Liu, and J. M. Rehg. “In the Eye of Beholder: Joint Learning of Gaze and Actions in First Person Video”. In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 639–655 (cit. on pp. 13, 17, 20–23, 25, 53–55, 67, 70, 73, 74, 76, 79–81, 85, 98).

[86]Y. Li, Y. Zhefan, and J. M. Rehg. “Delving into Egocentric Actions”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 287–295 (cit. on pp. 13–15, 53, 65, 67).

[87]Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. G. M. Snoek. “VideoLSTM Convolves, Attends and Flows for Action Recognition”. In: *Computer Vision and Image Understanding (CVIU)* 166 (2018), pp. 41–50 (cit. on p. 15).

[88]Z. Li, Y. Huang, M. Cai, and Y. Sato. “Manipulation-Skill Assessment from Videos with Spatial Attention Network”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2019, pp. 4385–4395 (cit. on p. 17).

[89]T.-Y. Lin, M. Maire, S. Belongie, et al. “Microsoft COCO: Common Objects in Context”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2014, pp. 740–755 (cit. on pp. 35, 41, 54, 109).

[90]P. Liu, X. Qiu, and X. Huang. “Adversarial Multi-Task Learning for Text Classification”. In: *ACL 2017*. Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 1–10 (cit. on p. 116).

[91]S. Liu, E. Johns, and A. J. Davison. “End-to-End Multi-Task Learning with Attention”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 1871–1880 (cit. on pp. 20, 105).

[92]D. G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110 (cit. on p. 29).

[93]G. Lu, Y. Yan, N. Sebe, and C. Kambhamettu. “Indoor Localization via Multi-View Images and Videos”. In: *Computer Vision and Image Understanding* 161 (2017), pp. 145–160 (cit. on p. 11).

[94]M. Lu, D. Liao, and Z.-N. Li. “Learning Spatiotemporal Attention for Egocentric Action Recognition”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2019, pp. 4425–4434 (cit. on pp. 17, 98).

[95]D. C. Luvizon, D. Picard, and H. Tabia. “2D/3D Pose Estimation and Action Recognition Using Multitask Deep Learning”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT, USA: IEEE, 2018, pp. 5137–5146 (cit. on p. 20).

[96]M. Ma, H. Fan, and K. M. Kitani. “Going Deeper into First-Person Activity Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 1894–1903 (cit. on pp. 11, 13, 15, 16, 19–21).

[97]K.-K. Maninis, I. Radosavovic, and I. Kokkinos. “Attentive Single-Tasking of Multiple Tasks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 1851–1860 (cit. on pp. 20, 105, 115, 116).

[98]K. Matsuo, K. Yamada, S. Ueno, and S. Naito. “An Attention-Based Activity Recognition for Egocentric Video”. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Columbus, OH, USA: IEEE, 2014, pp. 565–570 (cit. on pp. 13, 17).

[99]I. Misra, A. Shrivastava, A. Gupta, and M. Hebert. “Cross-Stitch Networks for Multi-Task Learning”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 3994–4003 (cit. on p. 20).

[100]D. J. Moore, I. A. Essa, and M. H. Hayes. “Exploiting Human Actions and Object Context for Recognition Tasks”. In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 1. 1999, 80–86 vol.1 (cit. on p. 13).

[101]K. Nakamura, S. Yeung, A. Alahi, and L. Fei-Fei. “Jointly Learning Energy Expenditures and Activities Using Egocentric Multimodal Signals”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6817–6826 (cit. on pp. 16, 20–23, 79).

[102]M. Nawhal, M. Zhai, A. Lehrmann, and L. Sigal. “Zero-Shot Generation of Human-Object Interaction Videos”. In: *arXiv:1912.02401 [cs, eess]* (2019). arXiv: 1912.02401 [cs, eess] (cit. on p. 116).

[103]T.-H.-C. Nguyen, J.-C. Nebel, and F. Florez-Revuelta. “Recognition of Activities of Daily Living from Egocentric Videos Using Hands Detected by a Deep Convolutional Network”. In: *Image Analysis and Recognition*. Springer International Publishing, 2018, pp. 390–398 (cit. on pp. 18, 48–50, 59).

[104]T.-H.-C. Nguyen, J.-C. Nebel, and F. Florez-Revuelta. “Recognition of Activities of Daily Living with Egocentric Vision: A Review”. In: *Sensors* 16.1 (2016), p. 72 (cit. on pp. 14, 29).

[105]A. Nibali, Z. He, S. Morgan, and L. Prendergast. “Numerical Coordinate Regression with Convolutional Neural Networks”. In: *CoRR abs/1801.07372* (2018). arXiv: 1801.07372 (cit. on pp. 68, 69, 82).

[106]K. Ogaki, K. M. Kitani, Y. Sugano, and Y. Sato. “Coupling Eye-Motion and Ego-Motion Features for First-Person Activity Recognition”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Providence, RI, USA: IEEE, 2012, pp. 1–7 (cit. on p. 13).

[107]P. Parmar and B. T. Morris. “What and How Well You Performed? A Multitask Learning Approach to Action Quality Assessment”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, 2019, pp. 304–313 (cit. on p. 20).

[108]T. Perrett and D. Damen. “Recurrent Assistance: Cross-Dataset Training of LSTMs on Kitchen Tasks”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. Venice, Italy: IEEE, 2017, pp. 1354–1362 (cit. on pp. 22, 92).

[109]K. Pfeiffer, A. Hermans, I. Sárándi, M. Weber, and B. Leibe. “Visual Person Understanding through Multi-Task and Multi-Dataset Learning”. In: *Pattern Recognition*. Springer International Publishing, 2019, pp. 551–566 (cit. on p. 21).

[110]H. Pirsiavash and D. Ramanan. “Detecting Activities of Daily Living in First-Person Camera Views”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012, pp. 2847–2854 (cit. on pp. 13, 14, 23, 30, 31, 33, 34, 36, 79, 81, 85).

[111]Y. Poleg, C. Arora, and S. Peleg. “Temporal Segmentation of Egocentric Videos”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 2537–2544 (cit. on pp. 22, 23).

[112]Y. Poleg, A. Ephrat, S. Peleg, and C. Arora. “Compact CNN for Indexing Egocentric Videos”. In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2016, pp. 1–9 (cit. on pp. 15, 16, 79).

[113]R. Poppe. “A Survey on Vision-Based Human Action Recognition”. In: *Image and Vision Computing* 28.6 (2010), pp. 976–990 (cit. on pp. 13, 14).

[114]R. Possas, S. Pinto Caceres, and F. Ramos. “Egocentric Activity Recognition on a Budget”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 5967–5976 (cit. on pp. 15, 16, 49, 115).

[115]W. Price and D. Damen. “Retro-Actions: Learning ’Close’ by Time-Reversing ’Open’ Videos”. In: *IEEE International Conference on Computer Vision Workshop (ICCVW)*. Seoul, Korea (South): IEEE, 2019, pp. 1371–1380 (cit. on p. 116).

[116]K. Qian, W. Zhao, Z. Ma, et al. “Wearable-Assisted Localization and Inspection Guidance System Using Egocentric Stereo Cameras”. In: *IEEE Sensors Journal* 18.2 (2018), pp. 809–821 (cit. on pp. 11, 12).

[117]S.-A. Rebuffi, H. Bilet, and A. Vedaldi. “Efficient Parametrization of Multi-Domain Deep Neural Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT: IEEE, 2018, pp. 8119–8127 (cit. on p. 115).

[118]J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788 (cit. on p. 34).

[119]J. Redmon and A. Farhadi. “YOLO9000: Better, Faster, Stronger”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517–6525 (cit. on pp. 32, 34, 35).

[120]J. Redmon and A. Farhadi. “YOLOv3: An Incremental Improvement”. In: *CoRR abs/1804.02767* (2018) (cit. on pp. 45, 52, 53, 70).

[121]S. Ren, K. He, R. Girshick, and J. Sun. “Faster R-Cnn: Towards Real-Time Object Detection with Region Proposal Networks”. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. NIPS’15. Cambridge, MA, USA: MIT Press, 2015, pp. 91–99 (cit. on p. 45).

[122]X. Ren and C. Gu. “Figure-Ground Segmentation Improves Handled Object Recognition in Egocentric Video”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. 2010, pp. 3137–3144 (cit. on pp. 13, 79).

[123]X. Ren and M. Philipose. “Egocentric Recognition of Handled Objects: Benchmark and Analysis”. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2009, pp. 1–8 (cit. on pp. 13, 54, 55).

[124]N. Riche, M. Duvinage, M. Mancas, B. Gosselin, and T. Dutoit. “Saliency and Human Fixations: State-of-the-Art and Study of Comparison Metrics”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2013, pp. 1153–1160 (cit. on p. 74).

[125] M. S. Ryoo and L. Matthies. "First-Person Activity Recognition: What Are They Doing to Me?" In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013, pp. 2730–2737 (cit. on pp. 13, 14).

[126] M. S. Ryoo, B. Rothrock, and L. Matthies. "Pooled Motion Features for First-Person Videos". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 896–904 (cit. on p. 15).

[127] O. Sener and V. Koltun. "Multi-Task Learning As Multi-Objective Optimization". In: *32nd International Conference on Neural Information Processing Systems*. 2018, pp. 525–536 (cit. on pp. 19, 20, 75).

[128] L. Shen, S. Yeung, J. Hoffman, G. Mori, and L. Fei-Fei. "Scaling Human-Object Interaction Recognition Through Zero-Shot Learning". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. Lake Tahoe, NV: IEEE, 2018, pp. 1568–1576 (cit. on p. 116).

[129] Y. Shen, B. Ni, Z. Li, and N. Zhuang. "Egocentric Activity Prediction via Event Modulated Attention". In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 202–217 (cit. on pp. 16, 17).

[130] G. A. Sigurdsson, A. Gupta, C. Schmid, A. Farhadi, and K. Alahari. "Actor and Observer: Joint Modeling of First and Third-Person Videos". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 7396–7404 (cit. on p. 15).

[131] G. A. Sigurdsson, A. Gupta, C. Schmid, A. Farhadi, and K. Alahari. "Charades-Ego: A Large-Scale Dataset of Paired Third and First Person Videos". In: *arXiv:1804.09626 [cs]* (2018). arXiv: 1804.09626 [cs] (cit. on pp. 79, 81, 86, 95).

[132] G. A. Sigurdsson, G. Varol, X. Wang, et al. "Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding". In: *European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 510–526 (cit. on p. 79).

[133] K. Simonyan and A. Zisserman. "Two-Stream Convolutional Networks for Action Recognition in Videos". In: *27th International Conference on Neural Information Processing Systems*. 2014, pp. 568–576 (cit. on pp. 15, 65).

[134] S. Singh, C. Arora, and C. V. Jawahar. "First Person Action Recognition Using Deep Learned Descriptors". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2620–2628 (cit. on p. 16).

[135] A. Sinha, Z. Chen, V. Badrinarayanan, and A. Rabinovich. "Gradient Adversarial Training of Neural Networks". In: *arXiv:1806.08028 [cs, stat]* (2018). arXiv: 1806.08028 [cs, stat] (cit. on p. 116).

[136] L. N. Smith. "Cyclical Learning Rates for Training Neural Networks". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017, pp. 464–472 (cit. on pp. 59, 71, 87, 99).

[137] Leslie N. Smith. "A Disciplined Approach to Neural Network Hyper-Parameters: Part 1 - Learning Rate, Batch Size, Momentum, and Weight Decay". In: *arXiv:1803.09820 [cs, stat]* (2018). arXiv: 1803.09820 [cs, stat] (cit. on p. 59).

[138]S. Song, V. Chandrasekhar, N.-M. Cheung, et al. “Activity Recognition in Egocentric Life-Logging Videos”. In: *Asian Conference on Computer Vision Workshops (ACCVW)*. Vol. 9010. Springer International Publishing, 2015, pp. 445–458 (cit. on p. 13).

[139]S. Song, V. Chandrasekhar, B. Mandal, et al. “Multimodal Multi-Stream Deep Learning for Egocentric Activity Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2016, pp. 378–385 (cit. on pp. 16, 17).

[140]S. Song, N.-M. Cheung, V. Chandrasekhar, B. Mandal, and J. Liri. “Egocentric Activity Recognition with Multimodal Fisher Vector”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Shanghai: IEEE, 2016, pp. 2717–2721 (cit. on p. 13).

[141]K. Soomro, A. R. Zamir, and M. Shah. “UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild”. In: *arXiv:1212.0402 [cs]* (2012). arXiv: 1212.0402 [cs] (cit. on p. 79).

[142]E. H. Spriggs, F. De La Torre, and M. Hebert. “Temporal Segmentation and Activity Classification from First-Person Sensing”. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2009, pp. 17–24 (cit. on p. 13).

[143]A. Stergiou, G. Kapidis, G. Kalliatakis, et al. “Saliency Tubes: Visual Explanations for Spatio-Temporal Convolutions”. In: *IEEE International Conference on Image Processing (ICIP)*. 2019, pp. 1830–1834 (cit. on p. 67).

[144]A. Stergiou and R. Poppe. “Analyzing Human–Human Interactions: A Survey”. In: *Computer Vision and Image Understanding* 188 (2019), p. 102799 (cit. on p. 14).

[145]G. Strezoski, N. van Noord, and M. Worring. “Many Task Learning With Task Routing”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 1375–1384 (cit. on pp. 20, 105).

[146]Y.-C. Su and K. Grauman. “Detecting Engagement in Egocentric Video”. In: *European Conference on Computer Vision (ECCV)*. 2016, pp. 454–471 (cit. on p. 79).

[147]Y.-C. Su and K. Grauman. “Leaving Some Stones Unturned: Dynamic Feature Prioritization for Activity Detection in Streaming Video”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2016, pp. 783–800 (cit. on p. 19).

[148]S. Sudhakaran, S. Escalera, and O. Lanz. “Gate-Shift Networks for Video Action Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 1099–1108 (cit. on pp. 15, 97).

[149]S. Sudhakaran, S. Escalera, and O. Lanz. “LSTA: Long Short-Term Attention for Egocentric Action Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9954–9963 (cit. on pp. 17, 18, 20, 21, 73, 74, 97, 98).

[150]S. Sudhakaran and O. Lanz. “Attention Is All We Need: Nailing Down Object-Centric Attention for Egocentric Activity Recognition”. In: *British Machine Vision Conference (BMVC)*. 2018 (cit. on pp. 17, 18, 73, 74, 98).

[151]Y. Tang, Y. Tian, J. Lu, J. Feng, and J. Zhou. "Action Recognition in RGB-D Egocentric Videos". In: *IEEE International Conference on Image Processing (ICIP)*. 2017, pp. 3410–3414 (cit. on pp. 16, 53–55, 65, 79).

[152]H. R. Tavakoli, E. Rahtu, J. Kannala, and A. Borji. "Digging Deeper into Egocentric Gaze Prediction". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)* (2019), pp. 273–282 (cit. on p. 17).

[153]B. Tekin, F. Bogo, and M. Pollefeys. "H+O: Unified Egocentric Recognition of 3D Hand-Object Poses and Interactions". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, 2019, pp. 4506–4515 (cit. on pp. 15, 115).

[154]J. G. Teriús-Padrón, G. Kapidis, S. Fallmann, et al. "Towards Self-Management of Chronic Diseases in Smart Homes: Physical Exercise Monitoring for Chronic Obstruction Pulmonary Disease Patients". In: *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 2018, pp. 776–781 (cit. on p. 29).

[155]D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. "Learning Spatiotemporal Features with 3D Convolutional Networks". In: *IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 4489–4497 (cit. on pp. 15, 61).

[156]D. Tran, H. Wang, L. Torresani, et al. "A Closer Look at Spatiotemporal Convolutions for Action Recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 6450–6459 (cit. on p. 15).

[157]Z. Tu, W. Xie, Q. Qin, et al. "Multi-Stream CNN: Learning Representations Based on Human-Related Regions for Action Recognition". In: *Pattern Recognition* 79 (2018), pp. 32–43 (cit. on pp. 15, 65).

[158]G. Vaca-Castano, S. Das, and J. P. Sousa. "Improving Egocentric Vision of Daily Activities". In: *IEEE International Conference on Image Processing (ICIP)*. 2015, pp. 2562–2566 (cit. on p. 12).

[159]G. Vaca-Castano, S. Das, J. P. Sousa, N. D. Lobo, and M. Shah. "Improved Scene Identification and Object Detection on Egocentric Vision of Daily Activities". In: *Computer Vision and Image Understanding (CVIU)* 156 (2017). Image and Video Understanding in Big Data, pp. 92–103 (cit. on pp. 12, 17, 33, 35, 36, 80, 86, 87).

[160]L. Wang, Y. Xiong, Z. Wang, et al. "Temporal Segment Networks: Towards Good Practices for Deep Action Recognition". In: *European Conference on Computer Vision (ECCV)*. 2016, pp. 20–36 (cit. on pp. 15, 60, 62).

[161]X. Wang, R. Girshick, A. Gupta, and K. He. "Non-Local Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 7794–7803 (cit. on p. 15).

[162]X. Wang, Y. Wu, L. Zhu, and Y. Yang. "Symbiotic Attention with Privileged Information for Egocentric Action Recognition". In: *34th AAAI Conference on Artificial Intelligence*. 2020, pp. 12249–12256 (cit. on pp. 97, 98).

[163]M. Wray, D. Moltisanti, W. Mayol-Cuevas, and D. Damen. “SEMBED: Semantic Embedding of Egocentric Action Videos”. In: *European Conference on Computer Vision Workshops (ECCVW)*. 2016, pp. 532–545 (cit. on p. 15).

[164]C.-Y. Wu, C. Feichtenhofer, H. Fan, et al. “Long-Term Feature Banks for Detailed Video Understanding”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 284–293 (cit. on pp. 16, 17, 73, 97, 116).

[165]J. Wu, A. Osuntogun, T. Choudhury, M. Philipose, and J. M. Rehg. “A Scalable Approach to Activity Recognition Based on Object Use”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2007, pp. 1–8 (cit. on p. 19).

[166]Y. Wu and K. He. “Group Normalization”. In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 3–19 (cit. on p. 21).

[167]F. Xiao, Y. J. Lee, K. Grauman, J. Malik, and C. Feichtenhofer. “Audiovisual SlowFast Networks for Video Recognition”. In: *arXiv:2001.08740 [cs]* (2020). arXiv: 2001.08740 [cs] (cit. on pp. 16, 81, 97).

[168]Y. Yan, E. Ricci, G. Liu, and N. Sebe. “Egocentric Daily Activity Recognition via Multitask Clustering”. In: *IEEE Transactions on Image Processing* 24.10 (2015), pp. 2984–2995 (cit. on pp. 13, 20).

[169]R. Yonetani, K. M. Kitani, and Y. Sato. “Recognizing Micro-Actions and Reactions from Paired Egocentric Videos”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2629–2638 (cit. on pp. 11, 13, 15).

[170]H. Yu, M. Cai, Y. Liu, and F. Lu. “What I See Is What You See: Joint Attention Learning for First and Third Person Video Co-Analysis”. In: *ACM International Conference on Multimedia*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1358–1366 (cit. on p. 17).

[171]C. Zach, T. Pock, and H. Bischof. “A Duality Based Approach for Realtime TV-L1 Optical Flow”. In: *Pattern Recognition*. Springer Berlin Heidelberg, 2007, pp. 214–223 (cit. on p. 62).

[172]H. Zaki, F. Shafait, and A. Mian. “Modeling Sub-Event Dynamics in First-Person Action Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 7253–7262 (cit. on p. 16).

[173]A. Zamir, A. Sax, W. Shen, et al. “Taskonomy: Disentangling Task Transfer Learning”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 3712–3722 (cit. on p. 20).

[174]K. Zhan, S. Faux, and F. Ramos. “Multi-Scale Conditional Random Fields for First-Person Activity Recognition”. In: *IEEE International Conference on Pervasive Computing and Communications (PerCom)*. Budapest, Hungary: IEEE, 2014, pp. 51–59 (cit. on p. 13).

[175]M. Zhang, K. T. Ma, J. H. Lim, Q. Zhao, and J. Feng. “Anticipating Where People Will Look Using Adversarial Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.8 (2019), pp. 1783–1796 (cit. on p. 74).

[176] Z. Zuo, L. Yang, Y. Peng, F. Chao, and Y. Qu. “Gaze-Informed Egocentric Action Recognition for Memory Aid Systems”. In: *IEEE Access* 6 (2018), pp. 12894–12904 (cit. on pp. 13, 17).

List of Publications

G. Kapidis, R. Poppe, E. Van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. “Egocentric Hand Track and Object-Based Human Action Recognition”. In: *19th IEEE Conference on Ubiquitous Intelligence and Computing (UIC)*. Leicester, UK, 2019, pp. 922–929.

G. Kapidis, R. Poppe, E. van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. “Multitask Learning to Improve Egocentric Action Recognition”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2019, pp. 4396–4405.

G. Kapidis, R. Poppe, E. van Dam, L. P. J. J. Noldus, and R. C. Veltkamp. “Object Detection-Based Location and Activity Classification from Egocentric Videos: A Systematic Analysis”. In: *Smart Assisted Living*. Springer International Publishing, 2020, pp. 119–145.

G. Kapidis, R. Poppe, and R. C. Veltkamp. “Multi-Dataset, Multitask Learning of Egocentric Vision Tasks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence, Special Issue on Egocentric Perception* (2021).

G. Kapidis, R. W. Poppe, E. A. van Dam, R. C. Veltkamp, and L. P. J. J. Noldus. “Where Am I? Comparing CNN and LSTM for Location Classification in Egocentric Videos”. In: *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 2018, pp. 878–883.

G. Kapidis, E. van Dam, R. Poppe, L. P. J. J. Noldus, and R. C. Veltkamp. “Action Detection from Egocentric Videos in Daily Living Scenarios”. In: *Measuring Behavior 2018*. Manchester, UK, 2018, pp. 405–407.

A. Stergiou, G. Kapidis, G. Kalliatakis, et al. “Class Feature Pyramids for Video Explanation”. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2019, pp. 4255–4264.

A. Stergiou, G. Kapidis, G. Kalliatakis, et al. “Saliency Tubes: Visual Explanations for Spatio-Temporal Convolutions”. In: *IEEE International Conference on Image Processing (ICIP)*. 2019, pp. 1830–1834.

J. G. Teriús-Padrón, G. Kapidis, S. Fallmann, et al. “Towards Self-Management of Chronic Diseases in Smart Homes: Physical Exercise Monitoring for Chronic Obstruction Pulmonary Disease Patients”. In: *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 2018, pp. 776–781.

Appendix: Code Repositories

A

The code for our experiments is open-sourced and publicly accessible online.

- Chapter 3:
  `https://github.com/georkap/object-based-location-classification`
- Chapters 4 and 5:
  `https://github.com/georkap/hand_track_classification`
- Chapter 6:
  `https://github.com/georkap/ego_md_mtl`