

MULTI-MODAL EGOCENTRIC ACTIVITY RECOGNITION
THROUGH DECISION FUSION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

MEHMET ALI ARABACI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

JANUARY 2023

**Multi-modal Egocentric Activity Recognition
Through Decision Fusion**

submitted by **MEHMET ALI ARABACI** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Information Systems Department, Middle East Technical University** by,

Prof. Dr. Banu Günel Kılıç
Dean, **Graduate School of Informatics**

Prof. Dr. Altan Koçyiğit
Head of Department, **Information Systems**

Prof. Dr. Alptekin Temizel
Supervisor, **Modelling and Simulation, METU**

Assoc. Prof. Dr. Elif Sürer
Co-supervisor, **Modelling and Simulation, METU**

Examining Committee Members:

Prof. Dr. Altan Koçyiğit
Information Systems, METU

Prof. Dr. Alptekin Temizel
Modelling and Simulation, METU

Assoc. Prof. Dr. Aykut Erdem
Computer Engineering, Koç University

Assoc. Prof. Dr. Erhan Eren
Information Systems, METU

Assoc. Prof. Dr. Burkay Genç
Computer Engineering, Hacettepe University

Date: 18.01.2023

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname : Mehmet Ali Arabaci

Signature :

ABSTRACT

MULTI-MODAL EGOCENTRIC ACTIVITY RECOGNITION THROUGH DECISION FUSION

Arabacı, Mehmet Ali

Ph.D., Department of Information Systems

Supervisor: Prof. Dr. Alptekin Temizel

Co-Supervisor: Assoc. Prof. Dr. Elif Sürer

January 2023, 106 pages

The usage of wearable devices has rapidly grown in daily life with the development of sensor technologies. The most prominent information for wearable devices is collected from optics which produces videos from an egocentric perspective, called First Person Vision (FPV). FPV has different characteristics from third-person videos because of the large amounts of egomotions and rapid changes in scenes. Vision-based methods designed for third-person videos where the camera is away from events and actors, cannot be directly applied to egocentric videos. Therefore, new approaches, which are capable of analyzing egocentric videos and accurately fusing inputs from various sensors for specified tasks, should be proposed. In this thesis, we proposed two novel multi-modal decision fusion frameworks for egocentric activity recognition. The first framework combines hand-crafted features using Multi-Kernel Learning. The other framework utilizes deep features using a two-stage decision fusion mechanism. The experiments revealed that combining multiple modalities, such as visual, audio, and other wearable sensors, increased activity recognition performance. In addition, numerous features extracted from different modalities were evaluated within the proposed frameworks. Lastly, a new egocentric activity dataset, named Egocentric Outdoor Activity Dataset (EOAD), was populated, containing 30 different egocentric activities and 1392 video clips.

Keywords: first-person vision, egocentric activity recognition, multi-modality, decision fusion

ÖZ

KARAR TÜMLEŞTİRME YOLUYLA ÇOK-KİPLİ BİRİNCİ ŞAHIS HAREKET TANIMA

Arabacı, Mehmet Ali

Doktora, Bilişim Sistemleri Bölümü

Tez Yöneticisi: Prof. Dr. Alptekin Temizel

Ortak Tez Yöneticisi: Doç. Dr. Elif Sürer

Ocak 2023, 106 sayfa

Sensör teknolojilerinin gelişmesiyle birlikte giyilebilir cihazların günlük hayattaki kullanımı hızla artmıştır. Giyilebilir cihazlardaki en yaygın bilgi birinci şahıs görüşü olarak adlandırılan ve optik sensörler ile elde edilmiş birinci şahıs perspektife sahip videolardır. Birinci şahıs videolar büyük miktarda birinci şahıs hareketi içermeleri ve sahnelerdeki hızlı değişimler nedeniyle üçüncü şahıs videolarından farklı özelliklere sahiptir. Kameranın olaylardan ve aktörlerden uzak olduğu üçüncü şahıs videolarına göre tasarlanmış görsel tabanlı yöntemler birinci şahıs videolarına doğrudan uygulanamamaktadır. Bu nedenle, birinci şahıs videolarını analiz edebilen ve tanımlanan görevler için çeşitli sensörlerden gelen verileri doğru şekilde birleştirebilen yeni yaklaşımlara ihtiyaç duyulmaktadır. Bu tezde, birinci şahıs hareket tanıma problemi için çok-kipli karar tümleştirme kullanan iki yeni çatı önerilmiştir. Bunlardan ilki, üretilen öznelikleri Çoklu Kernel Öğrenmesi ile birleştirmektedir. Diğer çatı ise derin öznelikleri iki aşamalı karar tümleştirme mekanizması ile kullanmıştır. Gerçekleştirilen deneyler, görsel, işitsel ve diğer giyilebilir sensör bilgilerinin birleştirilmesinin birinci şahıs hareket tanıma performansını arttırdığını ortaya çıkarmıştır. Ek olarak, önerilen çatılar ile farklı kiplerden çıkarılan çok sayıda öznelik test edilmiştir. Son olarak, 30 farklı birinci şahıs hareketi ve 1392 video kayıt parçası içeren Egocentric Outdoor Activity Dataset (EOAD) isimli yeni bir birinci şahıs hareket veri seti oluşturulmuştur.

Anahtar Kelimeler: birinci şahıs görüşü, birinci şahıs hareket tanıma, çok-kipli, karar tümleştirme

To my wife and my daughter

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisors, Prof. Dr. Alptekin Temizel and Assoc. Prof. Dr. Elif Sürer for their endless support, understanding, patience, and supervision throughout my studies which was really influential in shaping my experiment methods and critiquing my results. I always appreciated their determination to realize this study, which once was no more than a fragment of our thoughts and hopes. I am glad to see that our ideas came to fruition, were acknowledged by others, and influenced further studies.

I would like to thank Assoc. Prof. Dr. Erhan Eren and Assoc. Prof. Dr. Aykut Erdem for their invaluable guidance at every stage of this research.

I am deeply grateful to my wife and my daughter; Gülperi and Derin Arabacı; for their understanding and support throughout my studies.

TABLE OF CONTENTS

ABSTRACT.....	iv
ÖZ.....	v
DEDICATION.....	vi
ACKNOWLEDGMENTS.....	vii
LIST OF TABLES	xi
LIST OF FIGURES	xiii
LIST OF ABBREVIATIONS.....	xv
CHAPTERS	
1 INTRODUCTION	1
1.1 Problem Definition.....	3
1.2 Purpose of the Study.....	5
1.3 Research Questions.....	6
1.4 Contributions of the Study.....	6
1.5 Organization of the Thesis.....	7
2 RELATED WORK	9
2.1 First Person Vision (FPV).....	9
2.2 Egocentric Action Recognition (EAR).....	10
2.2.1 Vision-based EAR.....	10
2.2.2 Multi-modal EAR.....	10
2.2.3 Deep Learning in EAR.....	11
2.2.4 Fusion-based Methods in EAR.....	12
2.2.5 Attention-based Methods in EAR.....	13

3	EGOCENTRIC ACTIVITY RECOGNITION USING MULTI-KERNEL LEARNING	15
3.1	Proposed Framework	15
3.1.1	Feature Extraction	16
3.1.1.1	Visual Features	16
3.1.1.2	Audio Features	18
3.1.1.3	Sensor Features	19
3.1.2	Activity Recognition	19
3.1.2.1	Single Kernel Learning	19
3.1.2.2	MKL	20
3.2	Experiments	22
3.3	Results	25
3.3.1	JPL Dataset	25
3.3.2	DogC Dataset	27
3.3.3	MEAD Dataset	27
3.3.4	Comparative Results	29
3.4	Discussion	32
3.4.1	Base Kernel Selection	32
3.4.2	Feature Selection	32
3.4.3	Analysis of the Overall Framework	35
4	EGOCENTRIC ACTIVITY RECOGNITION USING TWO-STAGE DECISION FUSION	37
4.1	The Proposed Framework	38
4.1.1	Stream Decision Fusion	40
4.1.1.1	Deep Feature-based Stream Fusion	41
4.1.1.2	VAEs-based Stream Fusion	47
4.1.2	Class-aware Segment Fusion	50
4.2	Experiments	53
4.2.1	Egocentric Outdoor Activity Dataset (EOAD)	53
4.2.2	Frame Extraction and Data Augmentation	58
4.2.2.1	RGB	58

4.2.2.2	Optical Flow	58
4.2.2.3	Audio	58
4.2.3	Dataset Splitting	62
4.2.4	EAR Model Training	64
4.2.5	Feature Fusion	64
4.2.6	Stream Decision Fusion	65
4.2.6.1	Deep Feature-based Stream Fusion	66
4.2.6.2	VAEs-based Stream Fusion	66
4.2.7	Class-aware Segment Fusion	66
4.3	Results	67
4.3.1	Single Stream	67
4.3.2	Ideal Case for Stream and Segment Fusion	68
4.3.2.1	Stream Fusion	68
4.3.2.2	Segment Fusion	72
4.3.2.3	Stream and Segment Fusion	72
4.3.3	Model-based Stream Fusion	74
4.3.3.1	Deep Feature-based Stream Fusion	74
4.3.3.2	VAEs-based Stream Fusion	75
4.3.4	Class-aware Segment Fusion	78
4.3.5	Stream and Segment Fusion	80
4.3.6	Comparative Results	81
4.4	Discussion	82
5	CONCLUSION AND FUTURE WORK	85
	REFERENCES	89
	CURRICULUM VITAE	105

LIST OF TABLES

Table 1	Typical storage size for one-year lifelogging video recording (6 hours/day)	3
Table 2	Comparison of TAR and EAR	4
Table 3	Comparison of MKL and TSDF-based EAR frameworks	6
Table 4	F1-score of single features and their combinations for JPL (MKL-based EAR)	26
Table 5	F1-score of single features and their combinations for DogC (MKL-based EAR)	27
Table 6	F1-score of single features and their combinations for MEAD (MKL-based EAR)	29
Table 7	Comparative performances of MKL-based EAR	31
Table 8	Time complexity analysis of MKL-based EAR	32
Table 9	Regression types for stream and segment weighing	40
Table 10	Weight generation of streams for training (4-class problem)	41
Table 11	Prediction error formulations used for VAE	51
Table 12	Weight generation of segments for training (4-class problem)	52
Table 13	Summary of the most relevant egocentric action recognition datasets	54
Table 14	EOAD activity videos summary	56
Table 15	Dataset splitting for EOAD	63

Table 16	Deep feature fusion formulations	65
Table 17	Deep feature scalars	65
Table 18	Single stream performances for RGB, Flow, and Audio	68
Table 19	Ideal case for stream weighing	71
Table 20	Ideal case for segment weighing	72
Table 21	Ideal case for stream and segment weighing	73
Table 22	Deep feature-based stream weighing results	75
Table 23	EAR performances with deep feature-based stream weighing	76
Table 24	VAEs-based stream weighing results	76
Table 25	VAEs-based stream weighing results for different latent dimensions	78
Table 26	EAR performances with segment weighing	80
Table 27	EAR performances using stream and segment weighing	81
Table 28	Comparative performances of the proposed framework with Stacking Emsemble and MKBoost	81

LIST OF FIGURES

Figure 1	Number of citations and publications over time, related to FPV and wearable sensor analysis (until October 2022).	2
Figure 2	Stream and segment decisions for video clips.	5
Figure 3	The proposed solution for EAR using visual, audio, and sensor features [1]	16
Figure 4	Confusion matrices of SVM, Histogram Intersection, MKBoost and SimpleMKL learning methods for JPL.	26
Figure 5	Confusion matrices of SVM, Histogram Intersection, MKBoost, and SimpleMKL learning methods for DogC.	28
Figure 6	Confusion matrices of linear SVM, MKBoost, and SimpleMKL algorithms for MEAD [1].	30
Figure 7	The number of selected base kernels for each dataset [1].	33
Figure 8	The number of selected features for each dataset [1].	33
Figure 9	The number of selected feature combinations of JPL (<i>a</i>), DogC (<i>b</i>) and MEAD (<i>c</i>) for the given feature composition color codes (<i>d</i>) [1].	34
Figure 10	Recognition accuracy vs stream weight factor [2].	37
Figure 11	Stream and segment decisions for video clips.	38
Figure 12	The phases of the proposed weighing solution.	39
Figure 13	Simple linear regression with one independent variable.	39
Figure 14	Deep feature-based stream weighing.	42

Figure 15	Gradient boosting algorithm.	45
Figure 16	NGBoost algorithm [3].	46
Figure 17	Ordinary gradient descent and NGBoost regression sample outputs [3].	46
Figure 18	Illustration of an auto-encoders [4].	47
Figure 19	Autoencoder and variational autoencoder differences [4].	48
Figure 20	Using auto-encoders as the content generator [4].	48
Figure 21	The training of the proposed VAE regressor.	50
Figure 22	Class-aware segment fusion	52
Figure 23	EOAD activities.	57
Figure 24	Sample feature frames for RGB.	59
Figure 25	Sample feature frames for Optical Flow (left: horizontal, right: vertical).	60
Figure 26	Sample LMS feature frames for Audio.	62
Figure 27	ResNet50 architecture for RGB stream	64
Figure 28	Confusion matrices for single streams at segment level	69
Figure 29	Confusion matrices for single streams at video-level	70
Figure 30	<i>False</i> to <i>True</i> state transitions for ideal case.	73
Figure 31	Predicted and target weights for VAE regressor: training set (RGB (a) and Flow (b)) and validation set (RGB (c) and Flow (d)).	77
Figure 32	The prediction and target segment weights for train (left) and validation (right) samples	79

LIST OF ABBREVIATIONS

ARM	Activity Recognition Model
BERT	Bidirectional Encoder Representations from Transformers
BoVW	Bag-of-Visual-Words
CNNs	Convolutional Neural Networks
DogC	DogCentric Activity Dataset
DTs	Decision Trees
EAR	Egocentric Activity Recognition
EOAD	Egocentric Outdoor Activity Dataset
FF	Frequency-Domain Feature
FN	False Negative
FP	False Positive
FPV	First-Person Vision
FTMAF	Fourier Transform of Motion Direction Access Frame
FTMPF	Fourier Transform of Grid Motion Per-Frame
GMM	Gaussian Mixture Model
GOFF	Grid Optical Flow-based Features
GPS	Global Positioning System
HAR	Human Activity Recognition
HOF	Histogram of Optical Flows
HOG	Histogram of Oriented Gradients
IMUs	Inertial Measurement Units
KL	Kullback-Leibler

KNN	K-Nearest Neighbor
LASSO	Least Absolute Shrinkage and Selection Operator
LMS	Log-Mel Spectrogram
Log-C	Log-Covariance
LSTM	Long Short-Term Memory
MBH	Motion Boundary Histograms
MDHF	Motion Direction Histogram Features
MDHSF	Motion Direction Histogram Standard-Deviation Feature
MEAD	Multi-modal Egocentric Activity Dataset
MFCCs	Mel-Frequency Cepstral Coefficients
MKL	Multi-Kernel Learning
MMHF	Motion Magnitude Histogram Features
NGBoost	Natural Gradient Boosting
OLS	Ordinary Least Square
PCA	Principal Component Analysis
RBFs	Radial Basis Functions
RFID	Radio-Frequency Identification
RNNs	Recurrent Neural Networks
SAN	Stream Attention Network
SVMs	Support Vector Machines
SVR	Support Vector Regressor
TAR	Third-Person Activity Recognition
TN	True Negative
TP	True Positive
TSDF	Two-Stage Decision Fusion
UBM	Universal Background Model

VAEs	Variational Auto-Encoders
VATT	Video-Audio-Text Transformer
ViF	Virtual Inertia Features
ViT	Vision Transformer
VR	Virtual Reality
XGBoost	eXtreme Gradient Boosting
ZC	Zero-Crossing

CHAPTER 1

INTRODUCTION

Nowadays, wearable devices have started to attract more attention with the evolution of mobile devices and sensor technologies. Practical and affordable products have been presented in different categories such as hand-held cameras, sports and action cameras, wearable camera glasses, smartwatches, fitness trackers, smart clothing, wearable medical devices, and virtual reality (VR) headsets. Using different hardware, software, and sensors, it is possible to build applications for different prominent sectors including business operations (logistics, information access), security/safety (military, identity recognition), medical (activities of daily living to assist caregivers to track elderly people [5], vital sign monitoring, chronic disease management, brain/eye movement), wellness (physiological monitoring, gait/posture correction), sport/fitness (sports performance, fitness monitoring, virtual coaching), lifestyle computing (interactive gaming, shared experience), communication (interactive group, social interactions, physical expression), glamor (decorative display, reactive response), or recommendation systems (recommendation of music genres according to the type of activities [6]).

Optics and audio are the most commonly used sensors for wearable devices generating audio-visual information. Besides optic cameras and microphones, additional embedded sensors (e.g., eye tracking sensors, accelerometers, gyroscopes, depth sensors, magnetometers, light sensors, proximity sensors, body-heat detectors, and temperature sensors) are also used to get information about the users. The concept of audio-visual information taken through a wearable camera is called Egocentric Vision or First-Person Vision (FPV). All these various types of sensors can be used for different tasks such as action/activity recognition [1, 7–12], object/hand recognition [13, 14], action/gaze anticipation [15–17], egocentric video summarization [18, 19], social interactions [20, 21], human-object interaction [22, 23], pose estimation [24–26], or privacy in egocentric videos [27, 28].

Human-centric characteristic of egocentric sensor information offers great potential and challenges to present new approaches to some of these tasks. In Figure 1, the number of studies in egocentric vision and wearable sensors are listed annually according to their relevance to the keywords for “first-person vision”, “first-person video”, “egocentric video”, “lifelogging video”, “wearable video”, and “wearable sensor” on Web of Science. The graph reflects the research potential of egocentric video-sensor analysis since the number of articles per year has exploded in the last few years. The reason for that is possibly related to the increase of affordable commercial products on the market which makes it easier to acquire first-person vision and sensor data than before.

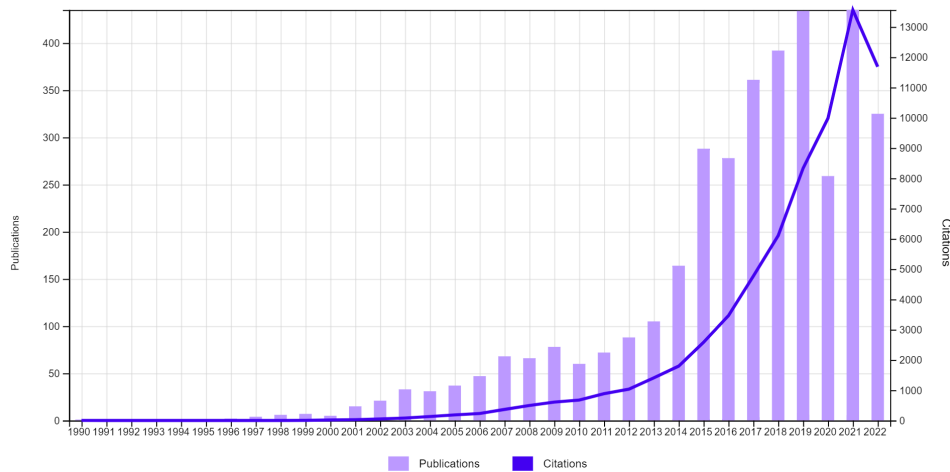


Figure 1: Number of citations and publications over time, related to FPV and wearable sensor analysis (until October 2022).

Although these wearable devices are employed for various tasks, a group of studies focused on the recognition of daily activities for users. It was shown that physical inactivity increases the risk of cardiovascular diseases, diabetes, breast cancer, and mental disorders such as depression [29]. Therefore, it is important to analyze the daily activities of users and warn them of the early stages of possible diseases. However, Egocentric Action Recognition (EAR) systems suffer from storing large amounts of video-sensor data since the resolutions of videos have increased and the number of sensors has diversified over time. One of the most challenging tasks here is using these data collectively to improve egocentric activity analysis, such as fast querying through hundreds of egocentric videos taken every day. Even though information obtained from all these sensors allows us to analyze egocentric activities in a variety of ways, this necessitates efficient algorithms capable of combining complementary sensor information effectively and providing modularity to facilitate the inclusion of additional sensors.

Publications that concentrate on EAR can be reviewed based on their sensor types: using only visual information [7, 8, 30], using mobile and wearable sensor networks [31–33], and combining visual information with other wearable sensors [15, 34–36]. Until recently, most of the vision-based activity recognition methods have been designed for a third-person perspective. In that case, actors are usually far away from the camera and do not participate in the activities or events [8]. Although vision is the primary source of information in FPV tasks, other types of wearable sensors are also available including audio, accelerometers, gyroscopes and Global Positioning Systems (GPS) [37].

Before proceeding to examine EAR approaches, it is better to define *action* and *activity*. In some studies [8, 31, 38], the terms *action* and *activity* were used interchangeably while the others [39–41] defined activity (or complex activity) as a collection of multiple actions (or atomic activities) that are identified over temporal snippets using pattern-mining algorithms.

This study’s focus was restricted to atomic ego-activities (e.g., walking, running, shaking hands, hugging, or driving) rather than complex activities composed of *verbs* and *nouns* (e.g., prepare pizza, take a banana, or pour milk). Therefore, egocentric activities are differentiated among video clips without establishing a time relationship between activities in this study. Additionally, there are no restrictions on processing videos in real-time, and we did not use any activity segmentation techniques as a preprocessing step.

Following are brief explanations of the EAR problem, the objective of this thesis, our research topics, and the significance of this study.

1.1 Problem Definition

The proliferation of wearable devices on the market has resulted in large amounts of acquired data, posing the challenge of analyzing them collected by these devices. In general, the difficulty is to efficiently process sensor data and make it usable for consumers. Nevertheless, the nature of the problem may vary depending on the uniqueness of the applications. In some cases, like interactive games or shared experiences, sensor data may need to be analyzed in real-time. Or one needs to combine different sensor information (multi-modality) to extract the key objects or events for the users (e.g., employing a front camera with an eye tracker). These types of use cases force researchers to develop novel FPV analysis methods.

As previously noted, storing egocentric video-sensor data is troublesome, particularly for lifelogging egocentric videos. Because lifelogging videos represent the recording of the users’ daily lives in varying degrees of detail, the resulting audio-visual data may be enormous. Table 1 illustrates an example of the total quantity of data acquired for one year utilizing various video resolutions and audio bitrates ¹. The total amount of audio-video data indicates that storage capacity is a significant problem. Improving video compression (or encoding) techniques is one viable option to reduce storage capacity. Removing redundant data from video footage is a second viable option. However, the redundancy of audio-visual content highly depends on subjective criteria. As a result, developing efficient and effective FPV algorithms capable of extracting meaningful information from tasks is critical in this domain.

Table 1: Typical storage size for one-year lifelogging video recording (6 hours/day)

Resolution	Video		Audio	
	Bitrate (Mbps)	Datasize (TByte)	Bitrate (Mbps)	Datasize (GByte)
480P	2.5	2.46	128	126
720P	5.0	4.93	384	378
1080P	8.0	7.88	384	378
4K	40.0	39.42	384	378

The fusion of information coming from different sensors (e.g., optic, audio, accelerometer, inertial measurement units) to recognize egocentric activities is still an active research area.

¹ It was assumed that the user would record their daily activities for 6 hours/day and the bitrates are taken from YouTube’s recommended settings for standard quality uploads.

Although the increase in sensor diversity brings out the need for adaptive fusion, there is a limited number of studies that uses multiple sensors in the EAR. In addition to the complexity of building effective fusion techniques, the temporal dynamics of the activities also require regulating sensors’ data to synchronize them in time. At this point, it is better to mention the main differences between egocentric activity recognition (EAR) and third-person activity recognition (TAR) listed in Table 2.

Table 2: Comparison of TAR and EAR

	TAR	EAR
Viewpoint	Usually far away from events and actors	The observer is involved in the events
Motion Characteristics	Mostly does not suffer from rapidly changing and uncontrollable visual content	Large amount of ego-motion such as spinning and falling down
Sensor Types	Video, audio	Rich sensor networks (video, audio, mobile, and wearable sensors)
Content	Surveillance videos, news, movies, sport videos, and documentaries	Daily activities, action videos, and lifelogging videos

Vision-based methods for activity recognition are typically designed for third-person videos and cannot be directly applied to FPV. Video analysis from a third-person perspective is generally inadequate when the observer is personally involved in the events. Since the camera undergoes a lot of ego-motion in FPVs according to the activities of the user, the motion information of actors (i.e., the head movement for head-mounted cameras or body movement for chest-mounted cameras) is as important as that of foreground objects in videos. Additionally, the lighting, location (indoors, outdoors), and ego-motion rapidly change the visual content. Third-person videos taken from wider angles tend to have a broader perspective than egocentric videos.

Employing a single model does not necessarily produce high accuracy for all activities, given that their characteristics vary according to their nature. Consequently, it was worthwhile to study the possibility of developing fusion mechanisms for modalities weighing them relative to their information levels. Thus, the proposed methods may dynamically weigh different modalities (i.e., appearance, motion, audio, or sensor) and discriminate activities with more precision.

In this research, we proposed two EAR frameworks that can be applied to multi-modal information. The first framework combines multi-modal information with Multi-Kernel Learning (MKL), a well-known method in ensemble learning. MKL offers an implicit fusion strategy that performs feature selection and classification simultaneously. The second framework, on the other hand, employs a two-stage decision fusion (TSDF) technique that adaptively weights the decisions of single-stream EAR models based on their relevance to activities. Unlike MKL, TSDF provided an explicit fusion technique with a confidence-based weighing mechanism. Even though the two proposed frameworks are expandable with new sensors, the

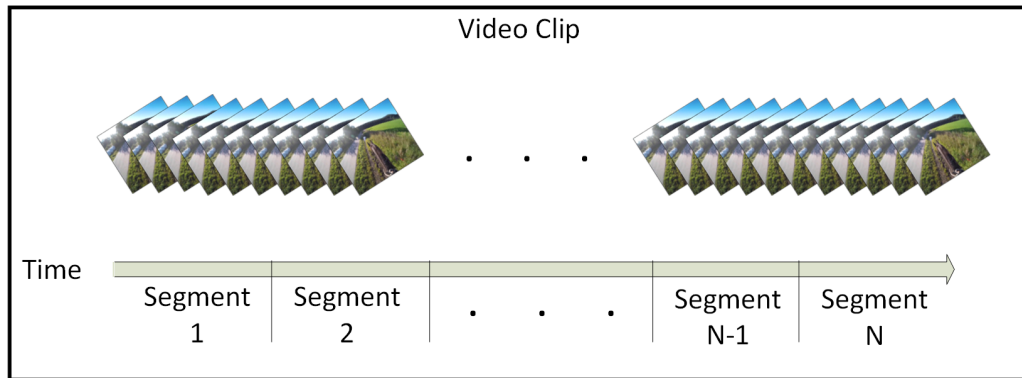


Figure 2: Stream and segment decisions for video clips.

addition of new modalities differs due to the nature of their fusion methods. For example, MKL needs to retrain its weak learners after reselecting features and kernel sets. TSDF, on the other hand, requires training a single-stream EAR model for the new modality (no need if previously trained) and training fusion models that generate decision weights afterward.

After testing both frameworks, we discovered several advantages and disadvantages of them compared to each other. For instance, MKL’s recognition performance on various egocentric activity datasets was encouraging. However, choosing a universal feature and kernel set for all datasets is challenging. On the other hand, it is easier to add new modalities for TSDF since it only requires pre-trained EAR models and modality features to train fusion models. Fusion models receive modality features as inputs and decision weights computed by the confidences of the pre-trained EAR models as targets. In addition, TSDF allows the use of heterogeneous EAR models (e.g., SVM and deep models) since the fusion models require only the confidence scores of activities. Nevertheless, the TSDF approach has two limitations. Firstly, it is obvious that EAR models should generate confidence scores, not just categorical outcomes. Secondly, the features of all modalities for each video segment should be available for training fusion models. Here, we need to clarify two terms which are video *clips* and *segments*. Video *clips* refer to activity videos that include a single activity in them and consist of hundreds or thousands of frames. On the other hand, video parts that are uniformly divided according to a predetermined number are called video *segments* (Figure 2).

Table 3 depicts the comparison of MKL and TSDF-based EAR frameworks. The details of the proposed frameworks will be discussed in the following chapters.

1.2 Purpose of the Study

This research aims to establish novel, adaptive, and expandable EAR frameworks utilizing multi-modal information. In the context of this work, adaptive means that the framework is expected to alter the weights of the particular modalities based on their contribution to the recognition performance. Here, we speculate that cross-modalities complement each other to discriminate egocentric activities. In addition to this, one of the other goals is to simplify the process of adding new modalities to the proposed frameworks.

Table 3: Comparison of MKL and TSDF-based EAR frameworks

	MKL-based EAR	TSDF-based EAR
Fusion Strategy	Implicit: using feature selection and classification	Explicit: using confidence-based decision fusion pipeline
Expandability	Reselect feature and kernel set	Train single-stream EAR model for new modality
	Retrain MKL model	Retrain fusion models
Stream Types for Model Training	Multi-stream for MKL	Single-stream for EAR models
		Multi-stream for fusion models
Advantages	Adaptiveness for different datasets	Easy to integrate new modalities
		Allows using of heterogeneous EAR models together (i.e., SVM, deep models)
Limitations	Hard to select universal feature and kernel set	EAR models should be calibrated and produce confidence scores

1.3 Research Questions

In accordance with the purpose of the study, the following questions will be explored in this study:

Question 1: Can Multi-Kernel Learning procedures be applied to fuse optical, audio, and wearable sensor data for EAR tasks?

Question 2: Is it possible to develop an EAR framework utilizing an explicit fusion strategy using confidence-based weights generated by pre-trained single-stream EAR models?

1.4 Contributions of the Study

This research proposed two novel frameworks to fuse multi-modal data from various sensors. The proposed framework recognizes egocentric activities using audio, video, and wearable sensors. According to our knowledge, this is one of the first studies to use the audio sensor in conjunction with visual data and other mobile sensors to investigate this particular issue. By altering the relative importance of the modality information, we present flexible frameworks that can balance modalities from different sources and adapt to a variety of scenarios involving distinct activity classes. The MKL-based technique provides an effective fusion procedure that maximizes recognition performance even if one or more sensor information is absent. Additionally, TSDF-based design allows easy integration of new modalities without needing to train the whole framework again. Through this research, we also publish a new egocentric activity dataset named Egocentric Outdoor Activity Dataset (EOAD) containing 30 different

activities. The results indicate that the proposed frameworks yield comparable or superior results compared to the state-of-the-art techniques.

We published the following papers and proceedings from this study:

- M. A. Arabacı, F. Özkan, E. Surer, P. Jančovič, A. Temizel, "Multi-modal egocentric activity recognition using multi-kernel learning," *Multimedia Tools and Applications*, vol. 80, no. 11, 16299-16328, 2021 [1].
- F. Özkan, M. A. Arabacı, E. Surer, A. Temizel, "Boosted multiple kernel learning for first-person activity recognition," in *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1050-1054. IEEE, 2017 [7].

1.5 Organization of the Thesis

Chapter 2 provides a comprehensive overview of egocentric action/activity recognition approaches. In Chapter 3, the proposed methodology for the MKL-based approach is provided, along with its system-level architecture and some implementation-specific information. In Chapter 4, the TSDF-based strategy for identifying egocentric actions/activities is described. A new egocentric activity dataset, the Egocentric Outdoor Activity Dataset (EOAD), which consists primarily of outdoor egocentric actions, is also presented in Chapter 4. Lastly, Chapter 5 summarizes the overall findings of this study and provides a future direction for further research studies.

CHAPTER 2

RELATED WORK

There is a strong connection between first-person vision and the development of egocentric activity recognition (EAR) literature. Therefore, the following section provides a brief summary of the first-person vision. After that, EAR-related studies will be discussed in detail.

2.1 First Person Vision (FPV)

The earliest examples of FPV research literature date back to the 1990s. Several experiments using wearable devices were presented in one of the earliest seminal works [42] on this topic. Following that, MediaLab (MIT) demonstrated the possibilities of FPV by playing "Patrol," a real-time space game [43]. One of the most promising studies [44] detected the strong relationship between eye movements and FPV by demonstrating that the eyes are typically focused on the same object throughout the action. There have also been commercial interests in FPV, and several products have been introduced, including SenseCam by Microsoft Research in 2006, GoPro Hero in 2010, and Google Glass Project in 2012.

Increasing attention to the FPV domain triggered some of the research areas consisting of egocentric action recognition [1, 8, 9], scene understanding from egocentric videos [45, 46], video summarization and retrieval [18, 19], eye gaze tracking and analysis with head-mounted cameras [47], social saliency and multi-user analytics [23], egocentric action anticipation [48–50], attention and saliency for understanding human-object interaction [51]. On the other hand, several emerging topics are expected in the near future, such as personalization of visual analysis, socio-behavioral modeling, understanding group dynamics and interactions, egocentric video as big data, and first-person vision for robotics.

FPV provides additional information compared to third-person videos, such as recording the essential parts of the scene [52], rapidly expanding egocentric video datasets with the new commercial wearable devices, scene recognition using illumination and global scene characteristics [53], and estimating significant objects by monitoring eye and head movements [54]. While FPV analysis has certain practical benefits, it also presents several challenges that must be overcome [55]. FPV analysis should deal with problems such as ego-motion, highly variable and uncontrollable videos due to rapidly changing scenes and time (i.e., day, night, indoor, outdoor), and real-time processing constraints for specific applications. Besides, it should also be efficient while processing videos to optimize the battery life of the devices.

2.2 Egocentric Action Recognition (EAR)

Even though there exist diverse research topics in the FPV domain, in this thesis, we will focus on activity recognition from egocentric videos. Hence, this section provides a summary of the works that are associated with the EAR. Within this scope, the studies related to the EAR are grouped according to their input (e.g., vision, or multi-modal), or method (e.g., deep learning, fusion, or attention) types.

2.2.1 Vision-based EAR

Vision-based activity detection algorithms primarily analyze videos captured from a third-person perspective [56–59]. Vision-based EAR can be categorized into the object and motion-based techniques [51]. In object-based methods, activity recognition is performed using the object(s) detected in videos (i.e., detection of cheese and bread objects imply "making cheese sandwich" activity) [60], which makes them dependent on the availability of objects in particular actions (thus can only be used to detect actions involving specific objects) and directly related to the object recognition performance, which is susceptible to occlusions. Methods that focus on objects to recognize egocentric activities have made it necessary to examine how people interact with those objects [61, 62]. In addition, hand segmentation [13, 14] has become a significant problem at a low level, as the accuracy of the EAR is tied to the segmentation performance [63, 64].

Motion-based techniques rely on the premise that different activities, such as *running*, *walking*, *stair climbing*, and *writing*, require distinct body motions and that these motion patterns can be utilized to identify activities [30, 65]. In [66], the authors utilized first-person dense trajectories in the motion pyramidal structure. The relative strengths of motion along the trajectories were then utilized to generate multiple bag-of-words descriptions that were later merged into a single action descriptor. In another study, motion features in egocentric videos were used to extract biometric data, and users could be accurately recognized from a few seconds of walking-related video [67]. In [8], a structural learning approach was used with HOF, Log-C, and cuboid features to discriminate interaction-related actions, such as *hugging* and *throwing objects*.

2.2.2 Multi-modal EAR

Processing of multi-modal sensor data to solve pre-defined tasks has some challenges, such as cross-modality regulation/synchronization, developing efficient fusion algorithms, or needing expertise for the selected input types. However, additional data from different sensor sources have the potential to obtain complementary information to describe the activities. Therefore, using multi-sensor data has become one of the promising strategies for EAR instead of a single modality. For that purpose, many embedded sensors on mobile and wearable devices were utilized to identify the user's activities. Each sensor type delivers essential information about an activity. For instance, motion sensors can monitor the user's movements to identify the motion patterns of various activities, such as *walking*, *standing*, and *running*. Other types

of sensors, such as the accelerometer, gyroscope, magnetometer, and inertial measurement units, can also derive this motion pattern (IMUs).

In [68, 69], the audio information was utilized for EAR in addition to visual information. The objects outfitted with radio-frequency identification (RFID) tags were also used for action recognition [70, 71], while some other works [72, 73], combined RFID tags with visual information. On the other hand, accelerometer data allow us to discriminate different physical activities [29, 74]. In addition, the information gathered by proximity and light sensors indicates whether the actor is in a dark or bright environment [75]. Pedometer sensors or specialized wearable gadgets that count steps and monitor heart rate or pulse can also provide important information regarding the health issues of their users [76]. To create and pick acceptable characteristics for a specific human activity recognition (HAR) system, however, several studies in this subject require significant heuristic expertise [37].

Only a few studies deal with the problem of combining information from optical and wearable sensors for EAR. In [15], a head-mounted camera and an eye tracker were used to recognize objects the actor interacted with in videos, which were then used to recognize the actor's activities. Similarly, in [34], EAR was accomplished using a camera coupled with a wearable eye tracker to acquire gaze measurements in which a 2-D image point in each frame represented the gaze location. A multi-modal approach combining new sensor features with dense trajectory features [35] was published and applied to their publicly available dataset (Multi-modal Egocentric Activity Dataset) [36]. An early study evaluated audio information with optical information for detecting human activities from a wearable camera and microphone [69].

Audio has not been extensively studied in the EAR field, with few datasets containing RGB frames and audio. However, it is a potential alternative for specific actions where visual appearance and motion are insufficient. For example, [1] suggested combining audio-visual features with MKL and MKBoost. Using the training set, MKL was used to learn the weights of various features, kernels, and their parameters. Concerning the features, the authors considered employing complementary features from different modalities: from videos, they extracted global features using the Grid Optical-Flow-based Features (GOFF), Vision-based inertial features (VIF) (both from [30]), and Log-Covariance (Log-C) features [77]; local features from videos leveraging Cuboids [57]; and, for audio features, they utilized Mel-frequency cepstral coefficients (MFCCs) [78]. In addition to the MKL and MKBOOST, an SVM was utilized for the classification stage. In another work [79], temporal-binding (the combination of modalities within a range of temporal offsets) of RGB, optical-flow and, audio was accomplished with a multi-modal architecture and demonstrated that audio is complementary to the appearance and motion representation of the RGB and optical-flow inputs.

2.2.3 Deep Learning in EAR

Even though hand-crafted features produce promising results for task-specific activity recognition problems, the difficulties of detecting complex activities still require additional study. Recent research has demonstrated that deep learning approaches can extract high-level representations on spatial and temporal dimensions, making them more suitable for complicated activity identification. In addition, deep networks enable the features to be automatically learned rather than manually developed. Deep learning-based algorithms can be utilized as an

alternative for various activity recognition subtasks, such as feature extraction [80–82], kernel fusion [83], and human-object interaction exploration [84].

Recently, many algorithms based on deep learning have been applied to the EAR. Feature extraction [81, 82, 85], kernel fusion [83], and human-object interaction exploration [86] are some of the subtasks that employ deep neural networks for activity detection. Another study [82] utilized auto-encoders to extract appearance and motion features from raw input and recreate it using its decoding technique. Then, the learned appearance and motion features produced an egocentric activity representation that can be easily fed to supervised learning models for activity recognition. One of the most influential research [65] suggested a 3D CNN architecture for long-term activity identification in egocentric movies by generalizing the concept of sparse optical flow volume-based temporal filtering [87]. A twin-stream network architecture was employed in a different study [88]. One stream analyzed appearance, and the other analyzed motion information by explicitly training the network to segment hands and localize objects to distinguish egocentric actions. Recent research [2] presented a two-stream CNN architecture that uses long-term fusion pooling operators to capture the temporal structure of actions by exploiting a succession of frame-wise appearance and motion elements in actions.

Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs)-based hybrid deep learning models have become state-of-the-art in image and video captioning research [89–91]. CNNs can capture the spatial relationship, whereas RNNs can utilize the temporal relationship. Combining CNNs and RNNs could thereby improve the ability to extract video features and transfer them to a sentence, which is likewise a sequence like a video.

In [92], following 4000 tests on sensor-based Human Activity Recognition (HAR) datasets, RNNs and Long Short-Term Memory (LSTM) were recommended to recognize short activities with the natural order, and CNNs were recommended for recognizing long-term repetitive activities. According to a separate study [93], LSTM-based techniques cannot describe long-lasting activities due to their rising complexity. In other words, deep learning models may obtain high recognition performance for a subset of tasks while receiving poorer performance ratings for the remainder.

2.2.4 Fusion-based Methods in EAR

Fusion of modalities may occur at the feature [94, 95] or classifier [96, 97] level. Before the final classification, feature-level fusion combines various features to produce more discriminative features. In [98], appearance and motion (generating a single feature vector) were processed by a 3D CNN, while a Faster Region-based Convolutional Neural Network was utilized for the object features. The detector’s features were fused with the motion and appearance features individually by applying learned gate weights (from the opposite branch). Prior to collecting the final feature vector for a branch, an extra step of attention was applied to make use of the features from the opposite branch. Another work [99] employed a non-linear fusion technique in which weighted addition was performed to combine shareable and distinctive components (obtained through non-linear mapping of the original features).

On the other hand, classifier (or decision) level fusion techniques use each individual feature independently in the classification process. The final decision is taken by combining decisions for individual features. In [100] a late fusion mechanism was introduced for three modalities of data: RGB (for appearance, using a Batch Normalised Inception), Optical-Flow (for motion, using a Temporal Segment Network), and object features (confidence scores obtained from an object detector) to anticipate actions.

2.2.5 Attention-based Methods in EAR

With the recent surge of interest in deep neural networks, attention-based models have shown promising results on a variety of complex tasks, including image and video captioning [101, 102], machine translation and handwriting synthesis [103, 104], image recognition (e.g., Street View House Numbers dataset [105]), game-playing and tracking [106], and visual question answering [107–110]. The visual attention mechanism aims to identify relevant regions in visual information and concentrate on these parts to extract information, mimicking the human perception and line of thinking while completing specific tasks. Various tasks, such as object recognition [105] and action recognition [93, 111–115], utilized visual attention techniques.

Attention models proposed for EAR incorporate object and egocentric cues using pre-processed inputs such as hand mask, homography [85], gaze prediction [116], and localized objects [88]. For spatial attention models gaze behavior is important since it reflects a person’s thinking process, represents human attention [54], and is coordinated with egocentric actions. Attention models in vision are divided into two categories: bottom-up attention and top-down attention [117, 118]. The bottom-up attention includes human attention when they are performing free-viewing on a scene or an image, in which the objects or regions that “stand out” relative to the neighboring parts (saliency) attract human attention. In comparison, human attention when they are performing certain tasks (e.g., object manipulation) belongs to the category of top-down attention, which is task-driven [116].

Attention models are also defined as soft attention models and hard attention models. In attention models, a probability distribution over a grid of features is first predicted to indicate the level of attention on each region. The soft attention models use the attention distribution to re-weight the features, while the hard attention models select the feature with the highest probability to represent the data. Both soft and hard attention models are explored in [102] to generate image captions. The soft attention model is trained using back-propagation and the hard attention model is trained using a reinforcement algorithm.

Attention mechanisms proposed for egocentric activity recognition generate spatial [119], temporal [93, 120], or spatio-temporal [113, 121] attention maps. The spatial attention mechanism attempts to identify representative visual regions for the selected task. Temporal attention is utilized to weigh the frame-level characteristics according to their importance in identifying actions. Finally, spatio-temporal attention maps are employed to choose frames and areas simultaneously.

Recently, convolution-free methods compete with CNNs on image recognition tasks [122–124]. In [125, 126], transformers were applied to video understanding tasks such as ac-

tion recognition and EAR. In one of the recent studies, [127], a multi-modal framework, Video-Audio-Text Transformer (VATT), was proposed that processes raw inputs (video, audio, text) applied on various tasks such as video action recognition, audio event classification, image classification, and text-to-video retrieval. VATT mainly kept the architecture of Bidirectional Encoder Representations from Transformers (BERT) [128] and Vision Transformer (ViT) [129] except for the separate usage of tokenization and linear projection layers for each modality.

CHAPTER 3

EGOCENTRIC ACTIVITY RECOGNITION USING MULTI-KERNEL LEARNING

This chapter describes the specifics of egocentric activity recognition using an MKL-based technique. We propose an EAR system that combines optical, audio, and wearable sensor data with multi-modal characteristics [1]. The suggested framework is capable of adaptable feature weighting and is expandable with additional features and modalities.

The fusion of modalities may occur at the feature level [36, 94, 95] or classifier level [96, 97]. Before the final classification, feature-level fusion combines distinct types of features to produce more discriminative features. Classifier (or decision) level fusion approaches, on the other hand, utilize each feature separately in the classification process. The ultimate decision is made by combining the decisions for each feature. Using two MKL approaches (MKBoost [130] and SimpleMKL [131]), we extract and combine diverse sets of characteristics collected from visual, audio, and wearable sensor data. SimpleMKL employs decision-level fusion to pick kernel weights using a weighted two-norm (L2) regularization that promotes sparse kernel combinations. MKBoost, on the other hand, combines a boosting strategy with MKL learning to enable concurrent feature selection and decision-level fusion.

The following section explains the details of the proposed MKL-based framework.

3.1 Proposed Framework

The suggested multi-modal framework based on MKL is depicted in Figure 3. First, features are retrieved from each sensor's raw data. Then, each extracted feature is utilized as an input to the MKL algorithm which simultaneously performs feature selection and classification. Weak learners are used in order to determine the optimal feature and kernel combinations following MKL training. Lastly, the EAR is executed using the trained model for test videos with previously selected features and kernels.

The visual, audio, and sensor aspects that were utilized in this study are described in the subsequent section. Additionally, the following section discusses the single and multi-kernel learning methodologies.

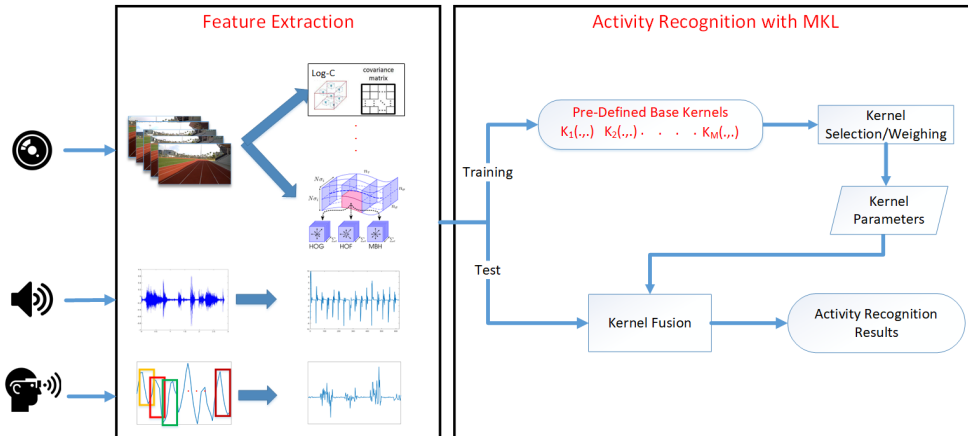


Figure 3: The proposed solution for EAR using visual, audio, and sensor features [1]

3.1.1 Feature Extraction

Various types of features were retrieved from visual, audio, and sensor data in this study. When picking the characteristics, three significant criteria were taken into account. One of them is to accommodate the variety of characteristics that may contain supplementary data regarding egocentric actions. Second, given that our primary objective is not to offer a novel feature, existing visual, audio, and sensor features that have been effectively used on EAR problems in the past were favored. In order to evaluate the effectiveness of the suggested learning technique with other state-of-the-art methods, a similar collection of features was chosen using the same datasets.

The following section describes the specifics of all visual, audio, and sensor features employed in this framework. The selection of feature sets for distinct datasets is discussed in Section 3.2.

3.1.1.1 Visual Features

Effective encoding of ego-motion is essential for EAR systems [30, 65], therefore, we chose a set of visual features (GOFF, VIF, HOG, HOF, and MBH) that maintains motion patterns globally and locally in the temporal dimension. Grid Optical Flow-based Features (GOFF) are motion-based video features taken from spatio-temporal information and intended exclusively for FPVs [30]. Virtual Inertia Features (VIF) [30] are used to model egocentric activities by approximating inertia data (velocity and acceleration). Log-Covariance (Log-C) [77] characteristics are dense video characteristics obtained from optical flow data and intensity gradient. The cuboid feature utilizes sparse 3D space-time data to extract local information [132]. Dense trajectory features are a set of visual features (Trajectory, Histogram of Oriented Gradients (HOG), Histogram of Optical Flows (HOF), and Motion Boundary Histograms (MBH)) that present an effective solution for motion-related vision problems by recognizing motion patterns over densely recorded sample locations utilizing optical flow fields.

Grid Optical Flow-based Features (GOFF):

GOFF is utilized to model the distinguishable motion patterns within optical flow information such as amplitude, direction, and frequency [30]. A set of features, including Motion Magnitude Histogram Features (MMHF), Motion Direction Histogram Features (MDHF), Motion Direction Histogram Standard-Deviation Feature (MDHSF), Fourier Transform of Motion Direction Access Frame (FTMAF), and Fourier Transform of Grid Motion Per-Frame (FTMPF), are created using video frames divided into grids to identify the motion characteristics of activities.

MMHF is the histogram representation of 15-level non-uniform quantization of grid optical flow magnitude values [30]. MDHF is another histogram form of grid optical flow when its direction values are quantized. MDHF was quantified evenly into 36 levels, with 10° between each level. MDHSF is a 36-dimensional vector that represents the standard deviation of each direction bin along the temporal dimension. FTMAF is a frequency-based characteristic that assesses the variance throughout the temporal dimension for each direction bin using decomposed frequency bands. FTMAF, unlike MDHSF, assigns 25 levels to the detailed dynamics of motion direction. FTMPF examines the variance of grid optical flow within a 25-level frame. After concatenating all of the sub-features, GOFF's resulting feature vector has a size of 137.

Virtual Inertia Feature (VIF):

Without actual inertial sensors, VIF transmits virtual inertial information derived from the intensity centroid across frames in a video [30]. Three distinct sub-features were extracted in the temporal dimension: zero-crossing (ZC), 4MEKS, and frequency-domain feature (FF). ZC calculates the zero-crossing rates of velocity and acceleration data using velocity and acceleration values derived from the intensity centroid for each frame. The time-domain properties of 4MEKS reveal each inertial signal's minimum, maximum, median, mean, standard deviation, energy, kurtosis, and average. The FF feature stores low-frequency variations in acceleration and velocity. Ten frequency components were selected for this study. Much like GOFF, the VIF sub-features were combined to create a 106-dimensional feature vector.

Log Covariance (Log-C):

A dense set of localized features can be effectively represented using the feature covariance matrix. With the aid of feature covariance matrices, local features can be represented in a lower dimension. In this study, the feature covariance matrix was determined by using optical flow and gradient vectors. The intensity gradient of raw video sequences was used to determine the temporal direction and first-order partial derivative of optical flow for spatial x and y directions, as well as the spatial divergence, vorticity, gradient tensor, and the rate of strain tensor for each pixel of a video frame [77].

Only the dimension of the feature vectors is connected to the dimension of the covariance matrix (i.e., 12×12 in this study). The Riemannian manifold on which covariance matrices are located was transformed into a Euclidean manifold using the matrix logarithm [133]. The size of the feature vector was reduced to 78 due to its symmetry. Each video clip's feature vectors were grouped using k-means after being normalized by standard deviation. Bag-of-Visual-Words (BoVW) is then used to define a descriptor for each unique activity video. With

a predetermined range of cluster sizes, a dictionary size of 300 produced the best classification results for a single feature. However, except for the classifiers utilizing histogram intersection kernels, which will be covered in more detail in the following section, principal component analysis (PCA) was used for the descriptor to reduce the dimension of sparse BoVW vectors.

Cuboid:

Local video features include a sparse 3D XYT space-time feature called cuboid [132] in addition to the global video features. Prior to [57], cuboids had been successfully used for activity recognition tasks. In order to take into account both the spatial and temporal dimensions, the cuboid feature was developed as an alternative to 2D interest point detectors.

Before the feature extraction process, an interest point detector found the corners in space and time by reacting strongly to the movement of small areas in space and time. At each point of interest, the brightness gradient and optical flow information were used to make a cuboid feature [132]. The cuboid was set up to generate descriptors using BoVW, like Log-C. The histogram size was set to 500 using a set of cluster sizes determined by the classification performance of each feature. PCA reduces the dimension, except for the kernels at the histogram intersection.

Dense Trajectory Features:

By recognizing motion patterns over densely tracked sample points using optical flow fields, dense trajectories provide an excellent solution for motion-related vision challenges. They were also utilized to represent ego motions in egocentric videos [36], comprising a set of visual characteristics including trajectory, HOG, HOF, and MBH. The information regarding a trajectory is merely the concatenation of normalized displacement vectors. HOG focuses on static appearance information, whereas HOF and MBH measure video motion information.

The Fisher vectors were employed to encode the dense trajectory descriptors using an approximated Gaussian Mixture Model (GMM) extracted as described in [59]. The cluster size for the GMM was 25, and 1% of the descriptors were randomly chosen to estimate the model while creating the codebook. Using PCA, the dimensions of the features are reduced by half. The power and L2 normalization processes were then applied to the Fisher vectors.

3.1.1.2 Audio Features

The utterance of an activity audio recording needs to be translated into a vector space to integrate video, audio, and sensor modalities using SVM and MKL-based frameworks. This was achieved by using a well-established methodology for speaker identification from the speech [134, 135] in this study.

The first step divides an audio signal into frames, and a spectrum-based feature vector represents each frame. Mel-frequency cepstral coefficients (MFCCs) [78] were utilized as frame characteristics for this purpose. Each frame was subjected to the discrete Fourier transform. The resulting magnitude spectrum was transferred to a bank of Mel-spaced triangular filters, followed by the discrete cosine transform, which yielded MFCCs. Before settling on the configuration, we investigated a range of parameter values: frame length of 40 ms, 10 ms

shift between subsequent frames, and 23 filter-bank channels. The initial 12 MFCCs were used. The frame energy was included as the 13th feature. These features were appended with their temporal derivatives, referred to as the delta and delta-delta coefficients, calculated as described in [136] using the span of +3 and +2 frames, respectively. This produced a 39-dimensional representation of each signal frame's features.

With diagonal covariance matrices, the Gaussian mixture model (GMM) was used to model the distribution of these feature vectors. Initially, a class-independent model, known as the Universal Background Model (UBM), was estimated utilizing all training data from all classes. Using class-specific training data, a class-dependent GMM is generated by performing a maximum a-posteriori adaptation [137] of the component mean vectors of the UBM. The mean vectors of the resultant class-dependent GMM components are then concatenated to generate a 'supervector' [138]. Each utterance of each class is assigned a supervector, resulting in a set of supervectors per class. Then, supervectors are employed as vector representations of each class to classify activities. Using varied numbers of GMM components ranging from 16 to 64 resulted in comparable performance.

Consequently, the number of components is set to 16 throughout the trials. Using the PCA, the dimension reduction of the supervectors was evaluated. However, applying PCA to the supervectors had no noticeable impact on the results.

3.1.1.3 Sensor Features

The data from the wearable sensors (accelerometer, gravity, gyroscope, linear acceleration, magnetic field, and rotation vector) include 19 dimensions of time-series information. In this research, each dimension of sensor data was regarded as a one-dimensional signal and translated into trajectories using sliding windows. Fisher encoding was executed similarly to dense trajectory features after creating numerous trajectories from sensor data. We employed the same feature extraction process and parameter settings as in [36].

3.1.2 Activity Recognition

Support vector machines (SVMs) were selected as the baseline approach representing single kernel learning, and they were compared to two well-known MKL algorithms: MKBoost and SimpleMKL. MKBoost employs boosting to solve a variant of the MKL problem, thereby avoiding the need to perform complex optimization tasks [130]. In contrast, SimpleMKL proved to be a fast and effective converging approach compared to other MKL optimization algorithms [139]. The subsection that follows briefly addresses the specifics of these strategies in connection to the EAR problem.

3.1.2.1 Single Kernel Learning

SVM is a kernel-based approach, and using kernels enables operation in feature spaces with greater dimensions than the original. In this study, SVM was employed in order to learn a

single kernel with features created by concatenating all individual features. The most common kernel types for support vector machines are linear, polynomial, and radial basis functions (RBFs). After conducting multiple experiments, linear and polynomial kernels (3rd order) yielded the best recognition results for two pre-specified feature sets applied to egocentric datasets defined in Chapter 3.2.

The linear and polynomial kernels used in this study are defined as:

$$\begin{aligned}\kappa(x_i, x_j) &= \langle x_i, x_j \rangle \\ \kappa(x_i, x_j) &= (\langle x_i, x_j \rangle + l)^p\end{aligned}\tag{1}$$

where κ represents kernel function, x 's are the features, p is the maximal order of monomials making up the new feature space and l is a bias towards lower order monomial. This kernel definition's underlying premise is that creating new features as byproducts of existing ones is frequently advantageous [140].

As a result of using the BoVW model for Log-C and cuboid, one of the feature sets includes histogram-based features. In addition to polynomial kernels, a modified version of the histogram intersection kernel (DC-Int) [38] was selected because it was explicitly built for histogram-based features:

$$\kappa(x_i, x_j) = \exp\left(-\sum_{c=1}^C D(H_i^c, H_j^c)\right)\tag{2}$$

where C is the number of channels, H_i^c and H_j^c are W dimensional histograms of c^{th} channel for i^{th} and j^{th} videos and $D(H_i^c, H_j^c)$ is the histogram distance defined as:

$$D(H_i^c, H_j^c) = 1 - \left(\sum_{m=1}^W \min(h_{im}, h_{jm}) / \sum_{m=1}^W \max(h_{im}, h_{jm})\right)\tag{3}$$

where h_{im} and h_{jm} are the m^{th} histogram bins identified for i^{th} and j^{th} videos, respectively.

3.1.2.2 MKL

SVM is an efficient technique for solving classification and regression problems [141] where the data representation is determined implicitly by the kernels $\kappa(x, x_i)$. MKL transforms the single kernel solution into the weighted sums of several kernels, as seen below:

$$\sum_{i=1}^N \alpha_i^* \kappa(x, x_i) + b\tag{4}$$

where N is the number of samples, α_i^* and b are coefficients to be learned from examples, while $\kappa(\cdot, \cdot)$ is a given positive definite kernel associated with a reproducing kernel Hilbert space. Multiple kernels have been shown to improve the interpretability of a decision function

and its performance [142],

$$\kappa(x, x^i) = \sum_{k=1}^K d_k \kappa_k(x, x_i) \quad (5)$$

where K is the number of kernels, $d_k \geq 0$ and $\sum_{k=1}^K d_k = 1$.

MKBoost:

MKBoost employs a boosting framework to learn an ensemble of multiple single-kernel base kernel classifiers. Using a similar technique to Adaboost [143], the kernel and classifier combination weights can be efficiently computed through the boosting learning process [130]. In this method, specific kernel classifiers (κ) with multiple kernels (K) are repeatedly learned using a subset of M examples ($r * M$ where $0 < r < 1$) via a series of boosting trials $t = 1, \dots, T$, where T signifies the total number of boosting trials.

The MKBoost algorithm was initially developed for binary classification tasks, despite being applied to multi-class situations in our framework. For this aim, multi-class classifiers perform classification tasks at each trial, and samples within each class are boosted to maintain class balance. In addition, the feature selection procedure has been updated to examine all feature combinations (P) for each trial (i.e., 7 combinations for 3 features). This procedure’s pseudocode is described in Algorithm 1.

At each boosting trial, the weight distribution χ_t , is changed to reflect the relative value of the training instances for learning. Additionally, the weights of incorrectly classified instances are increased while the others are reduced to effectively prioritize examples that are difficult to classify.

The selected kernels and feature combinations with their corresponding weights are utilized during the testing phase. The final prediction for test samples is based on the weighted total of the kernel predictions for each test sample.

SimpleMKL:

SimpleMKL provides an answer to MKL by employing a weighted l_2 normalization. The proposed technique is founded on a gradient descent wrapping standard SVM solver [131] that calculates the kernel combination.

SimpleMKL consists of two basic processes, as described in [144]: solving a standard SVM optimization problem with the given kernel weights d and updating kernel weights using the gradient calculated with another parameter (γ) obtained in the first step. In addition, the gradient update procedure must consider kernel weights’ non-negative and normalization properties [144]. In this algorithm, K denotes the number of kernels, κ the base kernel, d_k the kernel weights, J the differentiable objective function, and ∇_D the gradient descent directions at each step. In Algorithm 2, the pseudo-code for SimpleMKL is presented.

Algorithm 1 MKBoost

Input: $(x, y), \kappa, T$ **Output:** \hat{y}

- 1: training set (S_{train}): $(x_1, y_1), \dots, (x_M, y_M)$
 - 2: test set (S_{test}): $(x_{M+1}, y_{M+1}), \dots, (x_N, y_N)$
 - 3: labels: $y = 1, \dots, L$
 - 4: feature combinations: $p = 1, \dots, P$
 - 5: kernel pool: $\kappa_k(\cdot, \cdot) : X \times X \rightarrow \mathbb{R}$ where $k = 1, \dots, K$
 - 6: **Training Phase**
 - 7: **for** $t \leftarrow 1$ to T **do**
 - 8: Select $r * M$ sample indices (i_t) using distribution χ_t where $0 < r < 1$
 - 9: **for** $p \leftarrow 1$ to P **do**
 - 10: Select p^{th} feature combination for training: $S_{train}^p[i_t]$
 - 11: **for** $k \leftarrow 1$ to K **do**
 - 12: Train weak classifier (κ_k) with $S_{train}^p(i_t)$
 - 13: Compute training error over all samples (S_{train}^p): $\epsilon_p^k = \frac{1}{M} \sum_{m=1}^M f_k(x_m^p) \neq y_i$
 - 14: Select the best classifier for p^{th} feature combination : $\epsilon_p^k = \arg \min_k \epsilon_p^k$
 - 15: Select the best classifier (f_t) and feature combination (p_t) for trial : $\epsilon_t = \arg \min_p \epsilon_p^k$
 - 16: Set the weight for trial: $W_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
 - 17: Update sample distribution: $\chi_{t+1}(i) = \chi_t(i) \begin{cases} e^{-W_t} & \text{if } \kappa_t(x_i) = y_i \\ e^{W_t} & \text{if } \kappa_t(x_i) \neq y_i \end{cases}$ for $i = 1, \dots, M$
 - 18: $\chi_{t+1} = \frac{\chi_{t+1}}{Z_t}$ where Z_t is a normalization factor to make χ_{t+1} a distribution
 - 19: **Test Phase**
 - 20: **for** $i \leftarrow M + 1$ to N **do**
 - 21: $\hat{Y}_c \leftarrow 0$ where $c = 1, \dots, C$
 - 22: **for** $t \leftarrow 1$ to T **do**
 - 23: Predict the label for trial: $c_t = f_t(S_{test}^{p_t}[i])$
 - 24: $\hat{Y}[c_t] \leftarrow \hat{Y}[c_t] + W_t * c_t$
 - 25: Predict the final label: $\hat{y}_i = \arg \max_c \hat{Y}_c$
-

3.2 Experiments

Three egocentric datasets were used to evaluate the effectiveness of the proposed framework: JPL First-Person Interaction [8], DogCentric Activity Dataset (DogC) [38], and Multi-modal Egocentric Activity Dataset (MEAD) [31]. We have taken into account the variety of activities when selecting the datasets. For instance, the movies in JPL were recorded indoors with a passive actor, but the videos in DogC have an animal perspective. On the other hand, MEAD contains footage of human actors captured in diverse environments (indoor and outdoor) and at different times of the day.

JPL [8] consists of first-person footage of eight actors engaging in interaction-level tasks. For each actor, there are four positive (i.e., friendly) interactions (shaking hand, hugging, petting,

Algorithm 2 SimpleMKL

Input: d_k **Output:** κ_k

- 1: $d_k \leftarrow \frac{1}{K}$ for $i=1, \dots, K$
 - 2: **while** stopping criterion not met **do**
 - 3: Compute $J(d)$ by using an SVM solver with $\kappa = \sum_k d_k \kappa_k$
 - 4: Compute $\frac{\partial J}{\partial d_k}$ for $k = 1, \dots, K$ and descent direction ∇_D
 - 5: Set $\mu = \operatorname{argmax}(d_k)$, $J^\dagger = 0$, $d^\dagger = 0$, $\nabla_D^\dagger = \nabla_D$
 - 6: **while** $J^\dagger < J(d)$ **do**
 - 7: Set kernel weights and gradient descent: $d = d^\dagger$, $\nabla_D = \nabla_D^\dagger$
 - 8: $v = \operatorname{arg} \min_{(k | \nabla_D^k < 0)} \left(-\frac{d_k}{\nabla_D^k}\right)$, $\gamma_{max} = -\frac{d_v}{\nabla_D^v}$
 - 9: Update parameters: $d^\dagger = d + \gamma_{max} \nabla_D$, $\nabla_D^{\mu^\dagger} = \nabla_D^\mu - \nabla_D^v$, $\nabla_D^{v^\dagger} = 0$
 - 10: Compute J^\dagger by using an SVM solver with $\kappa = \sum_k d_k^\dagger \kappa_k$
 - 11: Linear search along D for $\gamma \in [0, \gamma_{max}]$ (calls an SVM solver for each trial value)
 - 12: $d \leftarrow d + \gamma \nabla_D$
-

waving hand), one neutral (pointing) interaction, and two negative (i.e., hostile) interactions (punching, throwing things). There are 84 videos with a resolution of 320x240 at 30 frames per second. The clips vary in length, with an average of 7.77 seconds.

DogC [38] covers 10 distinct types of activities from the dog’s perspective. Video resolutions are 320x240 at either 24 or 48 frames per second, and the average clip length is 4.12 seconds. Some activities include playing with a ball, drinking, feeding, looking left/right, patting, and shaking. Unlike the other datasets, the amount of videos for each activity differs (feed and shake have 25 videos while playing with a ball has just 14 videos), making it imbalanced in terms of its class samples.

MEAD [31] is unlike previous egocentric activity datasets since it contains multi-modal sensor data (optic, audio, accelerometer, gravity, gyroscope, linear acceleration, magnetic field, and rotation vector). Twenty life-logging activities are categorized into four categories: ambulation, daily activities, office work, and exercise. Each category has 10 clips, which are exactly 15 seconds long. There are 200 videos with 1280x720 resolution at 29.9 frames per second. The audio was captured at a frequency of 48 kHz with 16 bits per sample. No meaningful information was detected at higher frequencies, so the audio was downsampled to 24 kHz before feature extraction.

The feature sets were chosen based on the sensor information available in the datasets, the diversity of features (i.e., GOFF holds the global motion in frames while VIF presents video-based inertia information), and the features utilized by other state-of-the-art algorithms. Since JPL and DogC only had visual sensor information, only visual features were used. Additionally, GOFF and VIF were utilized for the first time in [30] with JPL, cuboid was proposed in [8] for JPL, and cuboid and Log-C were employed in [7] for both JPL and DogC. Therefore, a mixture of GOFF, VIF, Log-C, and cuboid was chosen as the feature set to demonstrate that the proposed method can effectively combine various visual aspects. In contrast, MEAD contains visual, audible, and wearable sensor data. In [31], dense trajectory and sensor char-

acteristics were recovered from MEAD; these features are also chosen for use in the current study. In contrast to [31], however, GMM-based supervector audio features were extracted and obtained by modeling MFCCs to demonstrate that it is possible to add the modality to the proposed MKL-based framework and increase the overall recognition performance. As a result, the suggested framework was assessed in two settings: one employs visual characteristics (GOFF, Log-C, cuboid) and virtual inertia (VIF) extracted from JPL and DogC, while the other uses visual (dense trajectory), audio, and sensor (FVS) information extracted from MEAD.

The kernel types employed for single kernel learning vary based on the chosen feature set. For example, the polynomial kernel of the third order was chosen for the first experimental setting consisting of GOFF, VIF, Log-C, and cuboid. In contrast, the linear kernel was selected for the second feature set consisting of FVS, audio, and dense trajectory information. The characteristics of the features have a direct bearing on the selection of various kernel types. GOFF, VIF, Log-C, and cuboid store data in a compact manner that typically necessitates non-linear decision bounds. However, FVS, audio, and dense trajectory information were encoded with Fisher vectors, which contain sparse vectors that are often separable using linear kernels.

A single feature vector in our tests represented each video clip. Consequently, the number of samples equals the number of videos in the datasets. Training and test sets were generated at random at each iteration, and the final assessment results were determined by averaging 100 test iterations. Each JPL exercise has nine training films and three test movies, while each MEAD activity includes eight training videos and two test videos. DogC, unlike JPL and MEAD, has a variable amount of video clips for each activity, of which around 75% are selected for training.

Before doing tests with MKL algorithms, a kernel pool should be predefined. However, looking for the optimal kernel set for each combination of features is time-consuming. Thus, only one set of basis kernels was chosen for each dataset, and all trials were conducted using the same pool of basis kernels.

As stated in (6), Precision (P), Recall (R), Accuracy (A), and F1-score (F) metrics were used to evaluate the findings, taking into account True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) scores.

$$\begin{aligned}
 P &= \frac{TP}{TP + FP} & R &= \frac{TP}{TP + FN} \\
 A &= \frac{TP + TN}{TP + TN + FP + FN} & F &= 2 \frac{P * R}{P + R}
 \end{aligned} \tag{6}$$

As an additional performance measurement metric, kappa statistics were chosen, a well-known discriminating statistical tool to evaluate the classification accuracy of different classifiers that generally provides better interclass discrimination than overall accuracy [145]. Using the marginal probabilities of the ground truth, kappa statistics predict labels whose probabilities add up to the values of the confusion matrix.

Below is the formulation of Kappa statistics:

$$p_0 = \sum_{i=1}^L p_{ii} \quad p_e = \sum_{i=1}^L p_i * p_i \quad \hat{\kappa} = \frac{p_0 - p_e}{1 - p_e} \quad (7)$$

where L is the number of classes, p_i is the probability of i^{th} class according to the ground truth, $p_{\hat{i}}$ is the probability of i^{th} class according to the prediction, p_0 is the observed accuracy and p_e is the sum of the marginal proportions.

The suggested system was mostly built with MATLAB 2018b, and OpenCV 3.2.0 was used to estimate optical flow. In addition, the following third-party toolboxes were employed: LibSVM 0.9.20 [146], the toolbox for cuboid extraction created by Dollar et al. [132], and the SimpleMKL toolbox [144]. The LibSVM and SimpleMKL toolboxes have been changed so that histogram intersection kernels can be used.

The evaluation process was identical for all datasets. The performance of each dataset’s characteristics was evaluated. The feature combinations were analyzed to determine the impact of feature types on the recognition scores. Each dataset’s results were tabulated individually. The performance of the proposed method was assessed utilizing all of its aspects and compared to state-of-the-art methods.

3.3 Results

In this section, we will present the results taken for 3 datasets (JPL, DogC, and MEAD) as well as comparative results with the other state-of-the-art methods.

3.3.1 JPL Dataset

Table 4 displays the average F1-scores for single features and their combinations in the JPL dataset. SimpleMKL works better for optical flow-based features, while the histogram intersection kernel performs better for histogram-based features, as determined by the results (Log-C and cuboid). The lower performance of the local visual feature compared to the global visual feature is another significant point.

The JPL videos were captured with a passive actor, and video clips are devoid of foreground items (only one or two persons appear in the videos). There are two standard motion features in videos: global camera motion (e.g., punching, hugging) and local motion occurring just in one location inside the field of view (e.g., throwing things, pointing), while the other regions lack motion information. As evidenced by the outcomes, these motion characteristics benefit the global features.

When multiple global features are combined, the combinations that include GOFF consistently receive the most significant results. This is expected given that GOFF is the most discriminative feature based on the performance of individual features. When all characteristics (global and local) are employed, however, MKBoost and SimpleMKL algorithms outperform SVM-based classifiers.

Figure 4 depicts the resultant confusion matrices for JPL activities using the selected classifiers. According to these findings, pet and point activities have the lowest recognition rates compared to other activities. In addition, MKBoost produces a better-balanced recognition performance, making it more trustworthy than its competitors.

Table 4: F1-score of single features and their combinations for JPL (MKL-based EAR)

		SVM-Poly	SVM-Hist	MKBoost	SimpleMKL
Single Features					
Global	GOFF	0.91	0.91	0.92	0.92
Global	VIF	0.85	0.86	0.86	0.87
Global	Log-C	0.80	0.84	0.82	0.82
Local	Cuboid	0.69	0.71	0.70	0.70
Combination of Global Features					
GOFF + VIF		0.92	0.92	0.93	0.93
GOFF + Log-C		0.90	0.87	0.93	0.92
VIF + Log-C		0.84	0.86	0.85	0.85
GOFF + VIF + Log-C		0.91	0.89	0.93	0.93
Combination of Global and Local Features					
Global + Local		0.92	0.92	0.93	0.93



Figure 4: Confusion matrices of SVM, Histogram Intersection, MKBoost and SimpleMKL learning methods for JPL.

3.3.2 DogC Dataset

DogC is an outdoor, imbalanced dataset comprised of animal perspectives. Additionally, ego-motion is far greater than JPL and MEAD, making it the most difficult dataset. Table 5 displays the average F1-scores for both single-feature and combination performances for the DogC dataset. In contrast to JPL, the performance gap between global and local visual elements is relatively minimal, indicating that local motion characteristics are equally as relevant as global motion. In addition, MKBoost and SimpleMKL have scores for global feature combinations that are remarkably close. However, SimpleMKL obtains the best performance when combining global and local features.

Table 5: F1-score of single features and their combinations for DogC (MKL-based EAR)

		SVM-Poly	SVM-Hist	MKBoost	SimpleMKL
Single Features					
Global	GOFF	0.56	0.59	0.59	0.61
Global	VIF	0.42	0.46	0.47	0.47
Global	Log-C	0.47	0.51	0.53	0.49
Local	Cuboid	0.43	0.36	0.41	0.41
Combination of Global Features					
	GOFF + VIF	0.60	0.61	0.63	0.63
	GOFF + Log-C	0.59	0.63	0.62	0.63
	VIF + Log-C	0.53	0.52	0.56	0.55
	GOFF + VIF + Log-C	0.62	0.63	0.63	0.63
Combination of Global and Local Features					
	Global + Local	0.64	0.62	0.63	0.65

Figure 5 depicts the confusion matrices of DogC for various learning strategies. Since they are frequently mistaken, "looking left" and "looking right" behaviors have the lowest categorization accuracy. These two activities share many similarities, except for their motion directions.

3.3.3 MEAD Dataset

MEAD has a greater number of egocentric activity classes and is more complex. In addition, this dataset's videos demonstrate greater variety because they were shot at different times of day and locations (indoor and outdoor). In addition, MEAD has access to multi-modal sensor information, including visual, audio, and wearable sensors. Therefore, the combination of characteristics was chosen based on their primary modalities to provide the results (video, audio, and sensor).

Table 6 displays the outcomes of the experiment. Specifically, dense trajectory features demonstrated their ability to differentiate activities by modeling the ego-motions in films

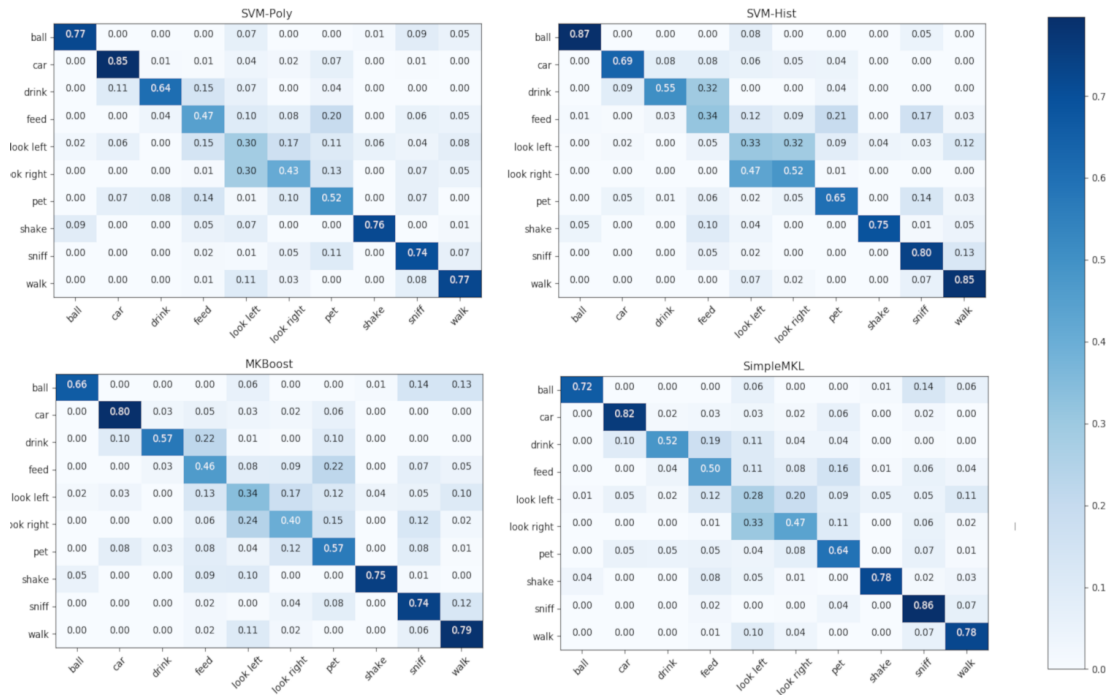


Figure 5: Confusion matrices of SVM, Histogram Intersection, MKBoost, and SimpleMKL learning methods for DogC.

with encouraging results in single-feature trials. The single-feature performance of classifiers is highly similar. Therefore, it is impossible to determine the relative performance of classifiers based on the performance of a single feature.

New modalities often improve final identification performance, except when the sensor and audio combinations are employed in conjunction with linear SVM. For instance, the best single-feature sensor and audio performances are 62% and 46%, respectively. When these two features are integrated with MKL, their performance score increases by as much as 69%, indicating that sensor and audio modalities provide complementary information for activities. Combining distinct modalities improved the performance of MKL-based learning algorithms in general. The difference between the maximum accuracy of individual features and the accuracy of all feature combinations is more pronounced for MEAD (from 82% to 87%) than for JPL (from 92% to 93%). This indicates that the additional capabilities (mainly audio) are more complementary when used in conjunction with the other features. In addition, when the sensor and audio features are merged, SimpleMKL outperforms other classifiers because of its superior weighing of multimodal variables.

Figure 6 shows the confusion matrices produced by SVM, MKBoost, and SimpleMKL. When the ego-motion level of activity is minimal (e.g., reading, organizing files, or texting), recognition performance is poor. In contrast, performance improves for tasks involving greater ego-motion (i.e., walking, doing push-ups, or walking downstairs).

Table 6: F1-score of single features and their combinations for MEAD (MKL-based EAR)

		SVM-Linear	MKBoost	SimpleMKL
Single Features				
Video	Trajectory	0.63	0.64	0.65
Video	HOG	0.72	0.72	0.72
Video	HOF	0.82	0.82	0.82
Video	MBH	0.72	0.72	0.73
Sensor	FVS	0.64	0.63	0.62
Audio	Audio	0.44	0.43	0.46
Combination of Modalities				
Sensor + Video		0.83	0.85	0.86
Video + Audio		0.84	0.85	0.86
Sensor + Audio		0.63	0.67	0.69
Combination of All Modalities				
Video + Sensor + Audio		0.84	0.86	0.87

3.3.4 Comparative Results

In this section, the average accuracy (A), precision (P), recall (R), Kappa value (K), and F1-scores (F) are compared with those of state-of-the-art approaches (Table 7). Utilizing all modalities, we obtained the following results: (a) global and local features for JPL and DogC, and (b) video, audio, and sensor features for MEAD.

For JPL, the results are compared to four other works: [7, 8, 30, 81]. GOFF and VIF characteristics were employed with SVM and kNN classifiers in [30]. In [8], a structural learning approach was used with HOF, Log-C, and cuboid features, whereas [7] presented an MKL-based solution with the same features. In [81], a convolutional long-short-term memory (LSTM) was employed to encode features with convolutional neural networks (CNNs) while retaining long-term temporal changes. The results indicate that SimpleMKL and MK-Boost have comparable performance and score the highest. In addition, increasing the feature set for the proposed MKL-based solution enhances the overall accuracy when compared to the findings in [7], which utilizes a subset of the features.

For DogC, we considered the outcomes of [7, 30, 38] in which DogC was also evaluated. The method described in [38] integrated global (dense optical flow and local binary pattern) and local (normalized pixel values, HOG, and HOF) motion descriptors with a modified histogram intersection kernel.

The results of MEAD were compared to those of [36], which utilized identical video and sensor features. In contrast to [36], MKL utilized an audio feature as an additional modality. It should be underlined that adding merely audio without modifying the learning mechanism employed, as in [36] increased recognition accuracy from 83 to 84 percent. In addition to

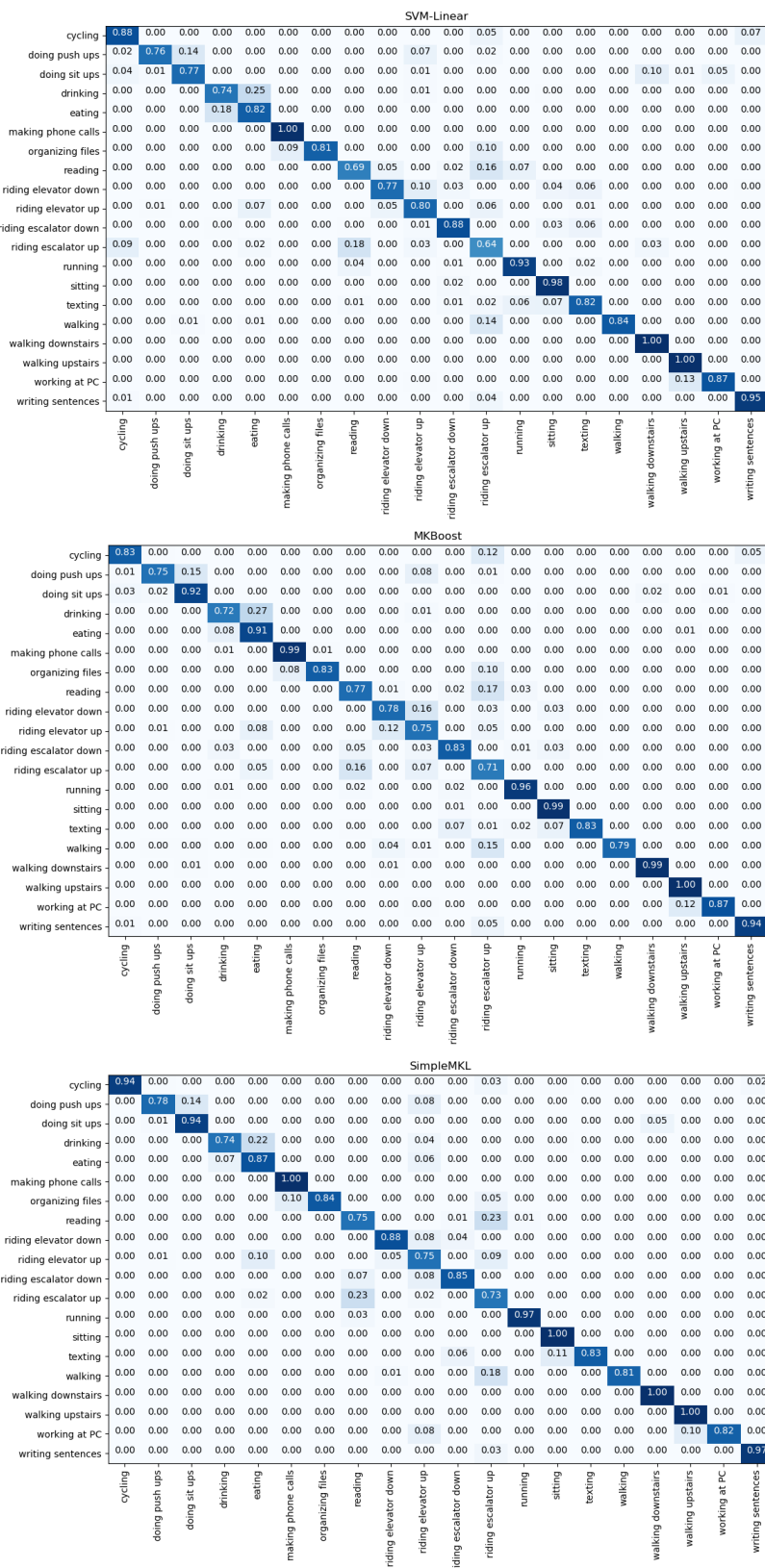


Figure 6: Confusion matrices of linear SVM, MKBoost, and SimpleMKL algorithms for MEAD [1].

increasing recognition accuracy from 84% to 87%, replacing linear SVM with MKL yielded another significant result.

Table 7: Comparative performances of MKL-based EAR

Dataset	Method	A	P	R	κ	F
JPL	Ryoo & Matthies [8]	0.90	-	-	-	-
	Abebe et al. [30]	-	0.87	0.85	-	0.86
	Ozkan et al. [7]	0.87	-	-	-	-
	Sudhakaran & Oswald [81]	0.91	-	-	-	-
	SVM	0.91	0.92	0.91	0.89	0.91
	DC-Int	0.87	0.90	0.87	0.87	0.89
	MKBoost	0.92	0.94	0.92	0.91	0.93
SimpleMKL	0.92	0.93	0.92	0.91	0.93	
DogC	Abebe et al. [30]	-	0.62	0.59	-	0.61
	Iwashita et al. [38]	0.61	-	-	-	-
	Ozkan et al. [7]	0.65	-	-	-	-
	SVM	0.63	0.65	0.63	0.58	0.64
	DC-Int	0.60	0.64	0.60	0.59	0.62
	MKBoost	0.61	0.64	0.63	0.57	0.63
SimpleMKL	0.64	0.66	0.65	0.61	0.65	
MEAD	SVM (Song et al. [36])	0.84	0.86	0.84	0.84	0.85
	MKBoost	0.86	0.85	0.86	0.85	0.86
	SimpleMKL	0.87	0.88	0.87	0.86	0.87

Table 8 demonstrates that a single kernel solution is the fastest for both training and testing, whereas MKBoost is the slowest. Modern SVM solvers approach a scaling rule that shows the computational cost of addressing the SVM issue contains both a quadratic and a cubic component expanding at least as n^2 when C is small and n^3 when C gets large [147]. MKBoost, on the other hand, consists of several kernel solvers whose number is proportional to the number of feature combinations (F), the number of trials (T), and the size of the base kernel pool (K_p). MKBoost’s timing performance is consistent with the theoretical calculation since its training time equals $F * T * K_p * T_{single}$, where T_{single} is the training time required for a single kernel. SimpleMKL, as opposed to MKBoost, employs the previous SVM solution that provides a good prediction for the current SVM training. Thus, the calculation time required for SVM training is drastically reduced. SimpleMKL might be favored as an alternate learning algorithm for sensor data streams based on its rapid convergence and efficient learning performance, although being slower than the single kernel technique.

Table 8: Time complexity analysis of MKL-based EAR

	Training Time per Sample (ms)			Test Time per Sample (ms)		
	Single Kernel	MKBoost	SimpleMKL	Single Kernel	MKBoost	SimpleMKL
Single Channel	6.6	14946.7	79.5	4.2	207.4	14.6
2 Channels	6.8	30320.6	182.7	4.3	254.3	18.9
3 Channels	7.2	71371.8	189.1	4.5	255.3	16.6

3.4 Discussion

Statistical analysis was performed using the base kernel and feature selections for MKBoost to figure out how flexible MKL techniques are. For this aim, each experiment’s base kernels and properties were recorded as their merits and limitations.

3.4.1 Base Kernel Selection

Base kernel selection is one of the most crucial training processes. This part presents the statistical outcomes of 100 test repetitions for base kernel type selection (Figure 7). According to the findings, the linear kernel was the most popular across all the datasets. However, the selection characteristics of the three datasets are distinct. After the linear kernel, the polynomial kernel was the most popular for JPL, whereas RBF, hist-int, and DC-hist were used less frequently. DogC selected hist-int, and poly at a similar rate, whereas DC-Hist was the least preferred kernel. The algorithm primarily selected a linear kernel for the MEAD dataset. Although RBF and polynomial kernels were also chosen, they were far less preferred. This demonstrates that the retrieved features from the MEAD dataset were predominantly linearly separable. A broader selection of base kernels for DogC suggests that most of its characteristics are non-linearly separable samples. JPL dataset features were revealed to be more diverse in terms of their linear separability.

3.4.2 Feature Selection

Feature selection is another crucial step in the training process. Figure 8 depicts the frequency of selection for distinct qualities. At each trial, a feature is believed to be chosen if it is included in selected feature combinations. The JPL and DogC datasets contain GOFF, VIF, Log-C, and cuboid features, whereas the MEAD dataset groups the features for the primary modalities (video, sensor, and audio) for readability. For instance, GOFF is assumed to be selected when any feature combination containing GOFF (for instance, GOFF+VIF+Log-C) is selected. According to the results, the optical flow-based characteristic GOFF is the most discriminative for JPL. GOFF is the second most popular feature of DogC, following cuboid. The most popular MEAD features are video features compared to sensors and audio.

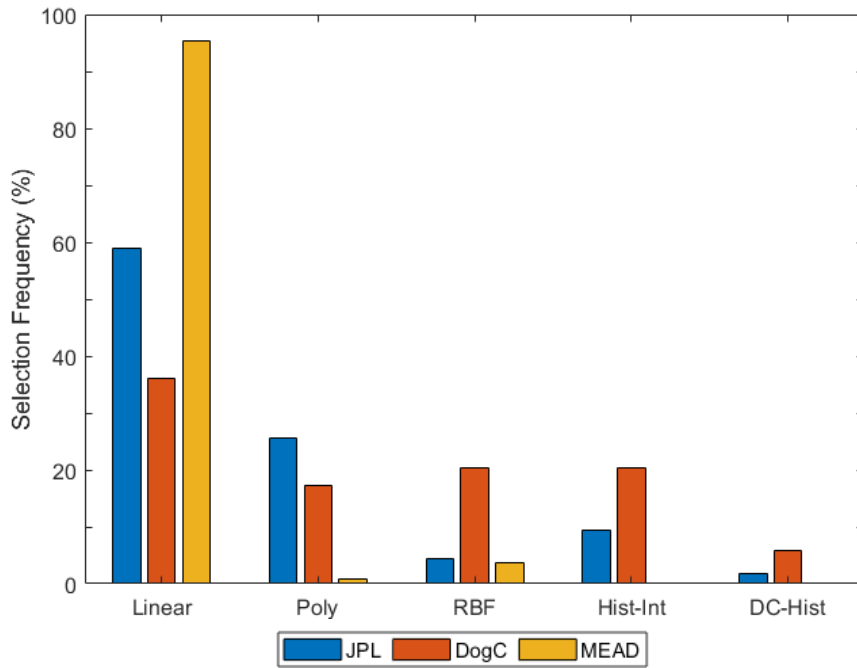


Figure 7: The number of selected base kernels for each dataset [1].

On the other hand, the selection rates of features reveal some information on the peculiarities of egocentric datasets. The characteristics can be arranged in the order of their selection frequency for various datasets as follows: JPL: GOFF > VIF > Log-C > Cuboid, DogC: Cuboid > GOFF > VIF > Log-C; MEAD: Video > Sensor > Audio.

The various ordering of the datasets demonstrates that MKBoost can adapt to input data with distinct features. While global features typically provided the most remarkable results for JPL, local characteristics were more crucial for DogC due to the more chaotic nature of dog movements.

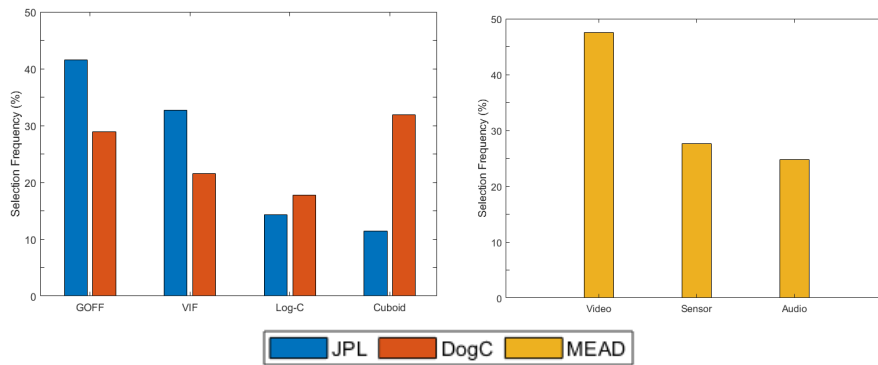


Figure 8: The number of selected features for each dataset [1].

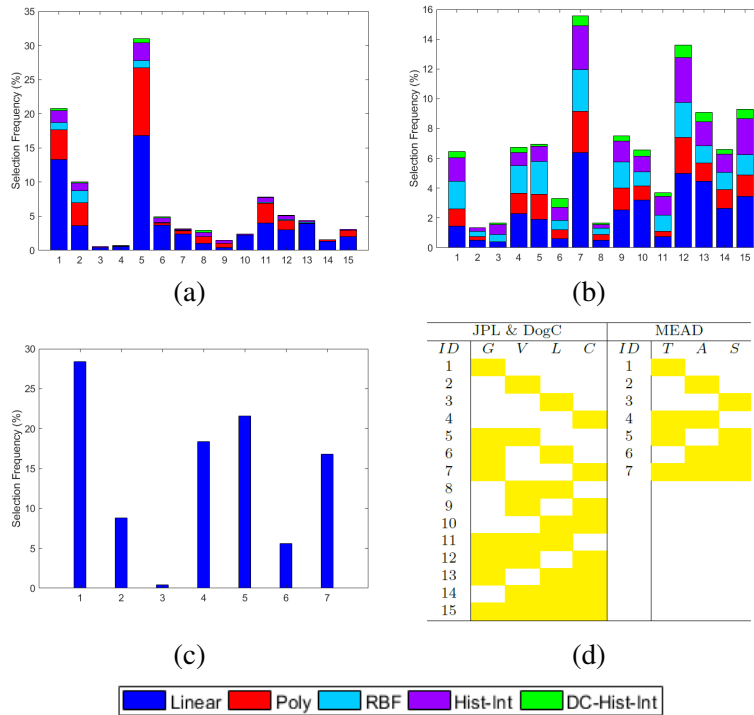


Figure 9: The number of selected feature combinations of JPL (a), DogC (b) and MEAD (c) for the given feature composition color codes (d) [1].

In addition, a more in-depth analysis of kernel and feature selection was conducted to determine which feature combinations are most frequently utilized with specific base kernels. As illustrated in Figure 9, the histograms of the selected base kernels were recovered for each feature combination. Figure 9-(d) displays the feature combination IDs and their respective feature compositions. The feature included in the associated feature combination is highlighted in yellow, and their names are abbreviated as follows: GOFF:*G*, VIF:*V*, Log-C:*L*, Cuboid:*C*, Audio:*A*, Dense Trajectory Features:*T*, Sensor Features:*S*. The feature combination IDs for JPL and DogC are identical. However, they differ for MEAD because it employs a distinct set of properties.

Figure 9-(a) demonstrates that GOFF, VIF, and GOFF+VIF (Feature IDs 1, 2, and 5) were JPL’s most often chosen features. For DogC, cuboid and GOFF (Feature IDs 4 and 1) were predominantly selected as single features, while cuboid was included in the most often selected feature combinations (Figure 9-(b)). Conversely, dense trajectory features were chosen in nearly all feature combinations for MEAD (Figure 9-(c)). Although audio has a high selection rate (Figure 8) for MEAD, it is typically selected in conjunction with other features, indicating that it gives complementary information to the other modalities.

3.4.3 Analysis of the Overall Framework

An MKL-based framework was applied to three types of sensor data (video, audio, and wearable sensors) to identify egocentric activities in this study. The results demonstrate that the suggested method effectively distinguishes egocentric actions and provides a framework that may be expanded with additional sensors. New characteristics from these modalities are essentially new information channels that basic learners must learn adaptively. Each characteristic's weight is determined by its contribution to classification performance. Thus, feature selection and model training are performed simultaneously.

The performance results demonstrate that the suggested framework consistently achieves higher performance than the current best practices. Multiple kernel learning is anticipated to produce generally superior results compared to single kernel learning. In some instances, single-kernel solutions, such as cuboid for DogC and FVS for MEAD, provide the optimum performance. Using the same set for all possible feature combinations does not always result in the ideal solution. Experiments have demonstrated that when the kernel pool for MKL is adequately modified, the performance outcomes are equivalent to those of the single kernel approach. In addition, it was discovered that MKBoost was inferior to SimpleMKL for DogC and MEAD. Boosting requires a substantial number of labeled samples to achieve acceptable classification performance [148]; however, the number of samples employed in this work is constrained by the number of videos in the datasets. As DogC and MEAD contain significantly fewer videos, the MKBoost performance for these datasets remains inferior. MKL approaches (particularly SimpleMKL) outperform the others when combining features, making it a prominent method for feature fusion.

Notably, the use of several modalities (i.e., video and audio) increases the effectiveness of activity recognition, suggesting that the multi-modal features contain complementary information from various domains. Furthermore, it is noted that MKL is a more effective solution for multi-modal features than other state-of-the-art techniques. For example, combining sensor characteristics with audio for MEAD significantly improves MKL performance relative to single-kernel learning.

Unlike single kernel learning and MKBoost, the temporal complexity of SimpleMKL, which executes an optimization procedure to calculate the kernel weights, is hard to quantify. Due to the fact that the procedure of feature extraction is the same for all learning strategies, the time complexity for training and testing was analyzed among classifiers. In Table 8, the average processing time for MEAD video clips is shown. Tests were conducted with one channel (video), two channels (video+sensor), and three channels (video+sensor+audio) in order to determine the variance in execution times with regard to the number of channels. Intel® Core™ i5-6200U @ 2.30GHz with 8GB RAM was the PC setup for the experiments.

In addition, the suggested framework yielded encouraging results on three distinct egocentric datasets containing between 7 and 20 actions, demonstrating its scalability. Even while MKL techniques achieved satisfactory learning results for the EAR problem, they need the configuration of a predefined kernel pool. MKL methods may not be able to converge to the optimal solution if the basis kernels are not chosen appropriately.

In this thesis, we offered a new framework for EAR based on multimodal features and multi-kernel learning classification. Experiments have demonstrated that mixing many modalities enhances recognition performance. On three distinct egocentric datasets, the suggested solution demonstrated superior performance compared to state-of-the-art methods. This study demonstrates that employing MKL with multimodal characteristics is an effective strategy for EAR.

Alternatively, the variation of the selected feature and kernel sets for MKL is closely related to the properties of egocentric datasets. Because the recording settings of videos (i.e., location, time, actors), accessible sensor information (i.e., visual, audio, sensor), and the dynamics of egocentric activities vary among datasets, it is impossible to establish a universal set of optimal features and kernels. To give a generic solution to recognize egocentric actions, it is necessary to propose an adaptable approach that dynamically learns the changing conditions of datasets such as the one presented in this study.

Combining visual data with audio or wearable sensors requires additional EAR research. In contrast to third-person activity recognition, egocentric activity datasets may contain more information about activities thanks to a range of sensors that capture the event directly. To recognize users' activities, it is required to design new frameworks that can mix features from diverse domains effectively and practically.

The suggested framework adaptively combines several modalities, although it depends on handcrafted characteristics. On the other hand, a growing number of research propose end-to-end solutions utilizing deep learning algorithms employing visual [2, 65, 88] and wearable sensor data [149, 150]. Developing a generic end-to-end solution that automatically learns the characteristics intrinsic to multiple modalities is an open research question.

CHAPTER 4

EGOCENTRIC ACTIVITY RECOGNITION USING TWO-STAGE DECISION FUSION

The primary objective in this part is to develop a two-stage decision fusion (TSDF) mechanism that adaptively weighs input modalities based on their relevance to activities. Using such a mechanism may contribute to the fusion process of the proposed solutions for different modalities. For example, the system may focus more on appearance for the activities having weak motion patterns (in other words having low magnitudes optical flow vectors), such as *rock climbing*, *paragliding*, or *scuba diving*. On the other hand, activities including strong motion patterns (such as *running* or *cycling*) may pay more attention to motion information. In addition, audio information may contain distinguishing characteristics, such as the sound of engines while driving cars or riding motorcycles. As depicted in Figure 10, one study [2] demonstrated that applying different weights during the decision fusion phase of appearance and motion streams improves action recognition performance. However, the weights were manually adjusted to examine their effects of them on the recognition performance. In contrast, in this work, we aimed to build adaptive weighing models that can improve the overall accuracy by generating weights for the segment and video-level decisions instead of manually providing weights. For that purpose, we design two distinct weighing models for segment and video levels named stream and segment fusion models.

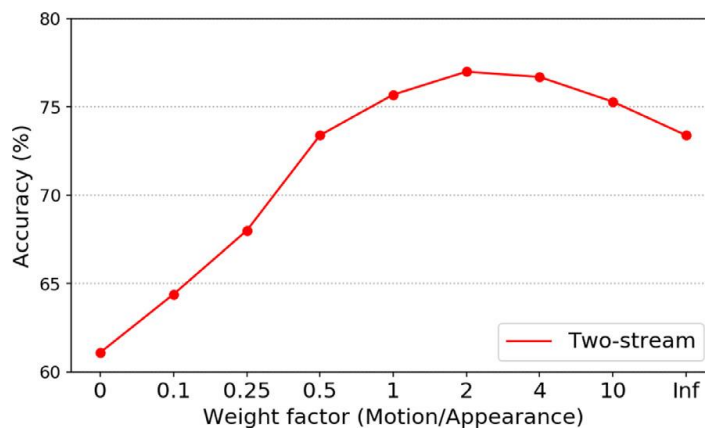


Figure 10: Recognition accuracy vs stream weight factor [2].

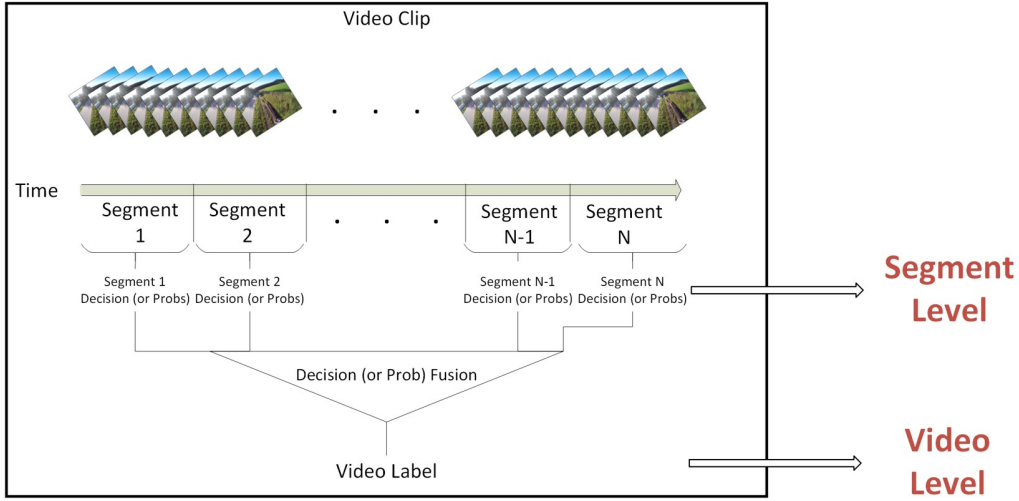


Figure 11: Stream and segment decisions for video clips.

Before discussing the designs of stream and segment fusion models in detail, it is better to make the definition of the segment and video-level decisions. Each video clip in our problem includes a single activity and consists of hundreds or thousands of frames. Since it is not feasible to process every frame in video clips, the frames are sampled throughout the videos. A video clip is divided uniformly in time into a predetermined number of parts for this purpose. Here, the classification of activities is completed for each segment, referred to as segment-level decisions. The segment decisions are then fused (mixed) to provide the final choice for the corresponding video clip. The final decision is known as the video-level decision (Figure 11). Therefore, the stream fusion model generates weights for the segment-level decisions. On the other hand, segment fusion model outputs are employed to weigh the fused segment decisions.

4.1 The Proposed Framework

The proposed solution consists of three distinct phases. The first phase is training single-stream, EAR models. In the second phase, the confidences of the pre-trained single-stream EAR models are employed to produce training weights for the stream and segment fusion models. The last phase involves the fusion of single-stream EAR model decisions for the test set at the segment and video levels according to the outputs of fusion models.

Stream and segment fusion problems have been considered as separate regression tasks. For stream fusion, two approaches were employed: one uses deep features combined with various types of regressors, and the other utilizes a deep regression architecture (Variational Auto-Encoders, or VAEs). On the other hand, a class-aware regression-based approach was proposed for segment fusion using the class predictions coming from segment-level decisions.

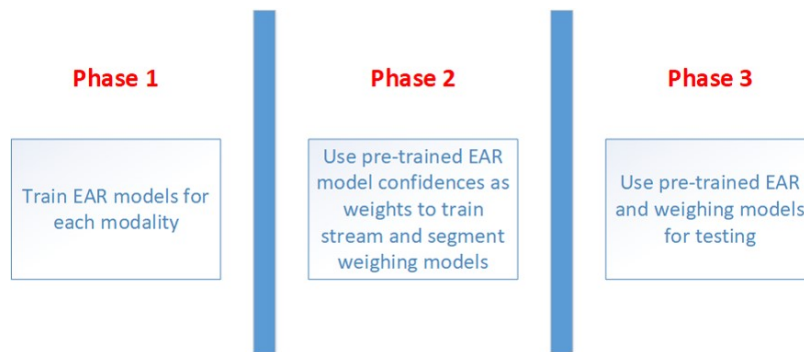


Figure 12: The phases of the proposed weighing solution.

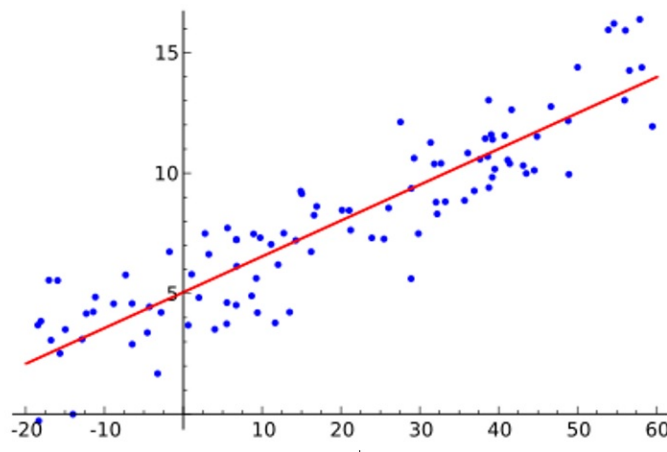


Figure 13: Simple linear regression with one independent variable.

Before proceeding, it is essential to clarify some basic concepts regarding regression. Simple regression refers to the scenario of a single scalar predictor variable x and a single scalar response variable y (e.g., simple linear regression in Figure 13).

Many regression, often known as multivariable regression, refers to the extension to multiple and vector-valued predictor variables. Multivariate regression is sometimes mistaken for multiple, or multivariable regression [151]. Multivariate regression (also known as multiple-output regression or multi-output regression) measures the degree to which numerous independent variables (predictors) and multiple dependent variables (responses) are associated. In multivariate regression, the outputs often depend on both the input and the other. This indicates that the outputs are frequently interdependent and may necessitate a model that predicts both outputs jointly or each output based on the other outputs.

In our case, the regressor model will generate weights for each stream depending on the input and each other that corresponds to multivariate regression. Multivariable or multiple regressors predict a single result at each time for segment-weighting regressors (Table 9).

Table 9: Regression types for stream and segment weighing

	Predictors	Response Variable	Regression Type
Stream Weights	Multiple-independent variables	Multiple-dependent variables	Multi-variate Regressor
Segment Weights	Multiple-independent variables	Single variable	Multivariable Regressor

To design a TSDF-based EAR system, we chose backbone models from ConvNet architectures. In literature, two-stream networks have recently become state-of-the-art activity recognition methods for both EAR [116] and HAR [152]. The two-stream ConvNets generally receive appearance and motion streams as inputs. While the appearance stream generates visual features from RGB images, the motion stream generates motion characteristics from stacked optical flows. In this study, we utilized an audio stream to diversify the modalities and also to test whether the cross-modalities contribute to the results similar to the MKL-based EAR framework.

The following section presents two distinct designs (deep feature-based, and VAEs-based) of stream fusion. Later, the class-aware segment fusion strategy will be discussed in detail.

4.1.1 Stream Decision Fusion

As mentioned before, we designed the weight estimation as a regression problem. For that purpose, a supervised learning procedure was defined to train regression models. Therefore, target weights for modalities need to be specified for training. However, it is hard to determine the importance of modalities for activities. We believed that the confidence scores of pre-trained single-stream EAR models can be utilized as modality weights. To do this, confidence scores of each single-stream EAR model are collected corresponding to the actual classes and normalized to determine target weights as shown below:

$$W_s^{stream} = [w_s^1 \ w_s^2 \ \dots \ w_s^M] \quad (8)$$

where W_s^{stream} represents normalized stream weight vector for the selected segment, M is the number of modalities, and w_s corresponds to the scalar weight of the selected segment. On the other hand, the scalar stream weight for the selected modality is computed as below:

$$w_s^{m_i} = \frac{c_t^{m_i}}{\sum_{j=1}^M c_t^{m_j}} \quad (9)$$

where $w_s^{m_i}$ corresponds to the scalar weight for the selected segment and modality index, and $c_t^{m_j}$ is the confidence value of the target class for the selected modality index.

Table 10 shows an example of a four-class scenario.

Table 10: Weight generation of streams for training (4-class problem)

Target Class	Stream Type	C1	C2	C3	C4	Stream Weights		
						RGB	Flow	Audio
C1	RGB	0.80	0.10	0.10	0.00			
	Flow	0.90	0.05	0.00	0.05	0.38	0.43	0.19
	Audio	0.40	0.30	0.20	0.10			
C4	RGB	0.20	0.50	0.20	0.10			
	Flow	0.30	0.10	0.10	0.50	0.12	0.59	0.29
	Audio	0.20	0.25	0.30	0.25			
C2	RGB	0.70	0.10	0.20	0.00			
	Flow	0.90	0.00	0.10	0.00	0.33	0.00	0.67
	Audio	0.20	0.20	0.55	0.05			

The following sections explain two distinct designs for model-based stream fusion, one of which uses deep features with various regressors while the other utilizes VAEs for the weight estimation problem.

4.1.1.1 Deep Feature-based Stream Fusion

The first approach uses some of the well-known regressors with deep features and stream weights generated by pre-trained EAR models. Some of these regressors explicitly enable multiple outputs (e.g., linear regression, k-neighbors, decision trees, random forest). Others (i.e., SVR) require some workarounds to predict multi-output variables. On the other hand, the collected features are initially fused which are then given to regression models as inputs. Figure 14 depicts the proposed flowchart of stream weighing with deep features.

Linear, k-nearest neighbor, decision tree, random forest, support vector (direct and chained multi-output), Lasso and Ridge, XGBoost, and NGBoost regressors were employed as multi-variate regressors and explained in the following paragraphs.

Linear Regression:

Multiple linear regression is an expansion of simple linear regression to the case of multiple independent variables and a particular case of general linear models with one dependent variable only. The basic model for multiple linear regression is:

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \dots + \beta_p X_{ip} + \epsilon_i \tag{10}$$

for each observation $i = 1, \dots, n$.

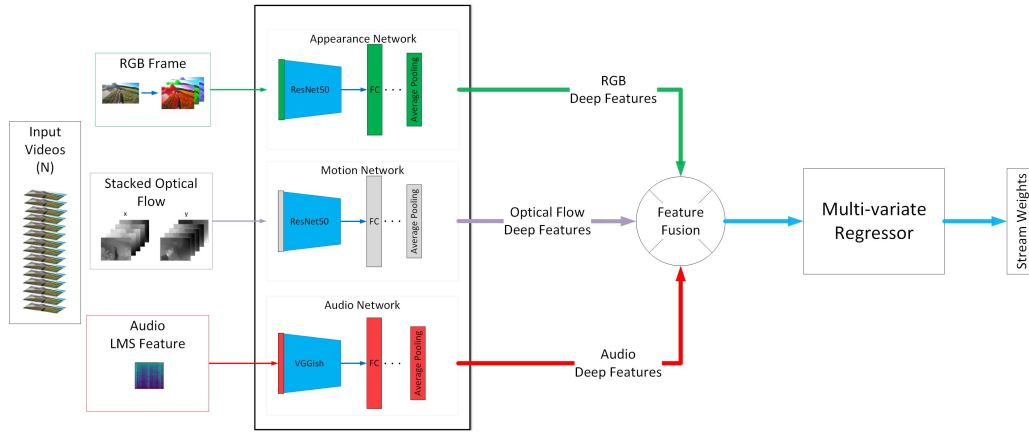


Figure 14: Deep feature-based stream weighing.

In the formula above we consider n observations of one dependent variable and p independent variables. Thus, Y_i is the i^{th} observation of the dependent variable, X_{ij} is i^{th} observation of the j^{th} independent variable, $j = 1, \dots, p$. The values β_j represent parameters to be estimated, and ϵ_i is the i^{th} independent identically distributed normal error.

K-Nearest Neighbor (KNN) Regressor:

Regression is performed based on k-nearest neighbors. The target is predicted by local interpolation of the targets associated with the nearest neighbors in the training set. In our work, the number of neighbors was selected as 10.

Decision Tree Regressor:

Decision Trees are one of the non-parametric methods of supervised learning used for classification and regression. The objective is to develop a model capable of predicting the value of a target variable using simple decision rules learned from the data features. A tree can be viewed as a piecewise constant approximation.

Random Forest Regressor:

Random Forest is an ensemble technique that can handle regression and classification tasks using many decision trees and the Bootstrap and Aggregation technique, also known as bagging. Rather than depending on individual decision trees to determine the final result, this method combines many decision trees.

Support Vector Regressor:

A support vector machine algorithm aims to locate a hyperplane in an n-dimensional space that classifies the data points in a distinct way. Support Vectors refer to the data points on either side of the hyperplane that is closest to the hyperplane. These variables affect the position and direction of the hyperplane and contribute to the formation of the SVM. Hyperplanes are decision boundaries used to predict the output of a continuous function. Support Vectors refer to the data points on either side of the hyperplane that is closest to the hyperplane. Support

Vector Regression is a technique for supervised learning that predicts discrete values. Support Vector Regression employs the same underlying concept as SVMs. The objective of SVR is to identify the optimal fit line. The best-fitting line in SVR is the hyperplane with the most significant number of points.

Unlike other Regression models, the SVR seeks to fit the best line within a threshold value, as opposed to minimizing the difference between the actual and predicted values. The threshold value is the distance between the hyperplane and the boundary line.

SVR is a regression technique with a single output. Therefore, the output dimension must be expanded relative to the number of input modalities. Direct and chain multi-output are the two primary methods to convert SVR to multi-output configuration, described in the following sections.

Direct Multi-output:

By dividing the multi-output regression problem into multiple sub-problems, it is possible to use regression models designed to predict a single value (i.e., SVR) for multi-output regression. The most apparent approach is splitting a multi-output regression problem into multiple single-output regression problems.

If, for instance, a multi-output regression problem required the prediction of three values y_1 , y_2 , and y_3 given an input X , this may be partitioned into three single-output regression problems:

- **Problem 1:** Given X , predict y_1
- **Problem 2:** Given X , predict y_2
- **Problem 3:** Given X , predict y_3

Developing a multi-output regressor may involve developing a separate regression model for predicting each output value. Direct multi-output can be thought of as a direct approach because each target value is modeled directly. The direct multi-output regression approach divides the problem into individual problems for each target variable to be predicted. This presumes that the outputs are independent, which may be different. Nonetheless, this method can yield reasonably accurate predictions on various scenarios and may be worth a try if only as a baseline for performance.

Our research used linear, RBF, and polynomial kernels to evaluate SVR regression models with a direct multi-output approach and linear, RBF, and polynomial kernels.

Chained Multi-output:

The second method for multi-output regression is a chained version of the direct multi-output method. The prediction from the first model is used as input for the second model, and the process of output-to-input dependency is repeated throughout the chain of models. The first model in the sequence uses the input and predicts one outcome. The second model uses the data and output from the previous model to generate a prediction.

For example, if a multi-output regression problem required the prediction of three values y_1 , y_2 , and y_3 given an input X , then this could be partitioned into three dependent single-output regression problems as follows:

- **Problem 1:** Given X , predict y_1
- **Problem 2:** Given X and \hat{y}_1 , predict y_2
- **Problem 3:** Given X , \hat{y}_1 , and \hat{y}_2 , predict y_3

Similar to the direct multi-output approach, SVR was used with linear, RBF, and polynomial kernels.

LASSO and Ridge Regression:

LASSO stands for the acronym of Least Absolute Shrinkage and Selection Operator. Regularization techniques like LASSO and Ridge regressions use variable selection and regularization to make the predictions of a statistical model more accurate and easy to understand. The method favors basic, sparse models—models with fewer parameters. This sort of regression is ideal for models with a high degree of multicollinearity or to automate certain aspects of model selection, such as variable selection and parameter removal.

LASSO Regression employs the $L1$ regularization approach. It is utilized when there are numerous features since it performs feature selection automatically. $L1$ regularization adds a penalty proportional to the absolute value of the coefficient's magnitude. This type of regularization may produce sparse models with few coefficients. Some coefficients can become zero and be eliminated from the model.

$$L_{Lasso}(\hat{\beta}) = \sum_{i=1}^n (y_i - x_i \hat{\beta})^2 + \lambda \sum_{j=1}^m |\beta_j| \quad (11)$$

where y_i is the outcome, $x_i := (x_1, x_2, \dots, x_p)$ is the covariate vector for the i^{th} case, and $\beta := (\beta_1, \beta_2, \dots, \beta_p)$ is the coefficient vector.

If the regularization is performed using the $L2$ regularization technique, it's called Ridge Regression.

$$L_{Ridge}(\hat{\beta}) = \sum_{i=1}^n (y_i - x_i \hat{\beta})^2 + \lambda \sum_{j=1}^m (\hat{\beta}_j)^2 \quad (12)$$

Setting λ to 0 is the same as using the Ordinary Least Square (OLS), while the larger its value, the stronger the coefficients' size penalized.

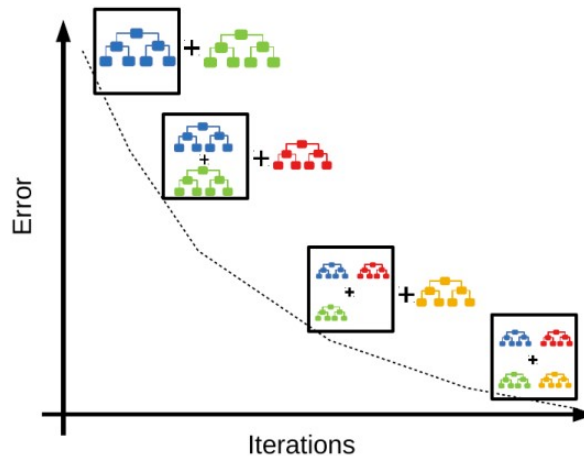


Figure 15: Gradient boosting algorithm.

XGBoost:

XGBoost stands for eXtreme Gradient Boosting [153]. The gradient-boosting decision tree algorithm is implemented by the XGBoost package. This approach is also known as gradient boosting, multiple additive regression trees, stochastic gradient boosting, and gradient boosting machines. Boosting is an ensemble strategy in which new models are introduced to existing models to correct their errors. Models are successively introduced until no further improvements are possible. A well-known example is the AdaBoost algorithm, which weighs difficult-to-predict data points.

Gradient boosting is a technique in which new models are constructed to predict the residuals or errors of previous models, which are then merged to form the final prediction (Figure 15). Gradient boosting is named so because it employs a gradient descent approach to minimize loss when adding new models. This method applies to both regression and classification predictive modeling issues.

XGBoost is a more regularized variation of Gradient Boosting with the following advantages:

- Utilizes advanced regularization ($L1$ & $L2$), which enhances the generalization capabilities of the model.
- Provides superior performance to gradient boosting.
- Training may be parallelized/distributed across clusters and is extremely quick.
- Computes second-order gradients, i.e., second partial derivatives of the loss function, which provide additional information regarding the gradient's direction and how to reach the loss function's minimum.
- Manages values absent from the dataset. XGBoost can handle missing values internally when missing values are not treated differently during data manipulation.

NGBoost:

Natural Gradient Boosting (NGBoost) is a technique for probabilistic prediction using gradient boosting [3]. Standard regression models yield a point estimate conditional on covariates, whereas probabilistic regression models return a complete probability distribution over the resulting space conditional on covariates. This enables the estimation of prediction uncertainty. Utilizing the natural gradient to accomplish gradient boosting by recasting it as a problem of estimating the parameters of a probability distribution is the fundamental idea here. NGBoost is configurable concerning the selection of the base learner, distribution, and scoring rule (Figure 16).

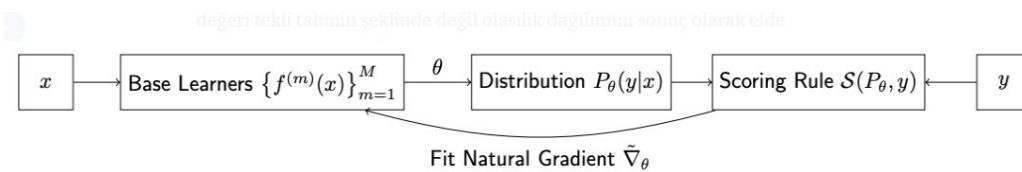


Figure 16: NGBoost algorithm [3].

Here, we need to add some notes about base learners, probability distribution, and scoring rules:

- **Base Learner:** The most common choice is Decision Trees, which tend to work well on structured inputs.
- **Probability Distribution:** The distribution needs to be compatible with the output type—Normal distribution for real-valued outputs, Bernoulli for binary outputs.
- **Scoring rules:** Maximum Likelihood Estimation is an obvious choice. More robust rules such as Continuous Ranked Probability Scores are also suitable.

Ordinary gradients are often inappropriate for learning multi-parameter probability distributions (such as the Normal distribution). As demonstrated in the preceding illustration of probabilistic regression, the training dynamics with natural gradients tend to be significantly more stable and produce a better match (Figure 17).



Figure 17: Ordinary gradient descent and NGBoost regression sample outputs [3].

4.1.1.2 VAEs-based Stream Fusion

We employed Variational Auto-Encoders (VAEs) regressor as the second approach for stream fusion. Convolutional-based deep architectures have been preferred for traditional computer vision tasks, such as image classification [154, 155] and object detection [156, 157] over the past decade. Simply, it consists of numerous convolutional layers, followed by a few fully-connected layers and a classification softmax layer with, for example, a cross-entropy loss (ConvNets). In addition to classification, ConvNets have been used to address also regression problems [158] with the increasing popularity of deep regression.

On the other hand, deep auto-encoders (AEs) have been explicitly utilized for unsupervised learning issues. They build a representation in latent space (z) for the provided inputs (x) by passing through an encoding function (e) and reconstruct them with a decoding function (d) from their latent representation (\hat{x}) (Figure 18). In the simple case, only the reconstruction loss is computed to measure between the input data x and the encoded-decoded data $d(e(x))$.

AEs may also generate content by decoding randomly sampled points from the latent space. At that moment, the regularity of the latent space determines the quality and relevance of generated data. The irregularity of data in the latent is one of the main problems for AEs. Variational Auto-Encoders (VAEs) have been suggested to address this anomaly by regularizing training to prevent overfitting and guarantee that the latent space has desirable properties that permit the generative process. During the encoding-decoding process, VAEs incorporate some regularization of the latent space rather than recording input as a single point. It is encoded as a distribution (p) over the latent space. Figure 19 illustrates the distinction between AE and VAE dataflow.

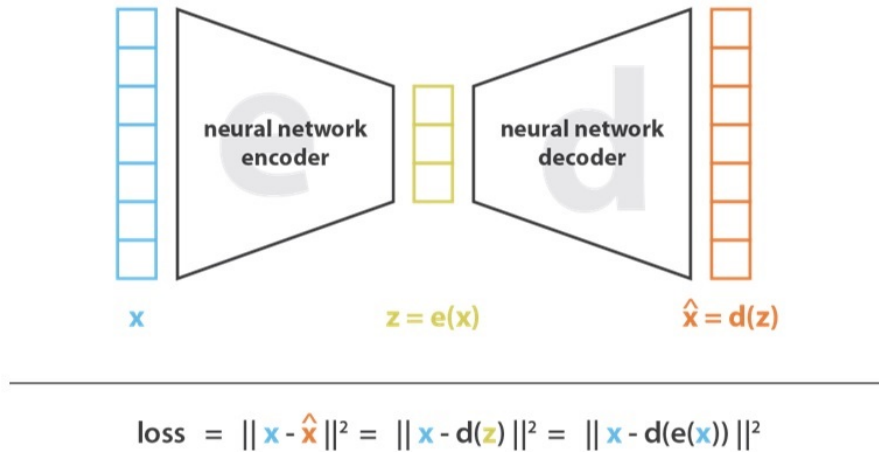


Figure 18: Illustration of an auto-encoders [4].

In practice, the encoded distributions are normally distributed ($x \sim \mathcal{N}(0, I)$), so the encoder may be taught to deliver the mean (μ_x) and covariance matrix (σ_x) that describes these Gaussians. Input is encoded as a distribution with some variance rather than a single point so that the latent space regularization may be expressed naturally: the distributions generated by the encoder must be close to a standard normal distribution. Consequently, the loss function that is minimized when training VAEs is composed of a "reconstruction term" (on the final layer),

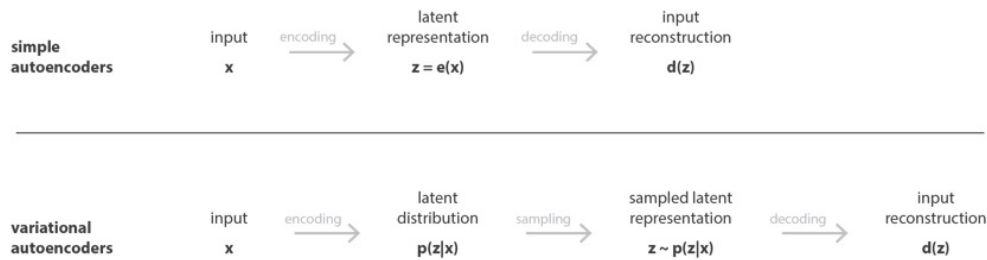
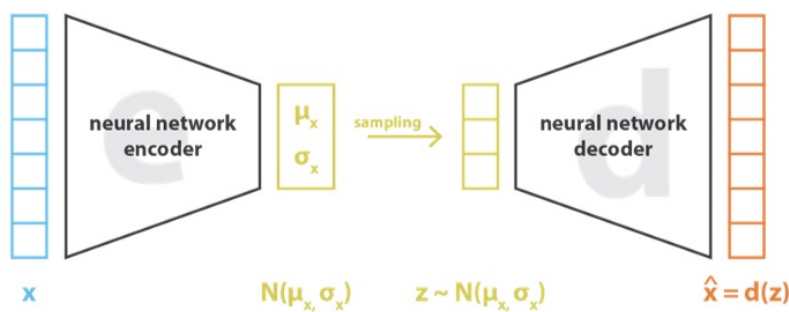


Figure 19: Autoencoder and variational autoencoder differences [4].



$$\text{loss} = \|x - \hat{x}\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)] = \|x - d(z)\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$$

Figure 20: Using auto-encoders as the content generator [4].

which tends to make the encoding-decoding scheme as efficient as possible, and a "regularization term" (on the latent layer), which tends to regularize the organization of the latent space by bringing the distributions returned by the encoder closer to a standard normal distribution. This term is the Kulback-Leibler (KL) divergence between the returned and Gaussian distributions. The Kullback-Leibler divergence between two Gaussian distributions has a closed form that may be written simply in terms of their means and covariance matrices. Figure 20 depicts the structure of VAEs and their loss function, which consists of a reconstruction term (which makes the encoding-decoding scheme efficient) and a regularization term (which makes the latent space regular).

VAEs are frequently employed to discover the complex distributions underlying imaging data [159]. VAEs have been effectively used for numerous neuroimaging challenges, including de-noising [159], anomaly detection [160], and clustering tasks [161], as an unsupervised learning framework. Some publications have used VAEs in the context of supervised regression; that is, regression seeks to predict a scalar outcome from a picture based on a specific training set. In neuroimage analysis, for instance, a scalar (outcome) prediction might be a binary variable indicating whether a subject belongs to the control group or a disease group or a continuous variable encoding a subject's age. The following section describes the application of VAEs to regression concerns.

Variational Auto-encoders for Regression:

Several attempts have been made to integrate regression models into the VAEs framework by doing regression analysis directly on the encoder's learned latent representations [162]. However, these studies continue to decouple the regression model from the auto-encoder such that the regression must be trained using a distinct objective function. Recent breakthroughs in learning disentangled latent representations [163, 164] were utilized to bridge the gap between the two models [161]. In the latent space, a representation is considered disentangled if changes along one dimension of that space are explained by a specific element of variation (such as age or modality weights in our case) while being relatively invariant to other factors (such as sex or race) [163]. In [161], a similar concept has been adopted to construct a unified model that combines regression and auto-encoding. The proposed architecture was evaluated for its ability to forecast a subject's age based only on its structural MR image which is dependent on age. The inference model constructed a probabilistic encoder for determining the latent representation and a probabilistic regressor for predicting age which is different from the traditional VAE modeling of latent representations.

The Proposed Solution with VAE Regressor:

Similar to the technique in [161], the deep features from various modalities can be represented in a disentangled manner in latent space using the stream weights as ground truth during the training phase. Figure 21 depicts the training of the proposed VAE regressor for stream weight predictions. Different from [161], the fused deep features were utilized as inputs to the VAE regressor instead of MRI or ROI images. The VAE regressor is expected to encode the given features in the latent space using the specified stream weights.

The features following fusion are supplied to the VAE regressor. The VAE regressor encodes the specified feature in the latent space using the specified stream weights. There are two main differences between the regressor in [161] and ours according to input types and output dimensions. In [161], the input is an MRI picture, however, the VAE-based stream fusion model receives the fused deep features. Additionally, the regressor in [161] produces a scalar output corresponding to age, but the proposed fusion model has multi-dimensional vector output for stream weights.

The proposed VAE regressor has three error sources during the training phase:

- **Reconstruction error (or expected negative log-likelihood of the datapoint):** The expectation is taken concerning the encoder's distribution over the representations by taking a few samples. This term encourages the decoder to learn to reconstruct the data using samples from the latent distribution. A significant error indicates that the decoder is unable to reconstruct the data.
- **KL Divergence:** The divergence between the encoder's distribution and the prior distribution. This divergence measures how much information is lost and encourages its values to be Gaussian.
- **Prediction error:** The loss between the predicted weights and the ground truth.

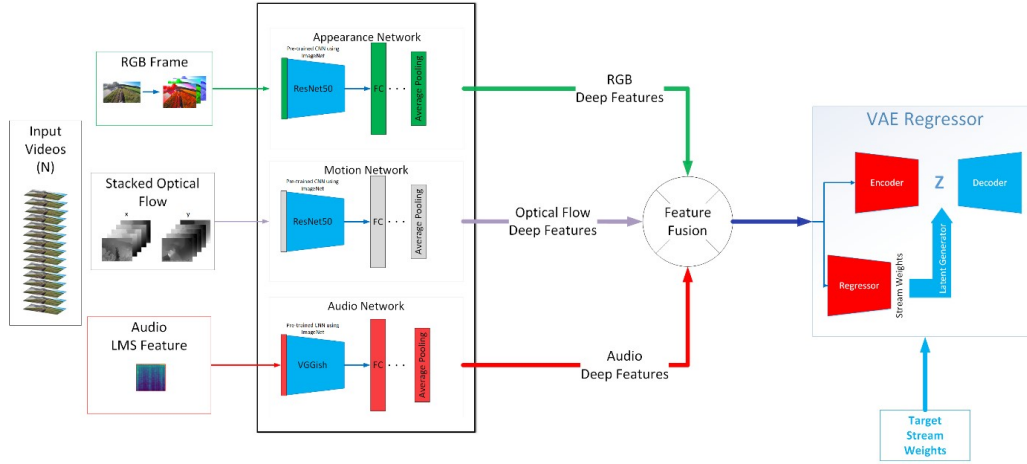


Figure 21: The training of the proposed VAE regressor.

Here, we applied several types of prediction errors listen in (Table 11) including the formulation and descriptions.

4.1.2 Class-aware Segment Fusion

Segment fusion is accomplished at the video level by weighing the segment decisions that have been fused. In a typical case, video-level decisions correspond to selecting the activity with the highest votes, and segment-level decisions are treated equally. Nonetheless, segment-level activity data may differ within video segments. As a result, we establish an additional weight parameter that determines the contribution of segment information to the associated activity. The purpose of segment fusion is to enhance the accuracy of video-level decisions. For that purpose, we utilized the confidence scores of actual classes generated by single-stream EAR models similar to the definition of stream weighing. However, this time the sum of confidence scores throughout all streams is considered as segment weights. The formulation of computing segment weights is given below:

$$W^{segment} = [w^1 w^2 \dots w^K] \quad (14)$$

where $W^{segment}$ represents segment weight vector for a video clip, K is the number of segments, and w corresponds to the scalar weight of the selected segment. On the other hand, the scalar segment weight is computed as below:

$$w^k = \sum_{m=1}^M c_{kt}^m \quad (15)$$

where w^k corresponds to the scalar weight of k^{th} segment, M is the number modalities, and c_{kt} is the confidence score of target class (t) for k^{th} segment.

Table 11: Prediction error formulations used for VAE

Prediction Error	Description
Mean Squared Error (MSE)	$\frac{1}{n} \sum_{t=1}^n e_t^2$
Mean Absolute Error (MAE)	$\frac{1}{n} \sum_{t=1}^n e_t $
Smooth L1 Loss	<p>Creates a criterion that uses a squared term if the absolute element-wise error falls below beta and an L1 term otherwise. It is less sensitive to outliers than MSE loss and in some cases prevents exploding gradients.</p> $\begin{cases} 0.5 * \frac{e_t^2}{\beta}, & \text{if } e_t < \beta \\ e_t - 0.5 * \beta, & \text{otherwise} \end{cases} \quad (13)$
Adaptive Robust Loss	<p>By introducing robustness as a continuous parameter, the loss function enables the generalization of algorithms based on robust loss minimization, enhancing the performance of fundamental vision tasks. The general probability distribution that results from interpreting the loss as the negative log of a univariate density includes the normal and Cauchy distributions as specific examples. This probabilistic interpretation allows the development of neural networks in which the robustness of the loss modifies itself automatically during training, enhancing the performance of learning-based tasks [165].</p>

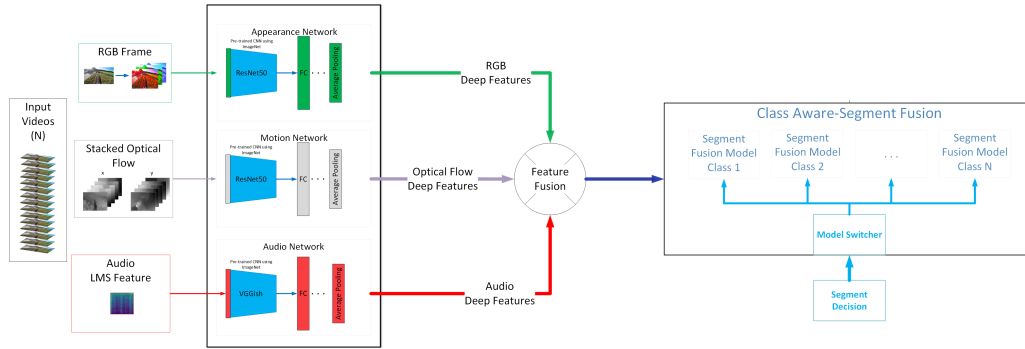


Figure 22: Class-aware segment fusion

Assuming there is a consensus between EAR models for the actual class (i.e., all single-stream EAR models generate the highest confidence score (1.0) for the actual class) regarding a particular segment (during training); then the segment will have the maximum weight that corresponds to the number of modalities (in our case 3.0). Otherwise, segment weights will be lower, especially in the case of miss-classification scenarios. Table 12 displays some examples of segment weight computation for training.

Table 12: Weight generation of segments for training (4-class problem)

Target Class	Stream Type	C1	C2	C3	C4	Segment Weight
C1	RGB	0.80	0.10	0.10	0.00	2.10
	Flow	0.90	0.05	0.00	0.05	
	Audio	0.40	0.30	0.20	0.10	
C4	RGB	0.20	0.50	0.20	0.10	0.85
	Flow	0.30	0.10	0.10	0.50	
	Audio	0.20	0.25	0.30	0.25	
C2	RGB	0.70	0.10	0.20	0.00	0.30
	Flow	0.90	0.00	0.10	0.00	
	Audio	0.20	0.20	0.55	0.05	

Figure 22 also depicts the solution proposed for segment weighing. Similar to stream fusion, deep features should be fused before training. Additionally, a set of regression models are trained for each class. A model switcher is employed to switch between weighing models based on the segment decision generated in the first phase (segment-level decision). The details of constructing a training set for segment weighing will be explained in Section 4.2.7.

4.2 Experiments

In this section, the experimental setup is described in detail, including the dataset used for the evaluation, frame extraction to provide input to the selected deep learning models, dataset splitting, the configuration for single-stream EAR models, and the stream and segment fusion settings.

4.2.1 Egocentric Outdoor Activity Dataset (EOAD)

Before implementing the proposed approach, we analyzed the publicly available egocentric activity datasets regarding their activity definitions, the number of samples/activities, and modality types (Table 13). A collection of datasets describes activities as composed of actions and interacting objects, typically presented as verb-noun pairs. Object detection/segmentation, hand segmentation, and hand pose estimation are crucial tasks for these datasets that examine human-object interactions. The scope of this study does not, however, include human-object interactions. Consequently, the selected datasets contain just verbs and no nouns.

On the other hand, previous tests demonstrated that incorporating audio information improves activity recognition performance. Therefore, we also decided to choose datasets with natural audio in addition to visual information.

After eliminating datasets based on the definition of activities and the selected modality types, we realized that only two of them (HUJI [65, 87] and FPVSum [166]) met the criteria. However, the number of samples and diversity of activities for HUJI and FPVSum were insufficient. Thus, we merged these datasets and also populated them with new YouTube videos.

The selection of YouTube videos was performed according to several criteria. Firstly, videos with overlaid text were not accepted. Secondly, we especially selected unstabilized video footage to get raw ego-motion characteristics that give clues about egocentric activities. Moreover, the videos should have natural sound (having no audio montage). Additionally, the videos should be continuous without a scene cut or jump in time. Video samples were trimmed depending on scene changes for long videos (such as *driving*, *scuba diving*, and *cycling*). The final dataset consists of video clips, including single actions for each video clip and having natural audio information.

The following is a list of EOAD activities: *american football, basketball, biking, boxing, bungee jumping, driving, go-kart, horse riding, ice hockey, jet ski, kayaking, kitesurfing, longboarding, motorcycle, paintball, paragliding, rafting, rock climbing, rowing, running, sailing, scuba diving, skateboarding, skiing, soccer, stair climbing, surfing, tennis, volleyball, walking*.

Table 13: Summary of the most relevant egocentric action recognition datasets

Dataset	Year	#clips	#activities	Sample activities	Modality
UEC [167]	2011	890	28	upright, jog, updown, downright, downforward, slowjog, run, twist, walk, pullups, down-rightleft, crounet, downleft	Video
ADL [168]	2012	436	32	combing hair, make up, brushing teeth, washing hands/face, making tea, vacuuming, watching tv	Video
GTEA Gaze [169]	2012	511	94	take bread, open peanut, close jam, pour milk, sandwich bread	Video, Audio, Gaze
GTEA Gaze+ [169]	2012	3371	44	prepare American breakfast, prepare pizza, prepare snack, prepare greek salad, prepare pasta salad, prepare turkey sandwich and prepare cheeseburger	Video, Audio, Gaze
JPL [8]	2013	57	7	shaking hand, hugging, petting, waving hand, pointing, punching, throwing	Video
LENA [170]	2014	260	13	read, watch videos, walk straight, walk up and down, drink, housework	Video
HUJI [65, 87]	2014	148	9	driving, biking, motorcycle, walking, boxing, horseback, running, skiing, stair climbing	Video, Audio
BEOID [23]	2014	742	34	prepare coffee using the machine, place the cup on the mat, add sugar, check the printer is loaded with paper manually, use the keypad, adjust the seat, adjust chest pad, weight then use the machine	Video, Gaze
MEAD [31]	2015	200	20	cycling, doing push-ups, making phone calls, running, sitting, working at PC	Video, Audio, Wearable Sensors
FPVSum [166]	2018	124	8	biking, horseback, skiing, longboarding, rock climbing, scuba, skateboarding, surfing	Video, Audio
First-Person Hand Action (FPHA) [171]	2018	1175	45	pour juice, sprinkle spoon, scoop spoon, put tea bag, wash sponge, read the paper, use calculator, high five, pour wine, pay coin, open wallet	Video, Depth Map, 3D hand poses, 6D Object Pose
EPIC-Kitchens [172]	2018	50547	2747	pour tofu onto the pan, open the bin, take onion, pick up a spatula, pour pasta into the container, open fridge, put down tofu container, pick up the bag, cut onion	Video, Audio
EGTEA Gaze+ [111]	2018	10325	106	inspect/read a recipe, open fridge, put eating utensil (somewhere), spread condiment (on) bread (using) eating utensil, pour oil (from) oil container (into) pan	Video, Audio, Gaze
Charades-Ego [173]	2018	30516	157	washing a window, holding a broom, closing a refrigerator, putting broom somewhere, opening a refrigerator, tidying up with a broom, lying on a bed, taking a broom, washing a mirror, drinking from a cup	Video
EPIC-Tent [174]	2019	921	11	pickup stake bag, open support bag, open tent bag, spreadtent, tie top	Video, Gaze
EPIC-Kitchens-100 [175]	2020	89979	4025	clean pan, dry hand, slice chili, take a banana, put bag, squeeze lemon, put plate	Video, Audio
H20 [176]	2021	184	36	read a book, take out cocoa, grab chips, place book, apply the spray, close milk	Video, 3D hand poses, 6D object poses

The resulting statistics after selecting the video clips are given below:

- **HUJI:** 49 distinct videos - 148 video clips for 9 activities (*driving, biking, motorcycle, walking, boxing, horse riding, running, skiing, stair climbing*)
- **FPVSum:** 39 distinct videos - 124 video segments for 8 activities (*biking, horse riding, skiing, longboarding, rock climbing, scuba, skateboarding, surfing*)
- **YouTube:** 216 distinct videos - 1120 video clips for 27 activities (*american football, basketball, bungee jumping, driving, go-kart, horse riding, ice hockey, jet ski, kayaking, kitesurfing, longboarding, motorcycle, paintball, paragliding, rafting, rock climbing, rowing, running, sailing, scuba diving, skateboarding, soccer, stair climbing, surfing, tennis, volleyball, walking*)

Table 14 provides a comprehensive listing of the overall number of videos and clips for each activity, as well as the total duration of videos. Multiple video clips depicting egocentric activities may be included in a video. Therefore, video clips were gathered from manually designated time intervals in videos, and each video clip contains only a single activity. As a result, we did not use any video segmentation technique to localize activities in time.

In this study, a three-stream architecture (RGB, Flow, and Audio) is preferred to test the proposed solution. For that reason, RGB, Optical Flow, and LMS frames were extracted for the videos in the dataset. The following section explains the extraction of data frames used with deep networks in detail.

Table 14: EOAD activity videos summary

Actions	Total Videos	Total Clips	Total Duration (hh:mm:ss)
American Football	11	79	00:12:31
Basketball	8	72	01:50:20
Biking	12	26	03:06:39
Boxing	15	23	00:56:38
Bungee Jumping	13	15	00:05:29
Driving	8	37	01:31:32
GoKart	8	11	01:11:32
Horse Riding	10	12	02:38:19
Ice Hockey	8	108	01:16:55
Jetski	8	15	00:45:00
Kayaking	10	54	01:08:39
Kitesurfing	8	53	00:29:01
Longboarding	10	13	00:42:54
Motorcycle	12	49	01:24:01
Paintball	15	15	00:54:52
Paragliding	17	19	00:58:48
Rafting	10	10	00:29:21
Rock Climbing	10	10	01:30:27
Rowing	11	11	01:03:52
Running	9	51	02:51:24
Sailing	10	17	01:09:52
Scuba Diving	10	10	01:17:37
Skateboarding	9	131	00:24:58
Skiing	11	38	03:29:29
Soccer	13	170	01:08:50
Stair Climbing	12	17	01:43:12
Surfing	10	50	00:26:47
Tennis	9	52	00:36:21
Volleyball	9	129	00:45:58
Walking	8	95	01:31:51
Total	30	1392	37:43:08

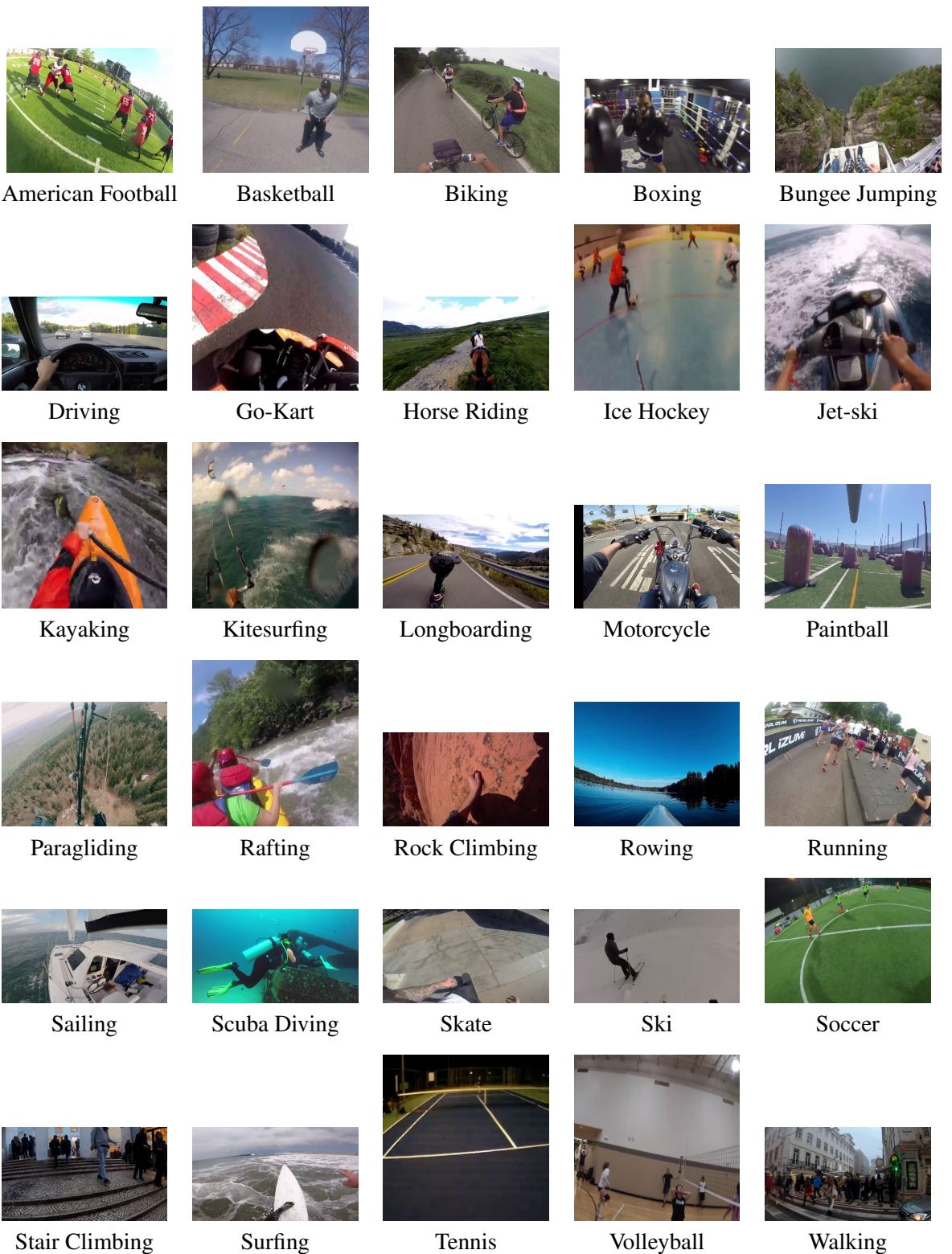


Figure 23: EOAD activities.

4.2.2 Frame Extraction and Data Augmentation

The details of frame extraction and the selected data augmentation methods for each modality are explained in the following sections.

4.2.2.1 RGB

RGB frames were extracted using the FFmpeg video and audio converter tool [177] with their native resolutions. After that, the resolutions were rescaled to 320x240. In addition, the multi-scale cropping [178] technique was employed as data augmentation that was adapted to the task of action recognition. For that purpose, we fixed the input image size to 320 x 240 pixels and randomly chose the cropping width and height from the range 256,224,192,168. The cropped regions were then resized to 224x224. Importantly, this cropping method introduces multi-scale augmentation and aspect ratio augmentation. Because, images generally contain more information than the corners, the center portion of the images (224x224) was cropped for the test phase. Figure 24 depicts some of the samples for RGB frames.

4.2.2.2 Optical Flow

The motion information of activities was modeled with optical flows. For that purpose, a code repository [179] on GitHub was utilized in which optical flows were estimated by the TVL1 method [180]. The estimation was performed over native resolutions of RGB frames which were then resized to a resolution of 320x240.

The motion directions were insignificant for the activities in our dataset. Therefore, horizontal flip augmentation was employed by reversing the rows of pixels to make the models robust against the changes in motion directions. Similar to RGB frames, the center portion of optical flow frames (224x224) was cropped during tests. Figure 25 shows some of the frames estimated for optical flow.

4.2.2.3 Audio

The LMS is a pretty effective method for studying an audio file’s spectral and temporal evolution. The Mel scale is a perceptual pitch scale proposed by [181] in 1940. In particular, the Mel scale attempts to imitate the non-linear human ear’s perception of sound by being more discriminative at lower frequencies and less at higher frequencies. Below is the relationship between pitch (in Mel scale) and frequency (in Hz):

$$p = Mel(f) = 2595 * \log \left(1 + \frac{f}{700} \right) \quad (16)$$

where $p = Mel(f)$ indicates the perceived pitch $p[Mel]$ of a sound at frequency $f[Hz]$.



American Football



Basketball



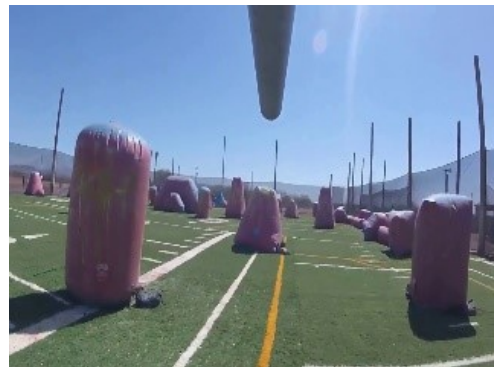
Go-kart



Horseback



Ice Hockey



Paintball

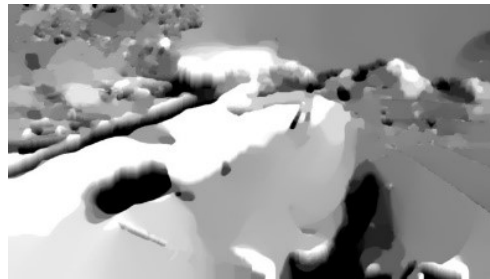
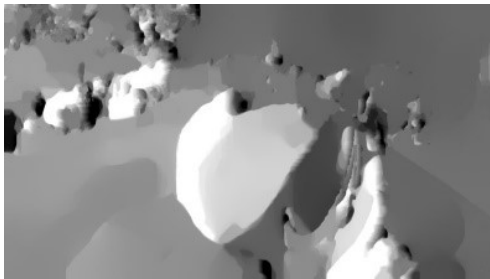


Paragliding

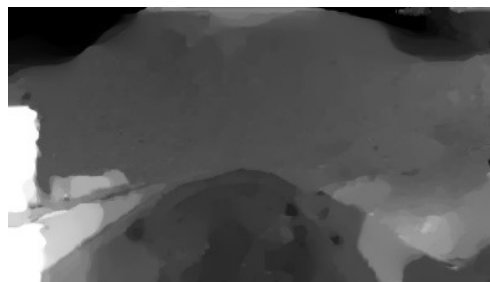


Tennis

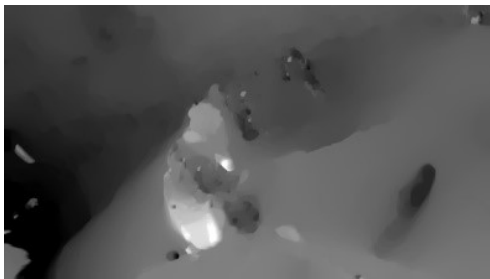
Figure 24: Sample feature frames for RGB.



Basketball



Go-kart



Ice Hockey



Tennis

Figure 25: Sample feature frames for Optical Flow (left: horizontal, right: vertical).

Inversely, $f = Mel^{-1}(p)$ can be stated as the inverse connection, allowing the frequency (Hz) to be calculated from the pitch (p) (Mel). The behavior of the human ear can be reproduced using the so-called Mel filterbank, a collection of K triangular filters where each filter has a maximum response at the center frequency and drops linearly toward 0 until reaching the center frequency of the two neighboring filters. Specifically, the Mel scale filter centered on pitch p can be modeled as follows:

$$\begin{cases} \frac{f - Mel^{-1}(p-1)}{Mel^{-1}(p) - Mel^{-1}(p-1)}, & Mel^{-1}(p-1) \leq f < Mel^{-1}(p) \\ \frac{Mel^{-1}(p+1) - f}{Mel^{-1}(p+1) - Mel^{-1}(p)}, & Mel^{-1}(p) \leq f < Mel^{-1}(p+1) \\ 0, & otherwise \end{cases} \quad (17)$$

The whole Mel filterbank can be described as a two-dimensional matrix H of size $F \times K$, where columns include the coefficients associated with the various filters $H_p(f)$ (corresponding to K distinct pitches) and rows corresponding to frequencies.

By applying the Mel filterbank H to the spectrogram of an audio signal, the Log-Mel Spectrogram (LMS) may be computed, which is an important and extensively used speech and audio processing tool [182–184]. LMS can be represented as a 2D matrix L of size $T \times K$ for a signal assessed over T temporal samples and F frequency bins. The matrix is computed as follows:

$$L = \ln(S \cdot H + \epsilon) \quad (18)$$

where S is a 2D matrix with size $T \times F$ containing the spectrogram of the audio signal (i.e., the magnitude of the Short-Time Fourier Transform (STFT), with frequency information along columns and time information along rows), \cdot computes the matrix multiplication, $\ln(\cdot)$ computes the natural logarithm, and ϵ is a small constant used to avoid feeding zeros to the logarithm. The resulting LMS brings information about the spectral content of the audio signal (in Mel scale) as a function of the temporal evolution: along columns, pitches are founded in Mel scale; along rows, the temporal evolution.

The configuration to extract the LMS feature in this work is given below:

- Audio sampling rate : 16 kHz
- Window length : 0.025 sec
- Hop length : 0.010 sec
- Number of Mel bins : 64

Figure 26 shows some samples of LMS features for the selected egocentric activities represented as RGB frames.

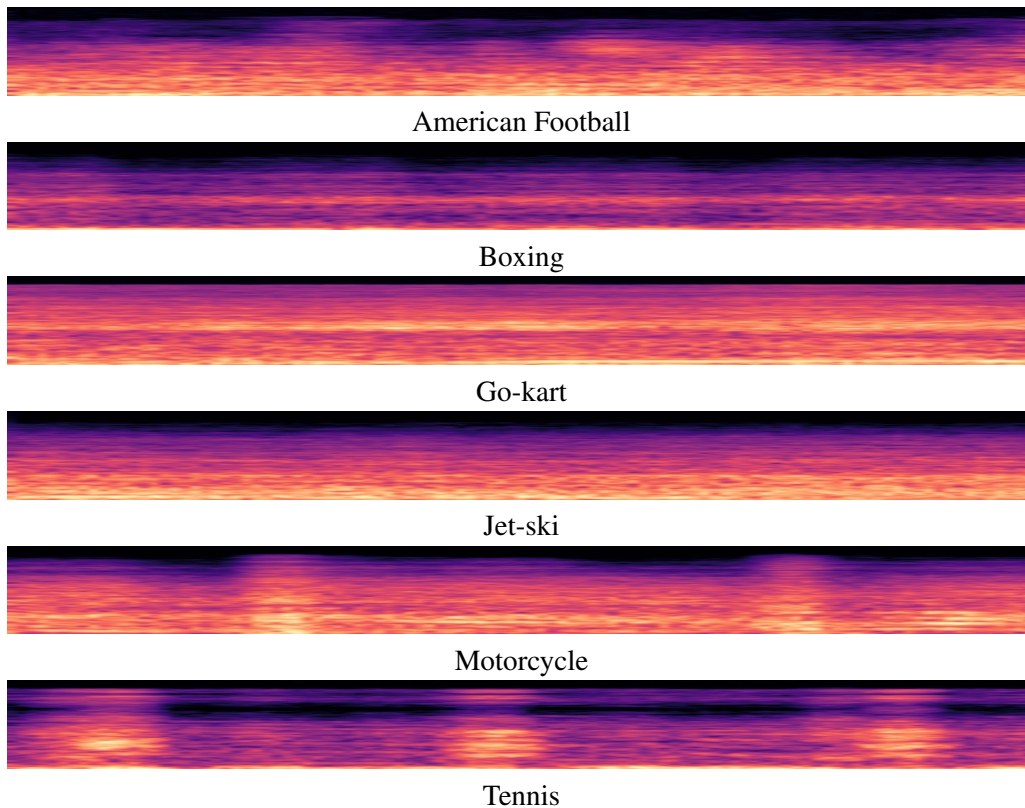


Figure 26: Sample LMS feature frames for Audio.

4.2.3 Dataset Splitting

The video clips used for training, validation and test sets for each activity are listed in Table 15. As mentioned before, multiple video clips may belong to a video because of trimming it for some reasons (i.e., scene cut, temporary overlaid text on videos, or video parts unrelated to activities). In addition, we know that each video clips contain single activity. Therefore, we did not apply any video segmentation method to localize activities.

The minimum number of videos for each activity was selected as 8, and the video samples in the experimental setup were divided as 50%, 25%, and 25% for training (4 videos), validation (2 videos), and test (2 videos), respectively. On the other hand, videos were split according to the raw video footage to prevent the mixing of similar video clips (having the same actors and scenes) into training, validation, and test sets simultaneously. Therefore, we ensured that the video clips trimmed from the same videos were split together into training, validation, or test sets to make a fair comparison.

Some activities have continuity throughout the video, such as *scuba*, *longboarding*, or *riding horse* which also have an equal number of video segments with the number of videos.

However, some activities, such as skating, occurred in a short time, making the number of video segments higher than the others. As a result, the number of video clips for training, validation, and test sets was highly imbalanced for activities (i.e., *jetski* and *rafting* have 4; however, *soccer* has 99 video clips for training).

Table 15: Dataset splitting for EOAD

Action Label	Train		Validation		Test	
	#Segments	Total Duration	#Segments	Total Duration	#Segments	Total Duration
AmericanFootball	34	00:06:09	36	00:05:03	9	00:01:20
Basketball	43	01:13:22	19	00:08:13	10	00:28:46
Biking	9	01:58:01	6	00:32:22	11	00:36:16
Boxing	7	00:24:54	11	00:14:14	5	00:17:30
BungeeJumping	7	00:02:22	4	00:01:36	4	00:01:31
Driving	19	00:37:23	9	00:24:46	9	00:29:23
GoKart	5	00:40:00	3	00:11:46	3	00:19:46
Horseback	5	01:15:14	5	01:02:26	2	00:20:38
IceHockey	52	00:19:22	46	00:20:34	10	00:36:59
Jetski	4	00:23:35	5	00:18:42	6	00:02:43
Kayaking	28	00:43:11	22	00:14:23	4	00:11:05
Kitesurfing	30	00:21:51	17	00:05:38	6	00:01:32
Longboarding	5	00:15:40	4	00:18:03	4	00:09:11
Motorcycle	20	00:49:38	21	00:13:53	8	00:20:30
Paintball	7	00:33:52	4	00:12:08	4	00:08:52
Paragliding	11	00:28:42	4	00:10:16	4	00:19:50
Rafting	4	00:15:41	3	00:07:27	3	00:06:13
RockClimbing	6	00:49:38	2	00:21:59	2	00:18:50
Rowing	5	00:47:05	3	00:13:21	3	00:03:26
Running	21	01:21:56	19	00:46:29	11	00:42:59
Sailing	7	00:39:30	4	00:14:39	6	00:15:43
Scuba	5	00:35:02	3	00:23:43	2	00:18:52
Skate	91	00:15:53	30	00:07:01	10	00:02:03
Ski	14	01:48:15	17	01:01:59	7	00:39:15
Soccer	102	00:48:39	52	00:13:17	16	00:06:54
StairClimbing	6	01:05:32	6	00:17:18	5	00:20:22
Surfing	23	00:12:51	17	00:06:52	10	00:07:04
Tennis	34	00:27:04	9	00:06:03	9	00:03:14
Volleyball	87	00:19:14	35	00:07:46	7	00:18:58
Walking	49	00:43:02	36	00:38:25	10	00:10:23
	740	20:22:37	452	09:20:23	200	08:00:08

4.2.4 EAR Model Training

The first phase of the proposed framework is to train EAR models using the training frames. Here, two different training strategies were adopted one of them corresponds to the deep feature-based solution and the other served for the VAEs-based approach. Both of them employed ResNet50 v1.5 (pre-trained with ImageNet [185]) as their backbone models.

The deep feature-based strategy used a pre-trained ResNet50 model as a feature extractor for RGB, Optical-Flow, and Audio streams. The information at the average pooling layer was utilized as deep features for modalities. Firstly, each video clip was divided into K equal parts in time corresponding to video segments. In this research, the number of segments was selected as 25 as in [2]. In other words, each video clip was represented with 25 samples. After that, the sample frames were selected within the pre-defined segments. While the samples were randomly selected within the segments in the training set, they were chosen as the middle frames for validation and test sets. Lastly, Support Vector Machines (SVM) classifiers were employed as EAR models with a grid-search mechanism using *linear*, *RBF*, and *polynomial* kernels. As a result, three independent SVM-based EAR models were trained for each modality.

On the other hand, ResNet50 models were fine-tuned for the VAEs-based regression strategy. For that purpose, a fully-connected layer was appended to the ResNet model with an additional Softmax layer as shown in Figure 27. We also applied a grid search according to the learning rate, and dropout ratio, during the training EAR models for each modality.

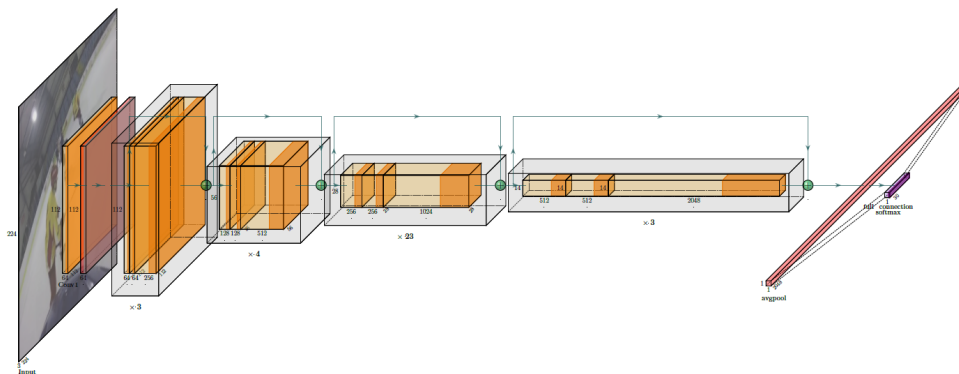


Figure 27: ResNet50 architecture for RGB stream

4.2.5 Feature Fusion

The proposed regression models employed the fused deep features as inputs. Therefore, it is necessary to combine the features coming from three modalities for model training. To do this, we used various fusion strategies taken from RGB, optical flow, and audio streams (Table 16). Additionally, various scalers were utilized, as shown in Table 17. Finally, the optimal configuration according to the results for the related regression model was chosen.

As the regression models utilize the fused features to weigh the streams, the efficiency of the feature fusion is a crucial point that should be emphasized. The fused features should contain coherent information from all modalities to anticipate their relevance. In addition to conventional fusing strategies such as summing or averaging features, we employed a pooling mechanism called Compact Bilinear Pooling [186] for deep feature fusion. For a two-stream case, Bilinear Pooling [153] combines streams while conserving spatial information to increase task performance. The result of our bilinear pooling layer is $x = f_{s1}f_{s2}$, where f_{s1} and f_{s2} are the first and second stream features, respectively. Sum pooling compresses spatial features before classification. Prior to assigning features to the regressors, the signed square root ($x \leftarrow \text{sign}(x)\sqrt{|x|}$) and L_2 normalization is applied. Compact Bilinear Pooling, as proposed in [186], was used to reduce memory and accelerate training without sacrificing performance.

Table 16: Deep feature fusion formulations

Feature Fusion	Definition
Feature sum	$x = f_{rgb} + f_{flow}$
Feature mean	$x = (f_{rgb} + f_{flow})/2$
Feature max	$x = \max(f_{rgb} + f_{flow})$
Feature append	$x = [f_{rgb}, f_{flow}]$
Compact Bilinear Pooling	[186]

Table 17: Deep feature scalers

Feature Scaling	Definition
Standard scaling	Standardize features by removing the mean and scaling to unit variance
Minimum maximum scaling	Transform features by scaling each feature to a given range
Maximum absolute scaling	Scale each feature by its maximum absolute value
Robust scaling	Scale features using statistics that are robust to outliers
Normalizer	Normalize samples individually to unit norm
Power transformer scaling	Apply a power transform featurewise to make data more Gaussian-like

4.2.6 Stream Decision Fusion

The configurations used for experiments to fuse stream decisions will be explained in this section. The following section contains the details of the deep feature-based stream fusion approach. After that, the configuration of the VAEs-based stream fusion model will be given.

4.2.6.1 Deep Feature-based Stream Fusion

After feature fusion, the fused features were used to train multivariate regression models listed in Section 4.1.1.1. Scikit-learn package (v1.0.2) [187] was utilized for the implementation of linear, k-nearest neighbor, decision tree, random forest, support vector (direct and chained multi-output), Lasso and Ridge regression models. In addition, XGBoost (v1.4.2) and NGBoost (v0.3.12) packages were also used independently to be able to use XGBoost and NGBoost regressors.

A Hyperparameter search was conducted for each regression model with the given function interfaces. Then, the modality weights computed from test samples were compared with the weights obtained by the trained regression models to determine if the model can accurately predict the stream weights.

4.2.6.2 VAEs-based Stream Fusion

VAEs regression model was implemented based on the GitHub repository [188] of the first author of [161]. We performed an ablation study for latent vector dimensions and the number of coder & decoder blocks. The best configuration was selected according to Mean Absolute Error (MAE) performance metric. The final configuration for the selected VAE regressor is given below:

- Feature dimension: 2048
- Latent space dimension: 16
- Number of code blocks for encoder & decoder: 4
- Activation function for code blocks: ReLU
- Optimizer: Adam
- Sample weighting for loss: No
- Label loss type: L1
- Learning rate: 10^{-4}
- Dropout: 0.8

4.2.7 Class-aware Segment Fusion

Practically, three different configurations were set for segment fusion models. The first one learns segment weights from all available samples without knowing their class labels (class-independent segment weighing). The second one adopted a class-aware fusion strategy using only the segment weights of the selected classes for training. Here, a set of regression models were trained independently for each class. Lastly, we adopted another class-aware approach,

however, this time the training sets were populated with the samples of other classes as well as the selected class samples. The weights of other classes are set to 0 which also means the model is expected to produce 0 weight for misclassified segments. To satisfy dataset balance, we set the total number of samples for other classes equal to the number of the selected class samples.

For experiments, we used the same set of regression models, feature fusion and scalers explained in Section 4.1.1.1 and 4.2.5, respectively. Firstly, the experiments were performed for RGB-Flow input to evaluate three different configurations. After that, the best configuration was selected for the following tests. The experimental results showed that the class-independent stream-weighting model could not learn the target weights effectively. The second configuration which is the training of class-aware fusion models with only the weights of the selected class samples achieved better results compared to the class-independent fusion strategy. However, the third configuration which uses the weights of other classes (as 0) and the selected classes had the best performance with minimum validation errors. As a result, the third configuration of segment fusion was used for future tests.

4.3 Results

In this section, the results of the proposed stream and segment weighing mechanism will be discussed in detail. Firstly, we shared the results for single modalities without a fusion mechanism. The case of fusing decisions with equal weights at the segment and video levels was taken as the baseline. On the other hand, we needed to validate the proposed confidence-based fusion process at first. To do this, the ideal cases were investigated while the fusion of stream and segment decisions is weighted based on the confidence values of actual classes. After that, model-based fusion results of two different stream fusion procedures are explained one of which uses deep features with various regressors and the other utilizes VAEs for the regression problem. Later, the outcomes of the class-aware segment fusion model are discussed. Finally, the stream and segment fusion models giving the best performances were concatenated to evaluate their effects on EAR performances.

4.3.1 Single Stream

Single-stream performances for each modality were selected as baselines before the fusion process. For that purpose, each EAR model was trained independently. It is apparent from Table 18, which shows the classification accuracies of the segment and video levels, appearance (RGB) provides the most discriminative information to recognize egocentric activities compared to motion and audio streams. Additionally, we observed that the aggregation of the segment decisions boosts the overall classification accuracy for all modalities at the video level. However, performance improvements were not at the same level for all modalities. For example, using the average of segment decisions almost doubled video level accuracy (from 31.14% to 57.50%) for motion stream. On the other hand, the aggregation of segment decisions did not make any significant effect on the precision of video decisions for the audio stream (from 24.16% to 25.50%).

Table 18: Single stream performances for RGB, Flow, and Audio

Stream Type	Classification Accuracy (%)	
	Segment-Level	Video-Level
RGB	59.80	78.50
Optical Flow	31.14	57.50
Audio	24.16	25.50

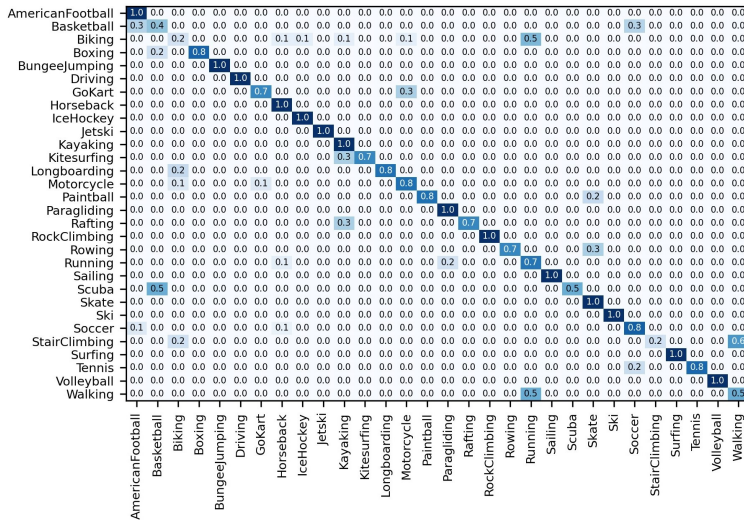
The comparative results of the confusion matrices at the segment and video levels are given in Figure 28 and Figure 29, respectively. The confusion matrix of the RGB stream indicates that the activity groups having similar visual scenes can be mixed up such as {*walking, running, stairclimbing*}, or {*rafting, kayaking*}. On the other hand, the activities having unique visual identifiers (steering wheel for *driving*, horses’ mane for *horseback*, or rock textures for *rockclimbing*) were recognized accurately. In general, the recognition characteristics of motion stream are similar to RGB except for the activities including the low magnitude of optical-flow vectors (i.e., *scuba*) or having similar motion dynamics (e.g., *americanfootball*, and *soccer*). Even if an audio stream can not discriminate most activities (e.g., *biking, horseback, icehockey, kayaking, longboarding, paintball, rockclimbing, rowing*), it had a greater classification accuracy for some of them such as *driving, ski, and surfing* (over 70%).

4.3.2 Ideal Case for Stream and Segment Fusion

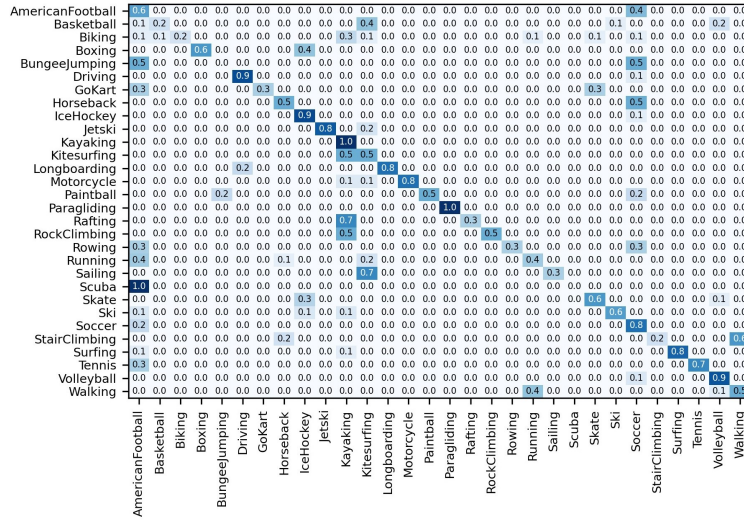
We conduct some experiments to validate the proposed method in the ideal case. Here, we defined the ideal case as computing stream and segment weights based on the confidence scores of actual classes.

4.3.2.1 Stream Fusion

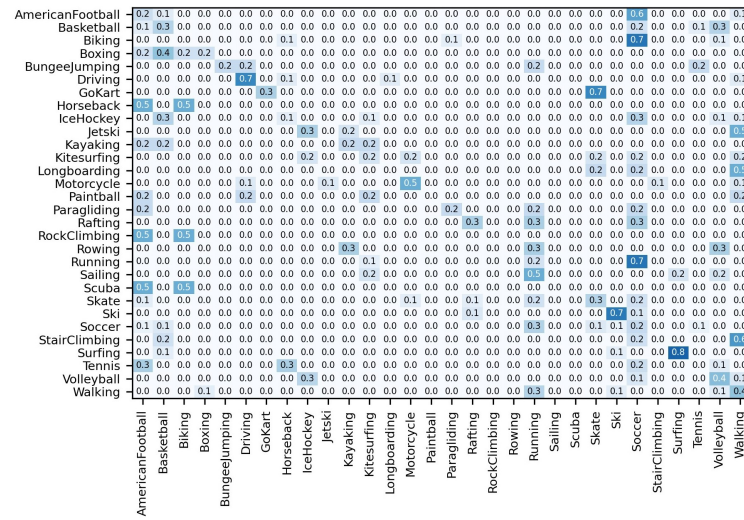
The proposed framework offers a solution to weight stream decisions based on the normalized confidence scores of single-stream EAR models for the actual classes. Therefore, we computed the stream weights for the ideal case at first and then fuse stream decisions according to the computed normalized weights. Table 19 depicts baseline and ideal cases for each combination of modalities. The results demonstrated the practical limits of the proposed solution.



RGB



Optical-Flow



Audio

Figure 29: Confusion matrices for single streams at video-level

Table 19: Ideal case for stream weighing

Stream Type			Segment-Level			Video-Level		
RGB	Flow	Audio	Stream Weighing	Accuracy (%)	Best Single (%)	Segment Weighing	Accuracy (%)	Best Single (%)
Selected	Selected	Not Selected	Equal	60.28	59.80	Equal	79.00	78.50
			Ideal	67.38	59.80		87.00	78.50
Selected	Not Selected	Selected	Equal	51.85	59.80		62.00	78.50
			Ideal	66.58	59.80		81.50	78.50
Not Selected	Selected	Selected	Equal	31.50	31.14		34.50	57.50
			Ideal	44.80	31.14		63.00	57.50
Selected	Selected	Selected	Equal	56.08	59.80		67.50	78.50
			Ideal	71.68	59.80		86.50	78.50

From Table 19, we can see that weighing streams according to the confidence scores of actual classes increases segment recognition accuracies for all multi-stream cases (RGB-Flow: 60.28% -> 67.38%, RGB-Audio: 51.85% -> 66.58%, Flow-Audio: 31.50% -> 44.80%, RGB-Flow-Audio: 56.08% -> 71.68%). Increasing segment recognition accuracy was also indirectly increases video level performances (RGB-Flow: 79.00% -> 87.00%, RGB-Audio: 62.00% -> 81.50%, Flow-Audio: 34.50% -> 63.00%, RGB-Flow-Audio: 67.50% -> 86.50%). As a result, the proposed confidence-based weighing scheme achieved better results in ideal conditions compared to the results of averaging decisions.

Exceeding only the accuracy of the baseline method (in our case, equally weighing decisions) is not enough to declare the efficiency of the proposed fusion mechanism. The fusion results should also achieve at least the best single-stream performances. Otherwise, it has no meaning to use multi-stream for the specified tasks. Therefore, the best single-stream performances were also listed in Table 19 to show another baseline for fusion models. It is clear that stream weighing can improve the accuracy compared to the baseline method and the best single-stream configurations for all multi-stream combinations. The results showed the great potential of a confidence-based weighing scheme.

Another important point worth mentioning is the relationship between segment and video accuracy. It is apparent that increasing the precision of segment classification contributes to the video-level results. However, better segment accuracy does not always necessitate to get better video classification results. For example, although RGB-Flow-Audio had the highest segment classification score (71.68%), its video-level accuracy (86.50%) was lower than RGB-Flow (87.00%). Additionally, the difference between segment level accuracy was lower than 1% for RGB-Flow and RGB-Audio whereas video-level classification of RGB-Flow was 5.50% better than RGB-Audio. These findings clearly showed that there is not a linear relationship between segment and video-level results. The most likely explanation for this situation is that some segments have a greater impact on changing the state of video-level predictions from *False* (i.e., misclassified) to *True* (i.e., correctly classified).

4.3.2.2 Segment Fusion

Segment fusion models were employed to weigh segment decisions in accordance with the sum of confidence scores acquired from single-stream EAR models. Table 20 shows the recognition results after weighing the fused segment decisions (averaged in this case) based on the confidence scores. In this case, segment decision fusion has only an effect on video-level performance. Similar to stream fusion, confidence-based segment fusion also increased the recognition performance for all multi-stream combinations (RGB-Flow: 79.00% -> 91.00%, RGB-Audio: 62.00% -> 74.00%, Flow-Audio: 34.50% -> 52.00%, RGB-Flow-Audio: 67.50% -> 81.50%). However, weighing segment decisions could not exceed the best single-stream EAR accuracy for RGB-Audio and Flow-Audio.

Table 20: Ideal case for segment weighing

Stream Type			Segment-Level			Video-Level		
RGB	Flow	Audio	Stream Weighing	Accuracy (%)	Best Single (%)	Segment Weighing	Accuracy (%)	Best Single (%)
			Equal	60.28	59.80	Equal	79.00	78.50
						Ideal	91.00	
				51.85	59.80	Equal	62.00	78.50
						Ideal	74.00	
				31.50	31.14	Equal	34.50	57.50
						Ideal	52.00	
				56.08	59.80	Equal	67.50	78.50
						Ideal	81.50	

4.3.2.3 Stream and Segment Fusion

After validating stream and segment fusion for the ideal case, these two fusion procedures were concatenated using ideal weights. Table 21 depicts the classification accuracies for both segment and video levels. Combining confidence-based weighing at the segment and video levels had significant improvement in recognition performances. For example, using only confidence-based stream weighing for RGB-Flow increased the classification accuracy from 60.28% to 67.38% which indirectly increased video classification accuracy (for equal segment weighing) from 79.00% to 87.00%. However, using also confidence-based segment weighing for decisions instead of equal weighing increased the accuracies from 87.00% to 95.00% which provides a significant improvement compared to the baseline having 78.50% video classification accuracy. It should also be emphasized that using only segment weighing for RGB-Audio and Flow-Audio could not improve video classification results. However, combining segment weighing with confidence-based stream weighing helps exceed baseline scores significantly (RGB-Audio: 74.00% to 95.00%, Flow-Audio: 52.00% to 85.00%).

Table 21: Ideal case for stream and segment weighing

Stream Type			Segment-Level			Video-Level		
RGB	Flow	Audio	Stream Weighing	Accuracy (%)	Best Single (%)	Segment Weighing	Accuracy (%)	Best Single (%)
Selected	Selected	Not Selected	Equal	60.28	59.80	Equal	79.00	78.50
			Ideal	67.38		Ideal	95.00	
Selected	Not Selected	Selected	Equal	51.85	59.80	Equal	62.00	78.50
			Ideal	66.58		Ideal	95.00	
Not Selected	Selected	Selected	Equal	31.50	31.14	Equal	34.50	57.50
			Ideal	44.80		Ideal	85.00	
Selected	Selected	Selected	Equal	56.08	59.80	Equal	67.50	78.50
			Ideal	71.68		Ideal	95.00	

Figure 30 depicts the state transitions of segments and videos from *False* (misclassification) to *True* (correct classification) after applying a confidence-based weighing scheme for the ideal case. The first finding is that the accuracies of segment classification can be improved for all activities. Another important finding is that increasing recognition accuracy at the segment level does not always guarantee improvement at video level classification, especially for the activities recognized with high accuracies such as *american football*, *driving*, *paragliding*, or *volleyball*. Lastly, the transition from *False* to *True* for some segments has a more significant impact on the results of video classification. For instance, changing the state of decisions of 27 segments from *False* to *True* for *motorcycle* provided the change of four videos from *False* to *True*. However, using confidence-based segment weighing fusion changed the states of approximately 60 segments for *soccer* activity which means helping only three videos to change their states from *False* to *True*.

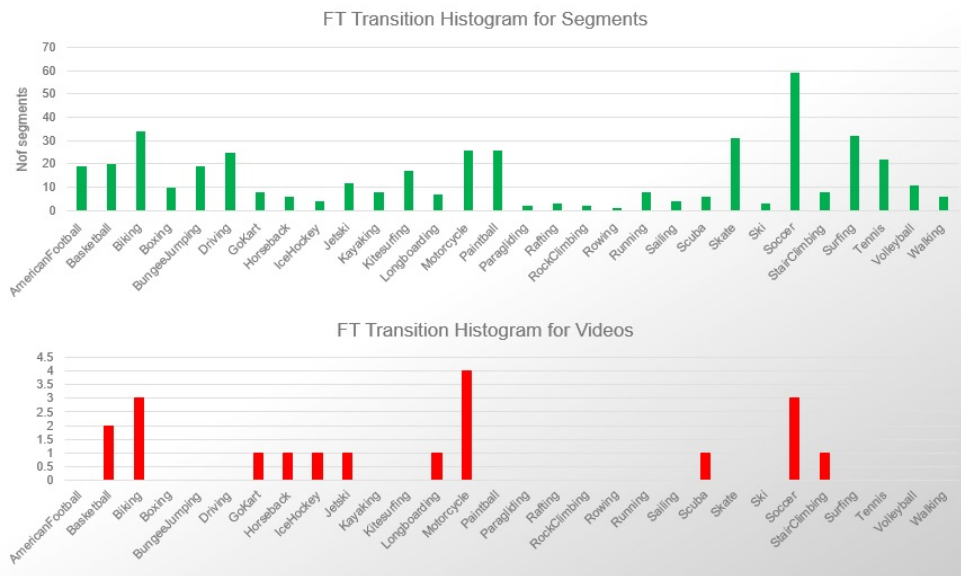


Figure 30: *False* to *True* state transitions for ideal case.

4.3.3 Model-based Stream Fusion

In this section, the results of deep feature-based and VAEs-based stream weighing models will be given in detail.

4.3.3.1 Deep Feature-based Stream Fusion

We utilized a parameter search approach for each regressor and chose the configuration that yielded the lowest MAE scores during training. The chosen model was employed to weigh the segment-level decisions for each modality while using equal weights for segment decisions at video levels. Another analysis was conducted to identify the segments whose predictions altered after segment fusion according to weights produced by regression models. This enables us to comprehend the impact of using regression weights. Table 22 shows the experimental outcomes of the deep feature-based regressors in which '*TF*' and '*FT*' represent the state transitions of decisions at the segment and video-levels after applying the model-based weights. Here, '*TF*' indicates the segments whose state was changed from *True* to *False*, and '*FT*' indicates those whose state was changed from *False* to *True*, compared to the baseline. In addition, percentage gains in accuracy were provided for both the segment and video levels.

According to Table 22, XGBoost yielded the best performance by giving the minimum MAE score. In ideal conditions (shown in the last row in Table 22), the status of 439 segments can be changed from *False* to *True*, corresponding to 8.60% improvement in segment classification accuracy. According to our findings, the optimal model altered the status of 225 segments from *False* to *True*, whereas 83 segments were changed from *True* to *False*. Moreover, the increase at the segment level did not result in an improvement at the video level; instead, the video classification results decreased. Similar outcomes were also reported for several other regressors (i.e., decision tree, SVR using Polynomial kernels, Ridge, or NGBoost regressors). In conclusion, the improvement in segment classification accuracy is not necessarily accompanied by an increase in the video-level classification, as segments shifting from *False* to *True* may not contribute to the video-level decisions.

Stream weighing models are employed to improve the classification accuracy of segments. However, it also indirectly enhances video classification accuracy since more accurate segment decisions produce better video classification results. On the other hand, the proposed fusion method should also exceed the best single-stream results for the selected modality types. Consequently, the evaluation was performed according to two criteria; the model-based fusion results compared with baseline&ideal fusion as well as the best single-stream results for the selected modality types. Table 23 includes the related comparison scores in which there are several definitions in *Weighing Strategy* heading as *Equal*, *Model*, and *Ideal*. *Equal* means the decisions were weighted equally. *Model* corresponds to using model-generated (the best model which was XGBoost in this work - Table 22) weights for stream fusion. Lastly, *Ideal* denotes the weights computed by the confidence scores of actual classes.

Table 22: Deep feature-based stream weighing results

Model		MAE \pm Std	Segment-Level		Video-Level	
			TF	FT	TF	FT
Linear		0.24 \pm 0.29	32	99	2	2
kNN		0.23 \pm 0.27	52	137	1	3
Decision Tree		0.24 \pm 0.28	38	70	3	2
Random Forest		0.25 \pm 0.29	46	133	1	3
Multi-output SVR	Linear	0.25 \pm 0.29	12	69	1	1
	RBF	0.23 \pm 0.27	53	176	4	5
	Polynomial	0.24 \pm 0.28	31	103	3	2
Chain SVR	Linear	0.24 \pm 0.29	18	79	1	2
	RBF	0.23 \pm 0.27	47	164	3	5
	Polynomial	0.24 \pm 0.28	30	99	3	2
LASSO		0.25 \pm 0.29	6	47	0	1
Ridge		0.24 \pm 0.28	27	98	3	2
XGBoost		0.23 \pm 0.27	83	225	6	5
NGBoost		0.24 \pm 0.24	31	102	3	2
Ideal Case		N/A	9	439	0	19

In this work, stream and segment fusion with equal weights were considered as the baseline. It is clear from Table 23 that model-based stream fusion improved the classification accuracies for all modality combinations (RGB-Flow: 60.28% \rightarrow 61.74%, RGB-Audio: 51.85% \rightarrow 54.64%, Flow-Audio: 31.50% \rightarrow 32.00%, RGB-Flow-Audio: 56.08% \rightarrow 61.94%). Additionally, model-based stream fusion had better recognition accuracy for all modality combinations compared to the best single-stream segment classification results except RGB-Audio. RGB-Audio had the best single-stream classification accuracy at 59.80% for segments while having 54.64% activity recognition accuracy after model-based fusion.

Indirect effects of stream fusion outcomes were observed with video-level classification results. Similar to segment-level classification results, model-based segment fusion improved video-level classification accuracy for all modalities (RGB-Flow: 79.00% \rightarrow 82.50%, RGB-Audio: 62.00% \rightarrow 69.00%, Flow-Audio: 34.50% \rightarrow 38.00%, RGB-Flow-Audio: 67.50% \rightarrow 83.00%). Additionally, model-based segment fusion had better recognition performances compared to the best single-streams for RGB-Flow (78.50% (best single) \rightarrow 82.50%), and RGB-Flow-Audio (78.50% (best single) \rightarrow 83.00%). However, video decisions using model weights remained under the best single-stream performances for RGB-Audio (78.50% (best single) \rightarrow 69.00%) and Flow-Audio (57.50% (best single) \rightarrow 38.00%).

4.3.3.2 VAEs-based Stream Fusion

We conducted some experiments to realize the learning capacity of the VAEs regressor before using it in the proposed framework. For that purpose, the model was trained with the configuration given in Section 4.2.6.2. The predicted stream weights for RGB and Flow modalities and their target values for the train set are shown in Figure 31 (a) and (b), respectively. Each

Table 23: EAR performances with deep feature-based stream weighing

Stream Type			Segment-Level			Video-Level		
RGB	Flow	Audio	Stream Weighing	Accuracy (%)	Best Single (%)	Segment Weighing	Accuracy (%)	Best Single (%)
Selected	Selected	Not Selected	Equal	60.28	59.80	Equal	79.00	78.50
			Model	61.74			82.50	
Equal	67.38	87.00						
Selected	Not Selected	Selected	Equal	51.85	59.80		62.00	78.50
			Model	54.64			69.00	
			Equal	66.58			81.50	
Not Selected	Selected	Selected	Equal	31.50	31.14		34.50	57.50
			Model	32.00			38.00	
			Equal	44.80			63.00	
Selected	Selected	Selected	Equal	56.08	59.80		67.50	78.50
			Model	61.94			83.00	
			Equal	71.68			86.50	

color code in the plots represents a different activity. On the other hand, Figure 31 (c) and (d) show the predictions of RGB and Flow weights for the validation set consisting of unseen video clips. In the ideal condition, the target and predicted weights should align on the diagonal axis in the figures. Even if the weights did not align on the diagonal axis, it is obvious that the prediction values have a correlation with the target values for both the training and validation set. The correlation for the training and validation sets show the model’s capacity to learn the labeled weights and its generalization performance, respectively.

Table 24 depicts the changes of states (*TF* or *FT*) for segment and video samples after using VAE regressor weights as stream weights. Additionally, the best configurations for deep feature-based stream regressors were selected that are listed in Table 22). The experiments demonstrated that XGBoost generated the best weights for segment fusion and improved segment-level classification accuracy (+2.84%). The VAEs regressor, on the other hand, had comparable results with XGBoost for segment-level decision weighing (+2.50%).

Table 24: VAEs-based stream weighing results

Model	MAE ± Std	Segment-Level		Video-Level	
		TF	FT	TF	FT
kNN	0.23 ± 0.27	52	137	1	3
Multi-output SVR RBF	0.23 ± 0.27	53	176	4	5
Chain SVR RBF	0.23 ± 0.27	47	164	3	5
XGBoost	0.23 ± 0.27	83	225	6	5
VAE	0.24 ± 0.24	31	102	3	2
Ideal Case	N/A	9	439	0	19

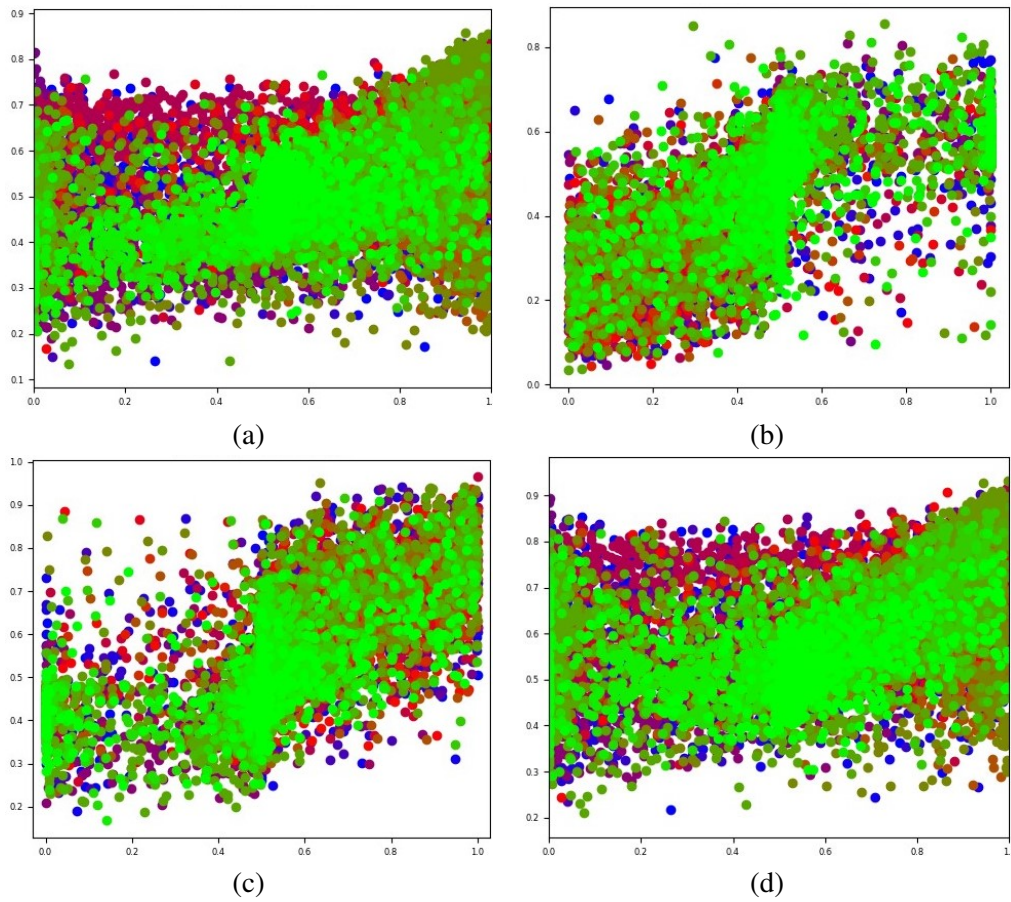


Figure 31: Predicted and target weights for VAE regressor: training set (RGB (a) and Flow (b)) and validation set (RGB (c) and Flow (d)).

Hyperparameter Tuning for VAEs Regressor

In this part, we shared the stream fusion results of various VAE configurations to realize whether a superior configuration existed. For that purpose, the numbers of coder-decoder blocks (default:4) and latent space dimensions (default:16) were altered, and the models were retrained for stream fusion.

The number of coder-decoder blocks was searched from 3 to 5 in the first place. However, changing the number of blocks had no significant effect on the positive or negative outcomes (Table 25). Secondly, various dimensions (4, 8, and 32) of latent space were tried to encode the fused features. Table 25 shows increasing or reducing the latent dimension decreased the regression performance. Therefore, the default configuration for the latent dimension was selected as 16.

Table 25: VAEs-based stream weighing results for different latent dimensions

Model	Latent Dimension	MAE \pm Std	Segment-Level		Video-Level	
			TF	FT	TF	FT
VAE	4	0.24 \pm 0.24	31	102	3	2
	8	0.23 \pm 0.27	52	137	1	3
	16	0.23 \pm 0.27	53	176	4	5
	32	0.23 \pm 0.27	47	164	3	5
Ideal Case	N/A	N/A	9	439	0	19

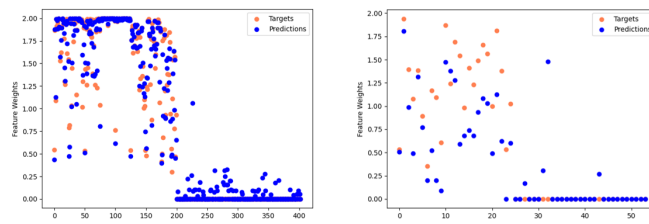
4.3.4 Class-aware Segment Fusion

The first stage of the proposed decision fusion framework aims to weigh input streams according to their information level for the related egocentric activities. However, if the information from all channels does not contain enough information for segments to recognize egocentric activities, then the importance of decisions for those segments becomes less critical. Therefore, segment decisions were weighted during video fusion in the second stage.

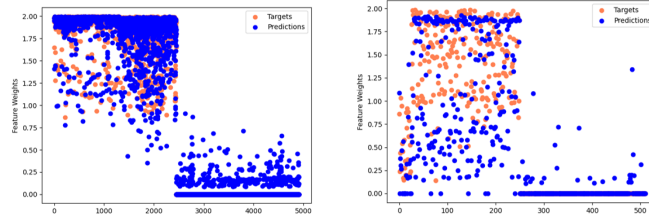
The second stage of the decision fusion framework needs the fused features and the predicted classes. The predictions after segment fusion are used to switch to the class-dependent regression models. On the other hand, the fused features were given as inputs to the selected models.

To test whether the regression model can learn segment weights and generalize them, a test set-up for RGB-Flow was prepared to receive the actual segment labels for model switching. According to this test configuration, we assumed that there is no misclassification for segments. Figure 32 shows the prediction and target weights of the train (left) and validation (right) sets for *Go-Kart*, *Ice Hockey*, *Motorcycle*, *Surfing*, *Running*, and *Driving*. The results clearly showed that the models have the capacity to learn the training weights. Additionally, they can generalize the weight prediction efficiently except for several activities such as *Running*, or *Driving*.

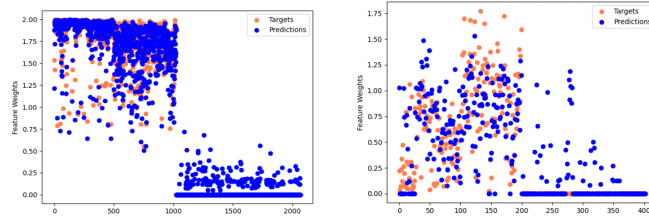
To summarize the segment fusion results, we performed various tests for all multi-modal combinations (Table 26). Considering that using the segment weighing model only affects the video-level fusion, segment-level results were ignored for this case. Therefore, we did not specify the results of segment-level performances. Similar to the stream fusion experiments, we shared the decision fusion results for *Equal*, *Model*, and *Ideal* conditions. The decision fusion with equal weights was considered as the baseline and *Ideal* cases give the practical limits for the proposed confidence-based approach. Lastly, the model-based segment fusion scores are presented to compare them with the baseline and ideal case.



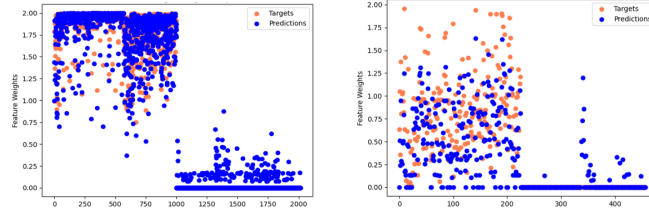
Go-Kart



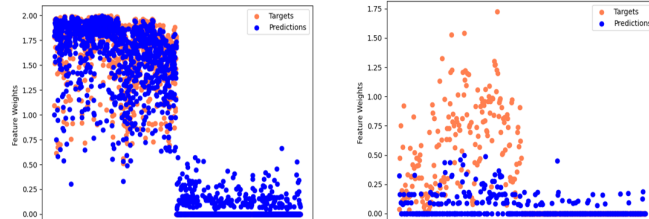
Ice Hockey



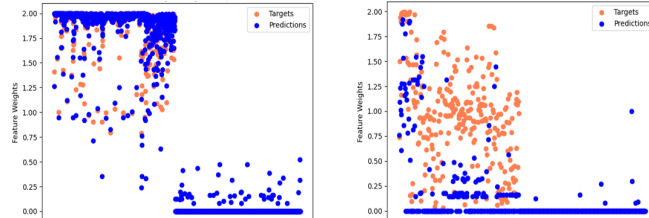
Motorcycle



Surfing



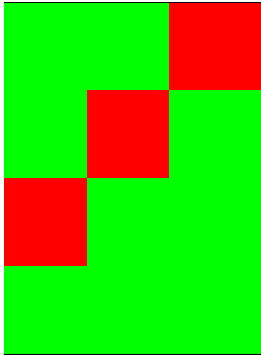
Running





Driving

Figure 32: The prediction and target segment weights for train (left) and validation (right) samples

Table 26: EAR performances with segment weighing

Stream Type			Segment-Level			Video-Level		
RGB	Flow	Audio	Stream Weighing	Accuracy (%)	Best Single (%)	Segment Weighing	Accuracy (%)	Best Single (%)
			Equal	N/A	N/A	Equal	79.00	78.50
						Model	82.00	
						Ideal	91.00	
						Equal	62.00	78.50
						Model	70.00	
						Ideal	74.00	
						Equal	34.50	57.50
						Model	40.00	
						Ideal	52.00	
						Equal	67.50	78.50
						Model	76.50	
						Ideal	81.50	

 Selected
 Not Selected

The results clearly show that using class-aware segment weighing models contributes to video-level accuracies while using the combination of two streams (RGB-Flow: 79.00% -> 82.00%, RGB-Audio: 62.00% -> 70.00%, Flow-Audio: 34.50% -> 40.00%) and three streams (RGB-Flow-Audio: 67.50% -> 76.50%) as inputs. Additionally, model-based segment-level fusion had better recognition accuracy for RGB-Flow (82.00%) when compared to the best single-stream video classification result (78.50%). However, the model-based approach could not exceed the best single-stream video accuracy for other combinations (RGB-Audio: 78.50% (best single) -> 70.00%, Flow-Audio: 57.50% (best single) -> 40.00%, and RGB-Flow-Audio: 78.50% (best single) -> 76.50%).

4.3.5 Stream and Segment Fusion

Selecting the most effective stream and segment-weighing models was the basis for the final experiments. Table 27 demonstrates the results.

The last experiments were performed by concatenating the best stream and segment weighing models (Table 27). In such a scenario, decision fusion is conducted in two stages, first at the segment level and then at the video level. The segment fusion results are the same as when using only the stream weighing models. Therefore, it is better to focus on the video-level results. These results showed us adding segment weighing models did not contribute to the video-level results except for Flow-Audio input. For Flow-Audio, the results were significantly improved at the video level (from 38% to 55.50%). However, for other combinations, video-level performances stayed at the same levels. Similar to the previous experiments, RGB-Flow and RGB-Flow-Audio inputs produced better results than the best single streams. On the other hand, RGB-Audio and RGB-Flow inputs could not reach the performances of the single streams.

Table 27: EAR performances using stream and segment weighing

Stream Type			Segment-Level			Video-Level		
RGB	Flow	Audio	Stream Weighing	Accuracy (%)	Best Single (%)	Segment Weighing	Accuracy (%)	Best Single (%)
Selected	Selected	Not Selected	Equal	60.28	59.80	Equal	79.00	78.50
			Model	61.74		Model	83.00	
			Ideal	67.38		Ideal	87.00	
Selected	Not Selected	Selected	Equal	51.85	59.80	Equal	62.00	78.50
			Model	54.64		Model	69.50	
			Ideal	66.58		Ideal	81.50	
Not Selected	Selected	Selected	Equal	31.50	31.14	Equal	34.50	57.50
			Model	32.00		Model	55.50	
			Ideal	44.80		Ideal	63.00	
Selected	Selected	Selected	Equal	56.08	59.80	Equal	67.50	78.50
			Model	61.94		Model	83.00	
			Ideal	71.68		Ideal	86.50	

4.3.6 Comparative Results

Lastly, the proposed two-stage decision fusion framework was compared to the state-of-the-art methods Stacking Ensemble [189] and MKBoost [130] in Table 28. Stacking Ensemble is one of the well-known ensemble learning techniques. Simply, it trains multiple classifiers for the same feature set. Then, train another classifier (called a meta-classifier) that takes pre-trained classifiers’ outputs as features. MKBoost, on the other hand, combines a boosting strategy with MKL learning.

The results showed that Stacking Ensemble performance worsens as the number of features increases. Additionally, MKBoost consistently produced better segment classification results. However, the proposed framework is significantly better at the video level than MKBoost.

Table 28: Comparative performances of the proposed framework with Stacking Ensemble and MKBoost

Stream Type			Segment-Level Accuracy			Video-Level Accuracy		
RGB	Flow	Audio	Stacking	MKBoost	Proposed	Stacking	MKBoost	Proposed
Selected	Selected	Not Selected	61.30	63.12	61.74	79.50	81.50	83.00
			48.32	57.22	54.64	56.50	67.00	69.50
Not Selected	Selected	Selected	29.90	33.74	32.00	36.00	41.50	55.50
			51.76	61.26	61.94	64.00	73.00	83.00

4.4 Discussion

Ideal case experiments have confirmed the observation of performance boost when the stream decisions are weighted correctly. Instead of making a brute force search for stream weights as in [2], we adopted an adaptive confidence-based fusion mechanism. However, the results had also shown the practical limits of performance boosting in the case of the stream and segment weights computed by the actual confidences of pre-trained EAR models.

The complexity of stream weighing is mainly caused by its diverse characteristics. Noisy samples may have higher weights than they should after the normalization step. For example, an RGB frame including a low entropy scene (i.e., composed of plain areas), the optical flow frames having no motion, or the audio frames having noise or silence contain irrelevant information for the selected activities. However, when these inputs are in the same segment, it becomes hard to determine the optimal weights for stream decisions. Additionally, another conflict arises because of the dependency of stream weights on the activity types. For example, a basketball hoop in an RGB frame will probably have high weight for *Basketball* activity. On the other hand, the same frame will have a lower weight for *Running* activity if an actor is running near a basketball court. That creates ambiguity by causing the same or similar frames to take different weights for different activities.

On the other hand, stream fusion models always helped to exceed the performance of baseline scores. Additionally, they also contributed to segment recognition precision compared to the best single-stream performances except for RGB-Audio input. As mentioned before, it is hard to weigh the streams efficiently without knowing their activity classes since the value of the same information differs according to its related class. Here, the segment-level classification improvement looks slightly better than the similar weighing procedure; nevertheless, the improvement at the video level was significantly better. This is most likely caused by the state transitions of segments from *False* to *True* that also make the misclassified videos to be predicted correctly.

We observed that most of the stream and segment fusion models suffered from overfitting during training except several ones such as XGBoost, and NGBBoost. In addition, XGBoost achieved the best weight estimations for both streams and segments in most cases compared to other regression models which also shows its generalization capability. XGBoost uses the power of ensemble learning by decreasing the variance of the general model through bagging, eliminating bias due to boosting, and having accurate predictions because of stacking. To prevent overfitting, XGBoost incorporates a regularized model similar to regularized greedy forest [190], however, simplifies the objective and algorithm for parallelization [153]. Moreover, in our problem using boosting mechanism may contribute to the weight estimations, especially for streams including noisy and ambiguous samples. We know that boosting is an effective way to divide samples into subsets to better discriminate them with the selected ensemble models.

Class-aware segment fusion models proved their efficiency of segment weighing in case of receiving segment labels from equally weighted stream decisions (Section 4.3.4). The proposed segment fusion models improved the video-level classification scores for all stream combinations (RGB-Flow, RGB-Audio, Flow-Audio, and RGB-Flow-Audio (Table 26). Video-level

recognition accuracy exceeded the best single-stream performance for RGB-Flow. Though, an apparent limitation of the method is that it suffers from falling behind the single-stream performances for RGB-Audio, and Flow-Audio.

Using standalone stream and segment fusion models succeeded in improving the performances of egocentric activity recognition (explained in Section 4.3.3, and 4.3.4). The anticipation of using the stream fusion model beforehand is to have better segment classification results, thereby having greater overall performance than using models independently. However, when they were used together, we could stay within the scores of using only stream and segment weighing models except for only Flow-Audio. As mentioned, segment fusion models are efficient when fed by actual segment labels or equally weighted stream decisions. Therefore, a possible explanation of these results is related to the impacts of segment state transitions to the video-level fusion. In our case, it is evident that increasing segment classification accuracy did not contribute to the model-based segment fusion results.

Stacking ensemble learning is designed to improve modeling performance, although is not guaranteed to result in an improvement in all cases. On the other hand, MKBoost consistently had better segment classification results than the proposed method, except for RGB-Flow-Audio. However, the TSDF-based approach achieved greater classification performance at the video-level, which shows the importance of segment weighing models.

The other issue that should be highlighted is the imbalance of the EOAD dataset. We ensured data balance for segment fusion models by selecting the same amount of samples from the selected and other classes. However, data imbalance may cause stream fusion models (deep-feature based and VAEs regressors) to overfit the target weights of major classes (i.e., *soccer* has 99 training samples while *rafting* has only 4). To overcome this problem, EOAD may be populated with additional video clips, especially for activities with insufficient samples, such as *jet ski*, *horseback*, *go – kart*, *longboarding*, *rafting*, *rock climbing*, *scuba*, and *stair climbing*. Another option may be to re-sample EOAD activities using under-sampling and over-sampling techniques. However, under-sampling may cause data insufficiency while training, especially VAEs regressor. On the other hand, over-sampling needs more complicated procedures because of the complexity of generating synthetic data from the observation of minor classes (e.g., Synthetic Minority Over-sampling Technique [191]).

Alternatively, the architecture of the VAEs regressor may be reviewed for the EAR task. For example, considering the success of ViTs, an attention mechanism may be designed in front of the VAEs regressor instead of using typical feature fusion techniques. In addition, we know that a subset of segments is more critical to change the state of video-level decisions from *False* to *True*. A custom loss function may be defined to force the models to pay more attention to these segments.

On the other hand, one of the alternative approaches that can be applied to EAR tasks is Multi-Instance Learning (MIL). MIL is a weakly supervised algorithm that learns over training instances that are composed of *sets* (also called *bags*) including multiple samples with a single label. For MIL, classification is performed at two levels: bag and instance which is similar to the proposed framework’s segment and video levels. Although level-based classification is similar, one should carefully analyze the problem characteristics of MIL consisting of data distribution, the definition of task, ambiguity of instance labels, and the composition

of bags. For example, one study [192] referred to the problem of high *witness rate* (i.e., the high proportion of positive samples in positive bags) eliminating the need for MIL usage. Consequently, the EAR problem can be characterized as a typical supervised problem with one-sided noise.

Additionally, transformer architectures have been recently used for multi-modal tasks mostly under vision-language problems such as visual questioning answering (VQA) [193], cross-modal retrieval [194], or image captioning [195]. Multi-modal transformers are categorized as multi-stream transformers (utilizing independent Transformers for each modality concatenated by another Transformer to learn cross-modal representation) [196–198] and single-stream transformers (feeding multi-modal input to a single transformer) [199–202]. Additionally, one of the recent research [127] proposed a multi-stream transformer architecture (called a Video-Audio-Text Transformer (VATT)) that takes raw signals as inputs and extract multi-modal representations to solve a variety of downstream tasks such as video action recognition, audio event classification, image classification, and text-to-video retrieval. The proposed solution takes raw video (3D RGB voxels), audio, and text information. Additionally, Noise Contrastive Estimation (NCE) and Multiple Instance Learning NCE (MIL-NCE) are employed to align video-audio and video-text pairs. Although EOAD lacks text information and the proposed solution is still computationally intensive, VATT architecture can be used as an alternative approach.

CHAPTER 5

CONCLUSION AND FUTURE WORK

This research offers two novel multi-modal EAR frameworks, one of which employs MKL classification for feature selection and decision fusion, and the other utilizes a two-stage decision fusion pipeline. These two approaches have several differences, advantages, and limitations. The most notable difference between them is their fusion strategies. The MKL-based approach combines sensor information using its feature selection and classification ability. Contrary, the two-stage decision fusion technique learns the weights for stream and segment decisions explicitly using EAR model confidences of actual classes. Another difference is about adding new sensors to the proposed frameworks. One should retrain the MKL framework in case of adding a new modality. In addition, feature and kernel sets should be re-elected. On the other hand, the two-stage decision fusion technique requires training of the EAR model for the new modality as well as training of stream and segment fusion models. The types of features are also different: the MKL-based solution receives hand-crafted features extracted for each video clip, and the other utilizes deep features acquired from segments after dividing video clips. Therefore, MKL-based framework features need expertise for different modalities for using hand-crafted features, whereas two-stage decision fusion does not expect intensive domain knowledge for feature extraction.

For the MKL-based framework, experiments have demonstrated that combining multiple modalities improves recognition performance. The suggested solution had a superior performance on three different egocentric datasets compared to state-of-the-art methods. Therefore, the results showed that employing MKL with multimodal characteristics is an effective strategy for EAR. Alternatively, the variation of the selected feature and kernel sets for MKL is closely related to the properties of egocentric datasets. Because the recording settings of videos (i.e., location, time, actors), accessible sensor information (i.e., visual, audio, sensor), and the dynamics of egocentric activities vary among datasets, it is impossible to establish a universal set of optimal features and kernels. To give a generic solution for recognizing egocentric actions, it is necessary to propose an adaptable approach that dynamically learns the changing conditions of datasets such as the one presented in this study.

Even if the MKL-based framework adaptively combines several modalities' features, it depends on handcrafted features. A growing number of research propose end-to-end solutions utilizing deep learning algorithms employing visual [2, 65, 88] and wearable sensor data [149, 150]. Therefore, the second part of the research focused on using deep models and deep features to fuse multi-modal information. For that purpose, we proposed a confidence-based EAR system with a two-stage decision fusion pipeline that utilizes deep features for

decision fusion models. At first, the stream decisions produced by EAR models are fused according to their decision weights. After that, the fused segment decisions are also weighted in the second stage. The decision weights are determined based on their relative significance to the selected activities. In this work, the significance of decisions is associated with confidence collected from pre-trained EAR models. Normalized confidence values are used as weights for stream decisions, and their sum is used for segment decisions. The experiments for the ideal case using confidences as weights validated the proposed approach and boosted the segment and video-level classification performances.

Adaptive stream weighting is intended to resolve two primary issues during fusion. One of them is to lessen the impact of inputs with poor quality (i.e., low light conditions or saturated frames). The second one is to prefer the modalities with the most information regarding the associated activity. The results demonstrated that it is hard to estimate the weights of a stream independent from its class, as the relevance of information may be more significant for one activity while it may be of lesser importance for another. Even if segment weighting models did not approach the practical limits of the ideal case, they have a considerable favorable impact on the results at the video level. On the other hand, we demonstrated that class-aware model-based segment weights considerably improve the classification accuracy at the video level. This allows the segments to be efficiently weighted based on the information they provide for the associated class. In addition, the proposed regression model architecture can learn this information and correlate it with segment weights. Finally, stream and segment models were concatenated to realize their effects on the results. However, the video-level precision did not alter or have limited improvements in the results.

We know that the class-aware segment fusion models need more accurate segment classification results to reach their potential which was also proved by experiments in this work. Therefore, increasing segment classification performance is crucial to effectively use the proposed two-stage decision fusion framework. One of the future works may be to add an attention mechanism in front of the VAEs regressor instead of using standard feature fusion techniques to increase segment classification accuracy.

30 different egocentric activities have been defined in EOAD. Due to this, it is not possible to recognize any other activities using the proposed frameworks. However, an additional binary classifier may be trained to discriminate the samples according to whether or not they belong to the pre-defined activity set.

Another alternative approach may be to use multi-instance learning (MIL) for egocentric activity recognition. The definition of bag and instance in MIL is similar to segment and video-level classification. However, MIL-based solutions should be carefully designed to take advantage of them. Otherwise, it may not achieve satisfactory results compared to typical supervised learning techniques.

MIL is a weakly supervised algorithm that learns over training instances that are composed of *sets* (also called *bags*) including multiple samples with a single label. For MIL, classification is performed at two levels: bag and instance which is similar to the proposed framework's segment and video levels. Although level-based classification is similar, one should carefully analyze the problem characteristics of MIL consisting of data distribution, the definition of task, ambiguity of instance labels, and the composition of bags. For example, one study [192]

referred to the problem of high *witness rate* (i.e., the high proportion of positive samples in positive bags) eliminating the need for MIL usage. Consequently, the problem can be characterized as a typical supervised problem with one-sided noise.

Even though the audio stream or wearable sensors have restricted single-stream performance levels, it is interoperable with other modalities. Therefore, it is evident that utilizing several modalities from various sensors improves EAR performance. For example, it was demonstrated in ideal case experiments for a two-stage decision fusion approach that it is possible to achieve satisfactory results by combining a low-precision stream (i.e., Audio) with a high-performance stream (i.e., RGB). However, combining visual data with audio or wearable sensors still requires additional EAR research. In contrast to third-person activity recognition, egocentric activity datasets may contain more information about activities thanks to a range of sensors that capture the event directly. To recognize users' activities, it is required to design new frameworks that can combine features from different domains effectively and practically.

REFERENCES

- [1] M. A. Arabacı, F. Özkan, E. Surer, P. Jančovič, and A. Temizel, “Multi-modal egocentric activity recognition using multi-kernel learning,” *Multimedia Tools and Applications*, vol. 80, no. 11, pp. 16299–16328, 2021.
- [2] H. Kwon, Y. Kim, J. S. Lee, and M. Cho, “First person action recognition via two-stream convnet with long-term fusion pooling,” *Pattern Recognition Letters*, vol. 112, pp. 161–167, 2018.
- [3] T. Duan, A. Anand, D. Y. Ding, K. K. Thai, S. Basu, A. Ng, and A. Schuler, “Ngboost: Natural gradient boosting for probabilistic prediction,” in *International Conference on Machine Learning*, pp. 2690–2700, PMLR, 2020.
- [4] J. Rocca, “Understanding variational autoencoders (vae),” 2019.
- [5] J. Pansiot, D. Stoyanov, D. McIlwraith, B. P. Lo, and G.-Z. Yang, “Ambient and wearable sensor fusion for activity recognition in healthcare monitoring systems,” in *4th international workshop on wearable and implantable body sensor networks (BSN 2007)*, pp. 208–212, Springer, 2007.
- [6] X. Wang, D. Rosenblum, and Y. Wang, “Context-aware mobile music recommendation for daily activities,” in *Proceedings of the 20th ACM international conference on Multimedia*, pp. 99–108, ACM, 2012.
- [7] F. Ozkan, M. A. Arabaci, E. Surer, and A. Temizel, “Boosted multiple kernel learning for first-person activity recognition,” in *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1050–1054, IEEE, 2017.
- [8] M. S. Ryoo and L. Matthies, “First-person activity recognition: What are they doing to me?,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2730–2737, 2013.
- [9] M. Liu, L. Ma, K. Somasundaram, Y. Li, K. Grauman, J. M. Rehg, and C. Li, “Egocentric activity recognition and localization on a 3d map,” in *European Conference on Computer Vision*, pp. 621–638, Springer, 2022.
- [10] V. Escorcia, R. Guerrero, X. Zhu, and B. Martinez, “Sos! self-supervised learning over sets of handled objects in egocentric action recognition,” *arXiv preprint arXiv:2204.04796*, 2022.
- [11] C. Plizzari, M. Planamente, G. Goletto, M. Cannici, E. Gusso, M. Matteucci, and B. Caputo, “E2 (go) motion: Motion augmented event stream for egocentric action recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19935–19947, 2022.

- [12] E. Kazakos, J. Huh, A. Nagrani, A. Zisserman, and D. Damen, “With a little help from my temporal context: Multimodal egocentric action recognition,” *arXiv preprint arXiv:2111.01024*, 2021.
- [13] W. Jia, M. Liu, and J. M. Rehg, “Generative adversarial network for future hand segmentation from egocentric video,” *arXiv preprint arXiv:2203.11305*, 2022.
- [14] M. Cai, F. Lu, and Y. Sato, “Generalizing hand segmentation in egocentric videos with uncertainty-guided model adaptation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14392–14401, 2020.
- [15] W. Yi and D. Ballard, “Recognizing behavior in hand-eye coordination patterns,” *International Journal of Humanoid Robotics*, vol. 6, no. 03, pp. 337–359, 2009.
- [16] Y. Wu, L. Zhu, X. Wang, Y. Yang, and F. Wu, “Learning to anticipate egocentric actions by imagination,” *IEEE Transactions on Image Processing*, vol. 30, pp. 1143–1152, 2020.
- [17] Y. Li, A. Fathi, and J. M. Rehg, “Learning to predict gaze in egocentric video,” in *Proceedings of the IEEE international conference on computer vision*, pp. 3216–3223, 2013.
- [18] Y. J. Lee, J. Ghosh, and K. Grauman, “Discovering important people and objects for egocentric video summarization,” in *2012 IEEE conference on computer vision and pattern recognition*, pp. 1346–1353, IEEE, 2012.
- [19] M. Bolanos, M. Dimiccoli, and P. Radeva, “Toward storytelling from visual lifelogging: An overview,” *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 1, pp. 77–90, 2016.
- [20] H. Li, Y. Cai, and W.-S. Zheng, “Deep dual relation modeling for egocentric interaction recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7932–7941, 2019.
- [21] R. Yonetani, K. M. Kitani, and Y. Sato, “Recognizing micro-actions and reactions from paired egocentric videos,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2629–2638, 2016.
- [22] T. Yagi, M. T. Hasan, and Y. Sato, “Hand-object contact prediction via motion-based pseudo-labeling and guided progressive label correction,” *arXiv preprint arXiv:2110.10174*, 2021.
- [23] D. Damen, T. Leelasawassuk, O. Haines, A. Calway, and W. W. Mayol-Cuevas, “You-do, i-learn: Discovering task relevant objects and their modes of interaction from multi-user egocentric video.,” in *BMVC*, vol. 2, p. 3, 2014.
- [24] S. Zhang, Q. Ma, Y. Zhang, Z. Qian, T. Kwon, M. Pollefeys, F. Bogo, and S. Tang, “Egobody: Human body shape and motion of interacting people from head-mounted devices,” in *European Conference on Computer Vision*, pp. 180–200, Springer, 2022.
- [25] J. Wang, L. Liu, W. Xu, K. Sarkar, and C. Theobalt, “Estimating egocentric 3d human pose in global space,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11500–11509, 2021.

- [26] E. Ng, D. Xiang, H. Joo, and K. Grauman, “You2me: Inferring body pose in egocentric video via first and second person interactions,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9890–9900, 2020.
- [27] D. Thapar, C. Arora, and A. Nigam, “Is sharing of egocentric video giving away your biometric signature?,” in *European Conference on Computer Vision*, pp. 399–416, Springer, 2020.
- [28] M. S. Ryoo, B. Rothrock, C. Fleming, and H. J. Yang, “Privacy-preserving human activity recognition from extreme low resolution,” in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [29] Y. Lu, Y. Wei, L. Liu, J. Zhong, L. Sun, and Y. Liu, “Towards unsupervised physical activity recognition using smartphone accelerometers,” *Multimedia Tools and Applications*, vol. 76, no. 8, pp. 10701–10719, 2017.
- [30] G. Abebe, A. Cavallaro, and X. Parra, “Robust multi-dimensional motion features for first-person vision activity recognition,” *Computer Vision and Image Understanding*, vol. 149, pp. 229–248, 2016.
- [31] S. Song, V. Chandrasekhar, N.-M. Cheung, S. Narayan, L. Li, and J.-H. Lim, “Activity recognition in egocentric life-logging videos,” in *Computer Vision - ACCV 2014 Workshops* (C. V. Jawahar and S. Shan, eds.), (Cham), pp. 445–458, Springer International Publishing, 2015.
- [32] F. J. Ordóñez and D. Roggen, “Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition,” *Sensors*, vol. 16, no. 1, 2016.
- [33] S. Bhattacharya and N. D. Lane, “From smart to deep: Robust activity recognition on smartwatches using deep learning,” in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pp. 1–6, March 2016.
- [34] Y. Li, Z. Ye, and J. M. Rehg, “Delving into egocentric actions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 287–295, 2015.
- [35] H. Wang, A. Kläser, C. Schmid, and L. Cheng-Lin, “Action recognition by dense trajectories,” in *CVPR 2011-IEEE Conference on Computer Vision & Pattern Recognition*, pp. 3169–3176, IEEE, 2011.
- [36] S. Song, N.-M. Cheung, V. Chandrasekhar, B. Mandal, and J. Liri, “Egocentric activity recognition with multimodal fisher vector,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2717–2721, IEEE, 2016.
- [37] H. F. Nweke, Y. W. Teh, M. A. Al-Garadi, and U. R. Alo, “Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges,” *Expert Systems with Applications*, vol. 105, pp. 233–261, 2018.
- [38] Y. Iwashita, A. Takamine, R. Kurazume, and M. S. Ryoo, “First-person animal activity recognition from egocentric videos,” in *2014 22nd International Conference on Pattern Recognition*, pp. 4310–4315, IEEE, 2014.

- [39] Y. Liu, L. Nie, L. Han, L. Zhang, and D. S. Rosenblum, “Action2activity: recognizing complex activities from sensor data,” in *Twenty-fourth international joint conference on artificial intelligence*, 2015.
- [40] Y. Liu, L. Nie, L. Liu, and D. S. Rosenblum, “From action to activity: sensor-based activity recognition,” *Neurocomputing*, vol. 181, pp. 108–115, 2016.
- [41] L. Liu, L. Cheng, Y. Liu, Y. Jia, and D. S. Rosenblum, “Recognizing complex activities by a probabilistic interval-based model,” in *Thirtieth AAAI conference on artificial intelligence*, 2016.
- [42] S. Mann, “Wearable computing: A first step toward personal imaging,” *Computer*, vol. 30, no. 2, pp. 25–32, 1997.
- [43] T. Starner, B. Schiele, and A. Pentland, “Visual contextual awareness in wearable computing,” in *Digest of Papers. Second International Symposium on Wearable Computers (Cat. No. 98EX215)*, pp. 50–57, IEEE, 1998.
- [44] M. F. Land and M. Hayhoe, “In what ways do eye movements contribute to everyday activities?,” *Vision research*, vol. 41, no. 25–26, pp. 3559–3565, 2001.
- [45] K. K. Singh, K. Fatahalian, and A. A. Efros, “Krishnacam: Using a longitudinal, single-person, egocentric dataset for scene understanding tasks,” in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–9, IEEE, 2016.
- [46] T. Do, K. Vuong, and H. S. Park, “Egocentric scene understanding via multimodal spatial rectifier,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2832–2841, 2022.
- [47] Y. Huang, M. Cai, Z. Li, and Y. Sato, “Predicting gaze in egocentric video by learning task-dependent attention transition,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 754–769, 2018.
- [48] H. S. Koppula and A. Saxena, “Anticipating human activities using object affordances for reactive robotic response,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 14–29, 2015.
- [49] J. Gao, Z. Yang, and R. Nevatia, “Red: Reinforced encoder-decoder networks for action anticipation,” *arXiv preprint arXiv:1707.04818*, 2017.
- [50] Y. Abu Farha, A. Richard, and J. Gall, “When will you do what?-anticipating temporal occurrences of activities,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5343–5352, 2018.
- [51] A. Betancourt, P. Morerio, C. S. Regazzoni, and M. Rauterberg, “The evolution of first person vision methods: A survey,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 5, pp. 744–760, 2015.
- [52] A. Fathi, J. K. Hodgins, and J. M. Rehg, “Social interactions: A first-person perspective,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1226–1233, IEEE, 2012.

- [53] Z. Lu and K. Grauman, “Story-driven summarization for egocentric video,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2714–2721, 2013.
- [54] A. L. Yarbus, “Eye movements during perception of complex objects,” in *Eye movements and vision*, pp. 171–211, Springer, 1967.
- [55] T. Kanade and M. Hebert, “First-person vision,” *Proceedings of the IEEE*, vol. 100, no. 8, pp. 2442–2453, 2012.
- [56] J. K. Aggarwal and M. S. Ryoo, “Human activity analysis: A review,” *Acm Computing Surveys (Csur)*, vol. 43, no. 3, pp. 1–43, 2011.
- [57] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, “Learning realistic human actions from movies,” in *CVPR 2008-IEEE Conference on Computer Vision & Pattern Recognition*, pp. 1–8, IEEE Computer Society, 2008.
- [58] X. Peng, L. Wang, X. Wang, and Y. Qiao, “Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice,” *Computer Vision and Image Understanding*, vol. 150, pp. 109–125, 2016.
- [59] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, “Dense trajectories and motion boundary descriptors for action recognition,” *International journal of computer vision*, vol. 103, no. 1, pp. 60–79, 2013.
- [60] A. Fathi, A. Farhadi, and J. M. Rehg, “Understanding egocentric activities,” in *2011 international conference on computer vision*, pp. 407–414, IEEE, 2011.
- [61] A. Behera, D. C. Hogg, and A. G. Cohn, “Egocentric activity monitoring and recovery,” in *Asian Conference on Computer Vision*, pp. 519–532, Springer, 2012.
- [62] C.-Y. Ma, A. Kadav, I. Melvin, Z. Kira, G. AlRegib, and H. P. Graf, “Attend and interact: Higher-order object interactions for video understanding,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6790–6800, 2018.
- [63] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, “Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1949–1957, 2015.
- [64] 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 *International Conference Image Analysis and Recognition*, pp. 390–398, Springer, 2018.
- [65] Y. Poleg, A. Ephrat, S. Peleg, and C. Arora, “Compact cnn for indexing egocentric videos,” in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–9, IEEE, 2016.
- [66] S. Narayan, M. S. Kankanhalli, and K. R. Ramakrishnan, “Action and interaction recognition in first-person videos,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 512–518, 2014.

- [67] Y. Hoshen and S. Peleg, “Egocentric video biometrics,” *CoRR*, *abs/1411.7591*, vol. 1, no. 3, p. 4, 2014.
- [68] B. Clarkson and A. Pentland, “Unsupervised clustering of ambulatory audio and video,” in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, vol. 6, pp. 3037–3040, IEEE, 1999.
- [69] B. Clarkson, K. Mase, and A. Pentland, “Recognizing user context via wearable sensors,” in *Digest of papers. Fourth international symposium on wearable computers*, pp. 69–75, IEEE, 2000.
- [70] D. J. Patterson, D. Fox, H. Kautz, and M. Philipose, “Fine-grained activity recognition by aggregating abstract object usage,” in *Ninth IEEE International Symposium on Wearable Computers (ISWC’05)*, pp. 44–51, IEEE, 2005.
- [71] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, and D. Hahnel, “Inferring activities from interactions with objects,” *IEEE pervasive computing*, vol. 3, no. 4, pp. 50–57, 2004.
- [72] J. Wu, A. Osuntogun, T. Choudhury, M. Philipose, and J. M. Rehg, “A scalable approach to activity recognition based on object use,” in *2007 IEEE 11th international conference on computer vision*, pp. 1–8, IEEE, 2007.
- [73] N. Krahnstoever, J. Rittscher, P. Tu, K. Chean, and T. Tomlinson, “Activity recognition using visual tracking and rfid,” in *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION’05)-Volume 1*, vol. 1, pp. 494–500, IEEE, 2005.
- [74] K. Van Laerhoven and H.-W. Gellersen, “Spine versus porcupine: A study in distributed wearable activity recognition,” in *Eighth International Symposium on Wearable Computers*, vol. 1, pp. 142–149, IEEE, 2004.
- [75] O. Durmaz Incel, “Analysis of movement, orientation and rotation-based sensing for phone placement recognition,” *Sensors*, vol. 15, no. 10, pp. 25474–25506, 2015.
- [76] T. Yilmaz, R. Foster, and Y. Hao, “Detecting vital signs with wearable wireless sensors,” *Sensors*, vol. 10, no. 12, pp. 10837–10862, 2010.
- [77] K. Guo, P. Ishwar, and J. Konrad, “Action recognition from video using feature covariance matrices,” *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2479–2494, 2013.
- [78] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.
- [79] E. Kazakos, A. Nagrani, A. Zisserman, and D. Damen, “Epic-fusion: Audio-visual temporal binding for egocentric action recognition,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5492–5501, 2019.
- [80] G. Abebe and A. Cavallaro, “Hierarchical modeling for first-person vision activity recognition,” *Neurocomputing*, vol. 267, pp. 362–377, 2017.

- [81] S. Sudhakaran and O. Lanz, “Convolutional long short-term memory networks for recognizing first person interactions,” *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 2339–2346, 2017.
- [82] X. Wang, L. Gao, J. Song, X. Zhen, N. Sebe, and H. T. Shen, “Deep appearance and motion learning for egocentric activity recognition,” *Neurocomputing*, vol. 275, pp. 438–447, 2018.
- [83] H. Song, J. J. Thiagarajan, P. Sattigeri, K. N. Ramamurthy, and A. Spanias, “A deep learning approach to multiple kernel fusion,” in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 2292–2296, IEEE, 2017.
- [84] C. Yu, S. Bambach, Z. Zhang, and D. J. Crandall, “Exploring inter-observer differences in first-person object views using deep learning models,” in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 2773–2782, Oct 2017.
- [85] S. Singh, C. Arora, and C. Jawahar, “First person action recognition using deep learned descriptors,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2620–2628, 2016.
- [86] S. Bambach, Z. Zhang, D. J. Crandall, and C. Yu, “Exploring inter-observer differences in first-person object views using deep learning models,” in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 2773–2782, 2017.
- [87] Y. Poleg, C. Arora, and S. Peleg, “Temporal segmentation of egocentric videos,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2537–2544, June 2014.
- [88] M. Ma, H. Fan, and K. M. Kitani, “Going deeper into first-person activity recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1894–1903, June 2016.
- [89] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2625–2634, 2015.
- [90] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, “Describing videos by exploiting temporal structure,” in *Proceedings of the IEEE international conference on computer vision*, pp. 4507–4515, 2015.
- [91] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, “Sequence to sequence-video to text,” in *Proceedings of the IEEE international conference on computer vision*, pp. 4534–4542, 2015.
- [92] N. Y. Hammerla, S. Halloran, and T. Plötz, “Deep, convolutional, and recurrent models for human activity recognition using wearables,” *arXiv preprint arXiv:1604.08880*, 2016.
- [93] Y. Shen, B. Ni, Z. Li, and N. Zhuang, “Egocentric activity prediction via event modulated attention,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 197–212, 2018.

- [94] C. Chen, R. Jafari, and N. Kehtarnavaz, “Improving human action recognition using fusion of depth camera and inertial sensors,” *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 1, pp. 51–61, chen2014improving.
- [95] B. Ni, Y. Pei, P. Moulin, and S. Yan, “Multilevel depth and image fusion for human activity detection,” *IEEE transactions on cybernetics*, vol. 43, no. 5, pp. 1383–1394, 2013.
- [96] D. Avola, M. Bernardi, and G. L. Foresti, “Fusing depth and colour information for human action recognition,” *Multimedia Tools and Applications*, vol. 78, no. 5, pp. 5919–5939, 2019.
- [97] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, “Berkeley mhad: A comprehensive multimodal human action database,” in *2013 IEEE Workshop on Applications of Computer Vision (WACV)*, pp. 53–60, IEEE, 2013.
- [98] X. Wang, Y. Wu, L. Zhu, and Y. Yang, “Symbiotic attention with privileged information for egocentric action recognition,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 12249–12256, 2020.
- [99] Y. Tang, Z. Wang, J. Lu, J. Feng, and J. Zhou, “Multi-stream deep neural networks for rgb-d egocentric action recognition,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 10, pp. 3001–3015, 2018.
- [100] A. Furnari and G. M. Farinella, “What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6252–6261, 2019.
- [101] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, “Bottom-up and top-down attention for image captioning and visual question answering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6077–6086, 2018.
- [102] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in *International conference on machine learning*, pp. 2048–2057, PMLR, 2015.
- [103] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [104] A. Graves, “Generating sequences with recurrent neural networks,” *arXiv preprint arXiv:1308.0850*, 2013.
- [105] J. Ba, V. Mnih, and K. Kavukcuoglu, “Multiple object recognition with visual attention,” *arXiv preprint arXiv:1412.7755*, 2014.
- [106] V. Mnih, N. Heess, A. Graves, *et al.*, “Recurrent models of visual attention,” *Advances in neural information processing systems*, vol. 27, 2014.
- [107] D.-K. Nguyen and T. Okatani, “Improved fusion of visual and language representations by dense symmetric co-attention for visual question answering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6087–6096, 2018.

- [108] H. Xu and K. Saenko, “Ask, attend and answer: Exploring question-guided spatial attention for visual question answering,” in *European conference on computer vision*, pp. 451–466, Springer, 2016.
- [109] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola, “Stacked attention networks for image question answering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 21–29, 2016.
- [110] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher, “Ask me anything: Dynamic memory networks for natural language processing,” in *International conference on machine learning*, pp. 1378–1387, PMLR, 2016.
- [111] Y. Li, M. Liu, and J. M. Rehg, “In the eye of beholder: Joint learning of gaze and actions in first person video,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 619–635, 2018.
- [112] S. Sharma, R. Kiros, and R. Salakhutdinov, “Action recognition using visual attention,” *arXiv preprint arXiv:1511.04119*, 2015.
- [113] W. Du, Y. Wang, and Y. Qiao, “Recurrent spatial-temporal attention network for action recognition in videos,” *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1347–1360, 2017.
- [114] J. Liu, G. Wang, L.-Y. Duan, K. Abdiyeva, and A. C. Kot, “Skeleton-based human action recognition with global context-aware attention lstm networks,” *IEEE Transactions on Image Processing*, vol. 27, no. 4, pp. 1586–1599, 2017.
- [115] K. Matsuo, K. Yamada, S. Ueno, and S. Naito, “An attention-based activity recognition for egocentric video,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 551–556, 2014.
- [116] M. Lu, Z.-N. Li, Y. Wang, and G. Pan, “Deep attention network for egocentric action recognition,” *IEEE Transactions on Image Processing*, vol. 28, no. 8, pp. 3703–3713, 2019.
- [117] A. Borji and L. Itti, “State-of-the-art in visual attention modeling,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 185–207, 2012.
- [118] F. Katsuki and C. Constantinidis, “Bottom-up and top-down attention: different processes and overlapping neural systems,” *The Neuroscientist*, vol. 20, no. 5, pp. 509–521, 2014.
- [119] S. Sudhakaran, S. Escalera, and O. Lanz, “Lsta: Long short-term attention for egocentric action recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9954–9963, 2019.
- [120] A. Piergiovanni, C. Fan, and M. Ryoo, “Learning latent subevents in activity videos using temporal attention filters,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

- [121] Z. Li, K. Gavriluyk, E. Gavves, M. Jain, and C. G. Snoek, “Videolstm convolves, attends and flows for action recognition,” *Computer Vision and Image Understanding*, vol. 166, pp. 41–50, 2018.
- [122] H. Zhao, J. Jia, and V. Koltun, “Exploring self-attention for image recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10076–10085, 2020.
- [123] H. Hu, Z. Zhang, Z. Xie, and S. Lin, “Local relation networks for image recognition,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3464–3473, 2019.
- [124] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, “Stand-alone self-attention in vision models,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [125] G. Bertasius, H. Wang, and L. Torresani, “Is space-time attention all you need for video understanding?,” in *ICML*, vol. 2, p. 4, 2021.
- [126] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, “Vivit: A video vision transformer,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6836–6846, 2021.
- [127] H. Akbari, L. Yuan, R. Qian, W.-H. Chuang, S.-F. Chang, Y. Cui, and B. Gong, “Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 24206–24221, 2021.
- [128] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [129] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.
- [130] H. Xia and S. C. Hoi, “Mkboost: A framework of multiple kernel boosting,” *IEEE Transactions on knowledge and data engineering*, vol. 25, no. 7, pp. 1574–1586, 2012.
- [131] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet, “Simplemkl,” *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2491–2521, 2008.
- [132] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, “Behavior recognition via sparse spatio-temporal features,” in *2005 IEEE international workshop on visual surveillance and performance evaluation of tracking and surveillance*, pp. 65–72, IEEE, 2005.
- [133] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, “Log-euclidean metrics for fast and simple calculus on diffusion tensors,” *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 56, no. 2, pp. 411–421, 2006.

- [134] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.
- [135] S. Safavi, M. Russell, and P. Jančovič, "Automatic speaker, age-group and gender identification from children's speech," *Computer Speech & Language*, vol. 50, pp. 141–156, 2018.
- [136] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, *et al.*, "The htk book (v3. 4)," *Cambridge University*, 2006.
- [137] J. . Gauvain and Chin-Hui Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, pp. 291–298, April 1994.
- [138] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff, "Svm based speaker verification using a gmm supervector kernel and nap variability compensation," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 1, pp. I–I, May 2006.
- [139] B. Ni, C. D. Nguyen, and P. Moulin, "Rgb-d-camera based get-up event detection for hospital fall prevention," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1405–1408, IEEE, 2012.
- [140] T. Gärtner, "A survey of kernels for structured data," *ACM SIGKDD Explorations Newsletter*, vol. 5, no. 1, pp. 49–58, 2003.
- [141] B. Schölkopf, A. J. Smola, F. Bach, *et al.*, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.
- [142] G. R. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and W. S. Noble, "A statistical framework for genomic data fusion," *Bioinformatics*, vol. 20, no. 16, pp. 2626–2635, 2004.
- [143] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *Journal-Japanese Society For Artificial Intelligence*, vol. 14, no. 771-780, p. 1612, 1999.
- [144] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of machine learning research*, vol. 12, no. Jul, pp. 2211–2268, 2011.
- [145] R. Fitzgerald and B. Lees, "Assessing the classification accuracy of multisource remote sensing data," *Remote sensing of Environment*, vol. 47, no. 3, pp. 362–368, 1994.
- [146] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.
- [147] L. Bottou and C.-J. Lin, "Support vector machine solvers," *Large scale kernel machines*, vol. 3, no. 1, pp. 301–320, 2007.
- [148] X. Li, L. Wang, and E. Sung, "Improving adaboost for classification on small training sample sets with active learning," in *Proceedings of Asian Conference on Computer Vision (ACCV)*, pp. 1–6, 2004.

- [149] A. Bulling, U. Blanke, and B. Schiele, “A tutorial on human activity recognition using body-worn inertial sensors,” *ACM Computing Surveys (CSUR)*, vol. 46, no. 3, p. 33, 2014.
- [150] J. Morales and D. Akopian, “Physical activity recognition by smartphones, a survey,” *Biocybernetics and Biomedical Engineering*, vol. 37, no. 3, pp. 388–400, 2017.
- [151] B. Hidalgo and M. Goodman, “Multivariate or multivariable regression?,” *American journal of public health*, vol. 103, no. 1, pp. 39–40, 2013.
- [152] C. Feichtenhofer, A. Pinz, and A. Zisserman, “Convolutional two-stream network fusion for video action recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1933–1941, 2016.
- [153] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.
- [154] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [155] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.
- [156] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.
- [157] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” *arXiv preprint arXiv:1312.6229*, 2013.
- [158] S. Lathuilière, P. Mesejo, X. Alameda-Pineda, and R. Horaud, “A comprehensive analysis of deep regression,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 9, pp. 2065–2081, 2019.
- [159] A. Benou, R. Veksler, A. Friedman, and T. Riklin Raviv, “De-noising of contrast-enhanced mri sequences by an ensemble of expert deep neural networks,” in *Deep learning and data labeling for medical applications*, pp. 95–110, Springer, 2016.
- [160] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, “Deep autoencoding models for unsupervised anomaly segmentation in brain mr images,” in *International MICCAI brainlesion workshop*, pp. 161–169, Springer, 2018.
- [161] Q. Zhao, E. Adeli, N. Honnorat, T. Leng, and K. M. Pohl, “Variational autoencoder for regression: Application to brain aging analysis,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 823–831, Springer, 2019.

- [162] Y. Yoo, S. Yun, H. Jin Chang, Y. Demiris, and J. Young Choi, “Variational autoencoded regression: high dimensional regression of visual data on complex manifold,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3674–3683, 2017.
- [163] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, “beta-vae: Learning basic visual concepts with a constrained variational framework,” 2016.
- [164] H. Kim and A. Mnih, “Disentangling by factorising,” in *International Conference on Machine Learning*, pp. 2649–2658, PMLR, 2018.
- [165] J. T. Barron, “A general and adaptive robust loss function,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4331–4339, 2019.
- [166] H.-I. Ho, W.-C. Chiu, and Y.-C. F. Wang, “Summarizing first-person videos from third persons’ points of view,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 70–85, 2018.
- [167] K. M. Kitani, T. Okabe, Y. Sato, and A. Sugimoto, “Fast unsupervised ego-action learning for first-person sports videos,” in *CVPR 2011*, pp. 3241–3248, IEEE, 2011.
- [168] H. Pirsiavash and D. Ramanan, “Detecting activities of daily living in first-person camera views,” in *2012 IEEE conference on computer vision and pattern recognition*, pp. 2847–2854, IEEE, 2012.
- [169] A. Fathi, Y. Li, and J. M. Rehg, “Learning to recognize daily actions using gaze,” in *European Conference on Computer Vision*, pp. 314–327, Springer, 2012.
- [170] S. Song, V. Chandrasekhar, N.-M. Cheung, S. Narayan, L. Li, and J.-H. Lim, “Activity recognition in egocentric life-logging videos,” in *Asian conference on computer vision*, pp. 445–458, Springer, 2014.
- [171] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim, “First-person hand action benchmark with rgb-d videos and 3d hand pose annotations,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 409–419, 2018.
- [172] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, *et al.*, “Scaling egocentric vision: The epic-kitchens dataset,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 720–736, 2018.
- [173] 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,” *arXiv preprint arXiv:1804.09626*, 2018.
- [174] Y. Jang, B. Sullivan, C. Ludwig, I. Gilchrist, D. Damen, and W. Mayol-Cuevas, “Epic-tent: An egocentric video dataset for camping tent assembly,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

- [175] D. Damen, H. Doughty, G. M. Farinella, A. Furnari, E. Kazakos, J. Ma, D. Moltisanti, J. Munro, T. Perrett, W. Price, *et al.*, “Rescaling egocentric vision,” *arXiv preprint arXiv:2006.13256*, 2020.
- [176] T. Kwon, B. Tekin, J. Stühmer, F. Bogo, and M. Pollefeys, “H2o: Two hands manipulating objects for first person interaction recognition,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10138–10148, 2021.
- [177] S. Tomar, “Converting video formats with ffmpeg,” *Linux Journal*, vol. 2006, no. 146, p. 10, 2006.
- [178] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao, “Towards good practices for very deep two-stream convnets,” *arXiv preprint arXiv:1507.02159*, 2015.
- [179] C. Feichtenhofer, “Gpu based optical flow extraction in opencv.” https://github.com/feichtenhofer/gpu_flow, 2018.
- [180] C. Zach, T. Pock, and H. Bischof, “A duality based approach for realtime tv-l 1 optical flow,” in *Joint pattern recognition symposium*, pp. 214–223, Springer, 2007.
- [181] S. S. Stevens and J. Volkman, “The relation of pitch to frequency: A revised scale,” *The American Journal of Psychology*, vol. 53, no. 3, pp. 329–353, 1940.
- [182] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, “Cnn architectures for large-scale audio classification,” in *2017 IEEE international conference on acoustics, speech and signal processing (icassp)*, pp. 131–135, IEEE, 2017.
- [183] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, *et al.*, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4779–4783, IEEE, 2018.
- [184] H. Meng, T. Yan, F. Yuan, and H. Wei, “Speech emotion recognition from 3d log-mel spectrograms with deep learning network,” *IEEE access*, vol. 7, pp. 125868–125881, 2019.
- [185] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.
- [186] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell, “Compact bilinear pooling,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 317–326, 2016.
- [187] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [188] Q. Zhao, “Vae-for-regression.” <https://github.com/QingyuZhao/VAE-for-Regression>, 2020.

- [189] S. Raschka, “Mlxtend: Providing machine learning and data science utilities and extensions to python’s scientific computing stack,” *The Journal of Open Source Software*, vol. 3, Apr. 2018.
- [190] R. Johnson and T. Zhang, “Learning nonlinear functions using regularized greedy forest,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 5, pp. 942–954, 2013.
- [191] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.
- [192] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, “Multiple instance learning: A survey of problem characteristics and applications,” *Pattern Recognition*, vol. 77, pp. 329–353, 2018.
- [193] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, “Vqa: Visual question answering,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.
- [194] K.-H. Lee, X. Chen, G. Hua, H. Hu, and X. He, “Stacked cross attention for image-text matching,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 201–216, 2018.
- [195] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2015.
- [196] J. Lu, D. Batra, D. Parikh, and S. Lee, “Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks,” *Advances in neural information processing systems*, vol. 32, 2019.
- [197] H. Tan and M. Bansal, “Lxmert: Learning cross-modality encoder representations from transformers,” *arXiv preprint arXiv:1908.07490*, 2019.
- [198] S. Lee, Y. Yu, G. Kim, T. Breuel, J. Kautz, and Y. Song, “Parameter efficient multimodal transformers for video representation learning,” *arXiv preprint arXiv:2012.04124*, 2020.
- [199] Y.-C. Chen, L. Li, L. Yu, A. El Kholy, F. Ahmed, Z. Gan, Y. Cheng, and J. Liu, “Uniter: Universal image-text representation learning,” in *European conference on computer vision*, pp. 104–120, Springer, 2020.
- [200] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, “Visualbert: A simple and performant baseline for vision and language,” *arXiv preprint arXiv:1908.03557*, 2019.
- [201] X. Li, X. Yin, C. Li, P. Zhang, X. Hu, L. Zhang, L. Wang, H. Hu, L. Dong, F. Wei, *et al.*, “Oscar: Object-semantics aligned pre-training for vision-language tasks,” in *European Conference on Computer Vision*, pp. 121–137, Springer, 2020.
- [202] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, “Videobert: A joint model for video and language representation learning,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7464–7473, 2019.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Arabacı, Mehmet Ali
Nationality: Turkish (TC)
Date and Place of Birth: 02.09.1982, Aydın

EDUCATION

Degree	Institution	Year of Graduation
M.S.	Dokuz Eylül University Electrical and Electronics Engineering	2008
B.S.	Dokuz Eylül University Electrical and Electronics Engineering	2005
High School	Yavuz Selim High School	1999

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
2005-2009	Vestel Electronics	Senior Software Design Engineer
2010-Present	Tübitak Space Technologies Research Institute	Chief Researcher

PUBLICATIONS

Journal Publications

- M. A. Arabacı, F. Özkan, E. Surer, P. Jančovič, A. Temizel, "Multi-modal egocentric activity recognition using multi-kernel learning," *Multimedia Tools and Applications*, vol. 80, no. 11, 16299-16328, 2021.
- Soysal, M., et al. Multimodal concept detection in broadcast media: KavTan. *Multimedia tools and applications*, 2014, 72.3: 2787-2832.

International Conference Publications

- F. Özkan, M. A. Arabaci, E. Surer, A. Temizel, "Boosted multiple kernel learning for first-person activity recognition," in *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1050-1054. IEEE, 2017.
- Esen, E., Ozkan, S., Atil, I., Arabaci, M. A., & Tankiz, S. (2014). An image community detection method for hierarchical visualisation. In *Multimedia and Expo Workshops (ICMEW), 2014 IEEE International Conference on* (pp. 1-6). IEEE.
- Esen, E., Ozkan, S., Atil, I., Arabaci, M. A., & Tankiz, S. (2014). Detecting image communities. In *Content-Based Multimedia Indexing (CBMI), 2014 12th International Workshop on* (pp. 1-4). IEEE.
- Esen, E., Arabaci, M. A., & Soysal, M. (2013). Fight detection in surveillance videos. In *Content-Based Multimedia Indexing (CBMI), 2013 11th International Workshop on* (pp. 131-135). IEEE.