Action Recognition and Prediction with Applications to Medical Diagnosis and Daily Living

Bilge Soran

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2015

Reading Committee:

Linda Shapiro, Chair

Ali Farhadi

Jenq-Neng Hwang

Program Authorized to Offer Degree:
University of Washington
Computer Science and Engineering

University of Washington

Abstract

Action Recognition and Prediction with Applications to Medical Diagnosis and Daily Living

Bilge Soran

Chair of the Supervisory Committee:
Prof. Linda Shapiro
Computer Science and Engineering

The purpose of this research is to explore the possible ways of improving people's lives using information from static or egocentric (wearable) cameras. The usage of this information can be diagnostic or preventive. In a diagnostic case, we consider medical applications that will make the diagnosis procedure more objective, while disturbing the person minimally. An example scenario is detecting hand tremors of people suffering from Parkinson's disease or similar illnesses. This can be done with a static camera without having the patient wear any motion sensors or other tools. Our research showed that such a system can be built very reliably using only a static camera observing the patient.

With the recent technological developments, egocentric cameras have become a part of our lives. They are designed as tiny cameras that can be worn without disturbing the person. In order to investigate the possible information gain from egocentric cameras, we used a multiple camera setting, where both an egocentric and multiple static cameras exist. Our research showed that when fused correctly, using the information from different types of cameras increases the recognition accuracy of actions. The model we proposed for this task is also suitable for other multi-modal settings. To prove the generality of the proposed model we also tested it on a setting with multiple static cameras and showed state-of-the-art results. Our model learns the importance of each camera in recognizing the actions, and it can also be used to direct the scenes automatically. We created examples of automatically directed scenes to show the concept.

We also addressed the problem of improving people's lives in a preventive way using egocentric cameras. In our work preventive refers to the general notion of reminders that can possibly prevent people from making mistakes that can cause problems. For example, when people are leaving a room while the stove is on, they might be reminded to turn the stove off. We proposed a notification decision mechanism that reasons about interdependencies between actions, checks at every time step whether there is a missing action that should be completed before the ongoing one ends, calculates a cost for missing it, and uses this cost to make a notification decision. Such a notification system requires recognizing the past actions and predicting the online action while segmenting the activity observed so far. For this purpose, we proposed a model that uses standard features and accomplishes these three tasks successfully. We showed promising results on the extremely challenging task of issuing correct and timely reminders on a new egocentric dataset.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to express deep and sincere appreciation to my adviser and my mentor, Linda G. Shapiro, who always believed in and supported me during my studies. Your encouragement and guidance made this PhD study possible. I learned from you that what makes us stronger is creating a positive impact on others' lives even when they are strangers. I truly admire and value everything I have learned from you. Thanks for being my adviser. Thanks for everything.

I am deeply grateful to Ali Farhadi for being my co-adviser, and for all the great ideas he shared with me during my research. I learned a new world from you, which is competitive, fun, depressing and joyful at the same time. I couldn't step into this world without your insight. Thanks for everything.

I am also very grateful to Steve Seitz and Magdalena Balazinska, who helped me start this journey in the first place. I wish to express my heartfelt appreciation to Fatih Erdogan Sevilgen, Yusuf Sinan Akgul and Ercan Oztemel for being my teachers, my mentors and my friends besides being great inspirations for me.

I also would like to express special thanks to Jenq-Neng Hwang and Su-In Lee for giving me the opportunity to discuss my research, and for their insightful remarks. A big thanks to Mark Ganter, who served as a GSR in my exams. I feel very lucky to be surrounded by a group of nice people in the Multimedia Lab. Thank you Natalia Larios, Jiun-Hung Chen, Indriyati Atmosukarto, Shulin Yang, Sara Rolfe, Jia Wu, Deepali Aneja, Yao Lu, Ravensara Travillian, Alfred Gui, Mabel Raza, Lauren Thorngate, Shu Liang, Jinna Lei and Irma W. Lam for your friendship during all those memorable years. I am also very lucky to have awesome office mates; thank you Robert Gens and Michael Jae Yoon Chung. I can't imagine being in a better office, and I appreciate your friendship and sincerity for all these years. I'd like to thank all of the CSE community, for making

this department such an exceptional place to pursue a PhD.

My special thanks to Ezgi Mercan, for being a sister to me more than being a friend. Thanks for always being there when I needed you the most. The most memorable part of my PhD is full of the moments we shared. The coffee is always good with you, even when the beans are over roasted. Thanks to Kivanc Muslu and Safiye Celik, and to all my friends for all the precious times we shared, which made this journey fun and memorable.

I want to express my deepest gratitude to my mother, Zeynep Basakcioglu, my greatest support in the world. Whatever I achieve is because of you. You taught me to see the color in the gray. You taught me to never ever give up. You taught me to dream and work for it. You taught me to be myself and be proud of it. I owe my every achievement to you. Thanks for being my mother.

I wish to thank the love of my life, Muhammed Serdar Soran. Thank you for all the moments you held my hand, for all the moments you laughed with me, for all the moments you erased my tears during this journey. I always know that our hearts are very old friends that enjoy the same sunrise. I am also very grateful to my daughter, Elif Melek Soran, for enlightening not only my every day but also my soul at every day. This PhD is more meaningful after having you.

Above all, I am humbled in front of the God, Allah, for this life and everything in it. Thanks God for all the blessings, opportunities and experiences that are destined to me. I trust you for everything I need.

DEDICATION

to my daughter, Elif Melek

Never lose faith, never lose hope, never ever give up.

Wherever you are, whenever you need it, my love will always be with you.

Most of the important things in the world have been accomplished by people

who have kept on trying when there seemed to be no hope at all.

Dale Carnegie

Chapter 1

INTRODUCTION

The goal of human activity recognition is to automatically analyze ongoing activities from an unknown video or a sequence of image frames [2]. There is a general confusion about what activities and actions are. In much of the literature actions represent simple motion patterns usually executed by one person like walking, sitting, swimming, and activities represent more complex movements which are usually a sequence of actions and may include more than one person [85][2].

Computer vision and pattern recognition techniques, involving feature extraction, object detection, clustering, and classification, have been successfully used for many action recognition systems. Image processing techniques such as analysis and detection of shape, texture, color, motion, optical flow, image enhancement, segmentation, and contour modeling, have also been found to be effective. Connectionist approaches, involving radial basis function networks, time-delay neural networks, and multilayer perceptrons, have been utilized in action recognition as well. While static gesture (pose) recognition can typically be accomplished by template matching, standard pattern recognition, and neural networks, the dynamic gesture recognition problem involves the use of techniques such as time-compressing templates, dynamic time warping, HMMs, and TDNN [59].

In the first part of our research, we developed a new feature to understand subtle tremor actions that might accompany some illnesses and used this feature with a discriminative model to distinguish tremor and non-tremor cases. We also used action recognition for understanding the steps of a kitchen activity in a multi-modal setting, and learned the information gain from different modalities. Then, we used this information in a new discriminative model setting to understand the actions.

Action prediction is another problem, different from action recognition, which Ryoo *et al.* defines as a probabilistic process estimating the activities in progress from partial videos, where

only the beginning part of the activity exists [75]. The traditional methods of action recognition fail on action prediction. [32, 34] used max-margin early event detectors for training temporal event detectors for early detection of actions; many others also followed their approach and use structured SVM based methods for early detection of actions [44, 45]. [42, 37] proposed methods for activity forecasting, which is predicting the future rather than understanding the partial observations. The last part of our research is on developing a system that generates notifications when the person forgets to perform an action, and we used prediction to understand the ongoing action to give proper notifications.

Vision-based systems use one or more cameras to capture images, at a frame rate of usually 30 Hz or more, and interpret those images to produce visual features that can be used to interpret human actions and recognize gestures. Typically the camera locations are fixed in the environment, although they may also be mounted on moving platforms or on people (egocentric). These systems for gesture recognition vary along a number of dimensions: number of cameras, speed and latency, structured environment, user requirements, primary features, two or three dimensional representation, representation of time, and availability of depth information.

Our main focus in this thesis will be the recognition and prediction of actions performed by a single individual using one or multiple static or egocentric cameras. We show examples of how to use this information for diagnosis or preventive purposes. Our work covers three problems: detecting hand-tremors from videos of people recorded by a static camera (Chapter 2), recognizing actions of people using an egocentric and multiple static cameras (Chapter 3), and generating notifications for missing actions using coupled action prediction and segmentation (Chapter 4). For a more comprehensive coverage, the related work in each domain is given in the appropriate chapter.

Chapter 2

DETECTING HAND-TREMORS USING MOTION FILTERING AND SVM

The hand tremor is one of the most common motion disorders caused by various neurological diseases. Currently diagnostic procedures for tremor evaluation are subjective, and there are no examinations available that can accurately indicate whether tremors are present in a patient's daily life. Early detection of tremor is extremely important for the cure of the disease that causes the tremor. Thus, in this study we aim to develop a computational method based on machine learning that can automatically detect hand tremors from a video of a patient when the patient is doing his/her daily activities. The main challenge in tremor detection is that motion is very subtle, and the signature of the motion can vary from patient to patient. We generated a training data set consisting of 173 simulated tremor/non-tremor video files. They contain very subtle and less discriminative motions. We evaluated our method through leave-one-out cross validation (LOOCV) testing and showed that in preliminary tests our method achieved 95.4% recognition accuracy. The main contributions of our study are the personalized skin detection, motion filtering and feature extraction pipeline.

2.1 Statement of The Problem

The use of computer vision in surveillance, monitoring and medical imaging has matured for years; with the advances in the hardware many time-consuming algorithms can be run in real time. The main motivation in this project is to produce a systematic methodology that can run in real time, adapt the system to users, and assist doctors in diagnoses. These types of systems are especially useful when there is no other objective measurement available. We show in this pilot study that with a simulated tremor/non-tremor dataset, our prediction system is fairly accurate and fast enough to

adapt to real time.

2.2 Related Work

The purpose of human action recognition systems is to automatically analyze the actions of the people from videos. The ability to recognize human actions and gestures is important for surveillance systems, rehabilitation purposes, diagnosing symptoms of a particular illness, home monitoring, intelligent robotics, human-computer interaction, etc. Among these, hand gestures recognition systems are especially important for virtual reality, robotics, games, smart surveillance, sign language translation and medical systems. Garg *et al.* [67] categorize the existing hand gesture recognition approaches into either 3D model-based approaches or appearance-based approaches. A straightforward method used in appearance-based approaches is to extract skin-colored regions from the image. Another method is to use the eigenspace to produce a compact description of a large set of high-dimensional data using a small set of eigenvectors [67]. Murthy and Jadon [62] adopted an appearance-based approach based on low-level hand features from a collection of 2D intensity images. However, if a system is able to extract appropriate features that describe characteristics of each action's 3-D (X, Y, T) volumes, the action can be recognized by solving an object matching problem [2].

Similar to ours, there are other studies targeted at healthcare. Cuppens *et al.* [40] developed an algorithm, based on the Horn-Schunck optical flow, to detect movement epochs from video in nocturnal datasets for pediatric epileptic patients. Ghali *et al.* tried to integrate virtual reality and machine vision technologies to produce innovative stroke rehabilitation methods. They proposed a combined object recognition and event detection system that provides real time feedback to stroke patients performing everyday kitchen tasks necessary for independent living [1]. The most similar work to ours was done by Uhrikova *et al.* [97], who also worked on hand tremors, but their purpose was different from ours in that they tried to measure the hand tremor frequencies instead of classifying tremor instances from non-tremors. They compared their calculated frequencies with those measured by an accelerometer.

2.3 Approach

As shown in Figure 2.1, our proposed method consists of three algorithmic stages. The first stage performs the (personalized) skin detection from the video of a person to locate the hand skin blobs frame-by-frame. Based on the segmented hand skin blobs, the second stage extracts the frequencies of the temporal motion change patterns as our features for tremor detection. Finally, the third stage distinguishes tremor instances from non-tremor instances. To perform skin detection, a personalized skin model is generated from one video of a person. This model is used to extract the skin blobs from training images; therefore the hand skin color in the training images should have (or have been calibrated with) similar skin color distribution with the dataset used for training the classifier and the subsequent testing videos. After skin detection, a blob extraction method is applied to extract the connected skin blobs corresponding to the hand locations (since the videos in the dataset contain only hand regions, as shown in Figure 2.2, additional hand detection is not needed). After hand blobs are extracted, directional motion features and the corresponding temporal changes can be obtained based on the optical flow of the pixels. These motion features are further converted, based on the Discrete Cosine Transform (DCT), to extract the frequency of the directional motion changes. Finally, a classifier is trained using a Support Vector Machine (SVM) with the Radial Basis Function (RBF) kernel.

Figure 2.1: Three algorithmic stages of our proposed hand tremor detection system.

The skin detection and blob extraction process is described in Section 2.3.1. The feature extraction process is explained in Section 2.3.2, and classification is discussed in Section 2.3.3.

2.3.1 Personalized Skin Detection

Our skin detection method is inspired by the adaptive skin detection method in [5]. For the training/testing set used in our experiments, a simple threshold-based skin detection would be enough,

since the hand motions are recorded on a black background. However, since the ultimate objective of this research is to develop a system that detects hand tremors of a person while he/she is doing his/her daily activities in real time in a normal environment, and since the skin models are usually not generic enough to work for everyone, we designed a personalized skin detection system. In order to train this skin detector model, the end-user is asked to move his/her hand for 3 seconds to obtain a training video. The skin detector decides on the skin regions by both thresholding and using the moving pixels of the video as described below.

For personalized skin modeling only 3 seconds of one 30 fps video (90 frames, converted from RGB to HSV format) of one person is used. For each frame of the training video, a hue threshold and an intensity threshold are applied to select probable skin pixels on hue and intensity planes to obtain thresholded hue and intensity planes (HueT, IntensityT). Then a median filter is applied on the HueT and IntensityT to remove salt and pepper noise on both planes, and the intersection of the filtered planes is computed to obtain a probable skin region (ProbableSkin). The dense optical flow [55] is then computed on the ProbableSkin region, and the pixels having a larger optical flow magnitude are selected (MovingPixels). By intersecting ProbableSkin and MovingPixels regions, a FinalMerged image is obtained, and two histograms corresponding to the hue and intensity colors of the FinalMerged image are derived. Finally, a Gaussian function is fit to each histogram and used as the skin model for further processing. After selecting the pixels according to the generated skin model, a standard blob detection algorithm for extracting 8-connected components [10] is applied, and regions having a size smaller than a pre-specified threshold are discarded.

2.3.2 Feature Extraction

The feature extraction stage starts with applying the Lucas-Kanade [55] optical flow (OF) detector for each blob in three consecutive frames. Because our goal is to detect subtle motions, we need to calculate dense optical flow, which is very computationally expensive if calculated on every pixel. Thus one of every 4 pixels of the blob area is used. Then the selected pixels' locations on the subsequent three frames are used to calculate the motion directional changes. Figure 2.2(a) shows the positions of the selected points in these three consecutive frames, with a sampling rate of 10

for a clearer representation.

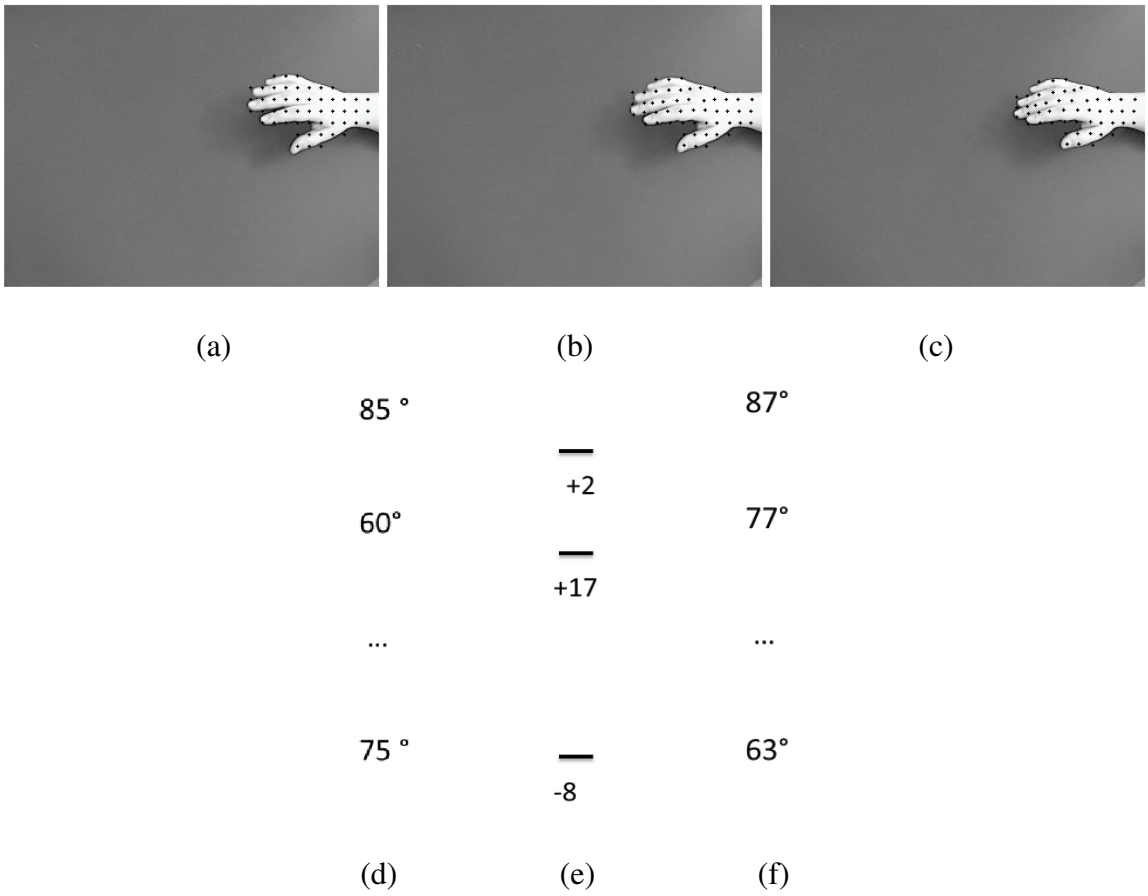


Figure 2.2: (a) Initial points. (b) Positions on the next frame. (c) Positions after two frames. (d) Motion angles from frame (a) to (b). (e) MDC from frame (a) to (c). (f) Motion angles from frame (b) to (c).

In a tremor movement, the main discriminative features are not the motion directions themselves, but the motion directional *changes* between consecutive frames. While directional features are sensitive to rotation, directional change features are more rotation, scale and translation invariant. Therefore, to extract the tremor signature patterns, motion directional changes (MDC) are calculated. To extract the angle of the MDC (see Figure 2.2 (e)) for each hand blob, we subtract the motion directional angles (see Figure 2.2 (d) (f)) between 3 consecutive frames. To have more precise hand location information, instead of using the position inferred from optical flow

and suffering from error propagation, the process of calculating MDCs should use the skin blobs, separately derived with skin detection and blob extraction from every frame.

Note that while one part of the hand can move upward, another part can move downward, and a tremor can exist in any region of a hand. We assume (and have observed through experiments) that if a tremor occurs, the directional change of the tremor dominates all directional changes that can exist in any region of the hand. Moreover, not all hand regions exhibit tremor behavior, and averaging the MDC features over the whole hand can smooth the values while decreasing the discriminative power of the features. Therefore, we need to identify the hand regions that exhibit significant MDCs related to the tremor. For this purpose, the MDC values in the hand blob are sorted from the most positive to the most negative. If the sum of the MDC in the hand blob is positive, we take the average of the top P% of the sorted values as our representative MDC; otherwise, if it is negative, we take the average of the bottom P% of the sorted values. The result of this process gives us the representative feature component of one frame. The procedure is repeated until all the frames in the video are processed, to get a 118-dimensional feature vector, out of 120 frames, which represents the temporal evolution of dominating motion directional changes. To detect tremor, the frequency of the MDC is more useful. Therefore, we apply the Discrete Cosine Transform (DCT), as defined in Equation 2.1, to the feature vectors.

$$y(k) = w(k) \sum_{n=1}^{N} x(n) cos(\frac{\pi(2n-1)(k-1)}{2N}) \tag{2.1}$$

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}} & k = 1 \\ \sqrt{\frac{2}{N}} & 2 \leq k \leq N \end{cases}$$

$$k = 1, 2, ...N$$

Here, $N = 118$ is the length of the feature vector x, and the resulting DCT vector y has the same size. Note that we explored other feature settings, like a combination of motion direction and magnitudes, motion directions or MDC but not in the frequency domain, and different percentages of MDC features from hand blobs. However, the DCT applied to MDC features with an MDL discretization [24] produced the most discriminative features among them (see Section 2.4).

2.3.3 Classification

A classifier, which can distinguish tremor instances from non-tremor instances, was developed using an SVM [9] with Gaussian RBF kernel. When using an SVM, training vectors (N-dimensional DCT features) are mapped into a higher (maybe infinite) dimensional space by the kernels. The SVM finds a linearly separable hyper-plane with the maximal margin in the higher dimensional space. The RBF kernel has less parameters than a polynomial kernel and helps to improve the mapping accuracy, especially when the relationship between class labels and attributes are nonlinear [36]. Since the number of features is not very large, and the data is not linearly separable, RBF kernel is chosen as the kernel. Its definition is given in Equation 2.2.

$$exp(-\gamma * |u - v|^2) \tag{2.2}$$

where $\gamma = \frac{1}{2\sigma^2}$ of a Gaussian with variance σ. In our experiments we used a γ of 0.1.

Besides SVM we also experimented with other classifiers such as Random Forests, Naive Bayes and Logistic Regression. Among all, SVM with the RBF kernel achieved either comparable or the best accuracy.

2.4 Experiments

We experimented with three cross-validation settings with different top/bottom P% of points selected from hand blobs. As can be seen from Table 2.1, our experiments showed that taking the top/bottom 25% of the MDC of the hand blob to describe tremor features gave the best results among all three settings. In the first setting, the classifier model was tested using a 5-fold cross-validation; then a 10-fold cross-validation and LOOCV were tried. Moreover, to improve the classification performance and to remove the noise, the minimum description length (MDL) discretization method, proposed by Fayyad and Irani [24], was applied to the feature vectors. A drastic improvement in the classification accuracy of the trained SVM is observed when MDL discretization is applied. The MDL discretization approach recursively partitions all the known values of a feature into subintervals until minimal description length is achieved [24]. Table 2.1 summarizes

% of MDC features from hand region	5 fold Cross-Validation	10 fold Cross-Validation	LOOCV
10 %	80.3%	80.3%	79.2%
10 % with MDL	90.8%	90.8%	89.6%
25 %	85.0%	86.1%	86.7%
25 % with MDL	94.8%	94.8%	**95.4%**

Table 2.1: Classification accuracy.

the results for the features selected from top/bottom 10% and 25% of the MDC features of the hand blob.

For this research, a tremor/non-tremor dataset consisting of 90 simulated tremor cases and 83 normal hand movement cases was recorded. Each video was recorded by a fixed network camera with static background in 30 fps in the indoor environment with 6 human subjects. They have a resolution of 320×240 and a duration of at least 4 seconds (120 frames). Some of the simulated tremors are extremely subtle, and some of the non-tremor videos contain movements very similar to tremor cases for the challenge. Since our final goal is to build a real time monitoring system that distinguishes hand tremors, we believe such a system should be able to distinguish these challenging cases from the tremor cases and catch the very subtle tremors at the same time, like tapping fingers versus tremor.

2.5 Summary

The hand tremor is one of the most common motion disorders and can be a sign of certain neurological diseases. In this study, we built a system that can automatically distinguish hand tremors from other kinds of hand movements. The proposed method requires only a video of a person, which makes it suitable for diagnosis purposes. It has the capability to classify very subtle motions and distinguish tremor-like movements, such as tapping fingers or waving hand very quickly, from tremor movements. In this research, a dataset of simulated tremor/non-tremor hand motions was

prepared and the methodology tested on this dataset. Although tremor detection is a hard problem, in this pilot study, with the described methodology a classification accuracy of 95.4% is achieved with LOOCV. We believe the highly accurate segmentation of the videos has also contributed to this high accuracy.

Chapter 3

ACTION RECOGNITION IN THE PRESENCE OF ONE EGOCENTRIC AND MULTIPLE STATIC CAMERAS

We next studied the problem of recognizing human actions in the presence of a single egocentric camera and multiple static cameras. Some actions are better presented in static cameras, where the whole body of an actor and the context of actions are visible. Some other actions are better recognized in egocentric cameras, where subtle movements of hands and complex object interactions are visible. In this section, we introduce a model that can benefit from the best of both worlds by learning to predict the importance of each camera in recognizing actions in each frame. By joint discriminative learning of latent camera importance variables and action classifiers, our model achieves successful results in the challenging CMU-MMAC dataset. Our experimental results show significant gain in learning to use the cameras according to their predicted importance. The learned latent variables provide a level of understanding of a scene that enables automatic cinematography by smoothly switching between cameras in order to maximize the amount of relevant information in each frame. Although this is not the main focus of our problem, our approach is general and applicable to the standard setting of multiple static cameras. To this end, we evaluate our method on the IXMAS datasets and show promising results.

3.1 Statement of The Problem

Activities that people perform in their daily lives span a wide spectrum of actions. Recognizing some actions requires reasoning about complex human-object interactions and detailed observation of the actions. For example, recognizing the cracking of an egg requires observations about the state change of the egg and characteristic postures of the hand. Some other actions, like walking to a refrigerator, are better recognized when a holistic view of an actor is visible. The movement

of the human body provides strong cues for these kinds of activities.

The conventional setting of activity recognition involves studying the behavior of an actor from one or multiple static cameras [71]. There has been significant improvement over the last decade on recognizing actions that require observing the movements of the human body. However, in this setting, there are major challenges in recognizing actions that require subtle movements/gestures. This is mainly due to severe occlusions and distractions from image regions where the actual action is taking place.

An alternative is to use egocentric cameras (also called first-person or wearable cameras), with which the actions are observed from the actor's perspective (e.g.[74]). Although less susceptible to occlusion, egocentric cameras provide their own set of challenges. For example, the human body, which is one of the main cues for some actions, is not visible in an egocentric camera. The camera has complex motion resulting in frequent blurs and appearance distortions. Furthermore, people tend to look away when they perform actions they are comfortable with. This results in losing major parts of signals that correspond to the main action. For example, when stirring a food mixture, people look around to determine what ingredients they need next, check the time, or read the next step in the recipe. The image marked with "C" in Figure 3.1 corresponds to a sample frame from a stirring action in which the actor is actually reading the recipe.

Our goal in this research is to study the problem of understanding human actions in the presence of a single egocentric and multiple static cameras. If an oracle provides information about the importance of each camera, then the problem of recognizing human actions becomes a multi-modal classification problem. In our formulation we consider a latent variable to encode the importance of each camera for each frame and introduce a model to jointly learn the latent camera variables and the action classifiers.

Our experimental results show significant success on the challenging CMU-MMAC dataset that includes both static and egocentric cameras. Although it is not the main focus of this research, our model is general enough that it can be applied to the standard settings of multiple static cameras. Our method achieves promising results on the two versions of the IXMAS dataset.

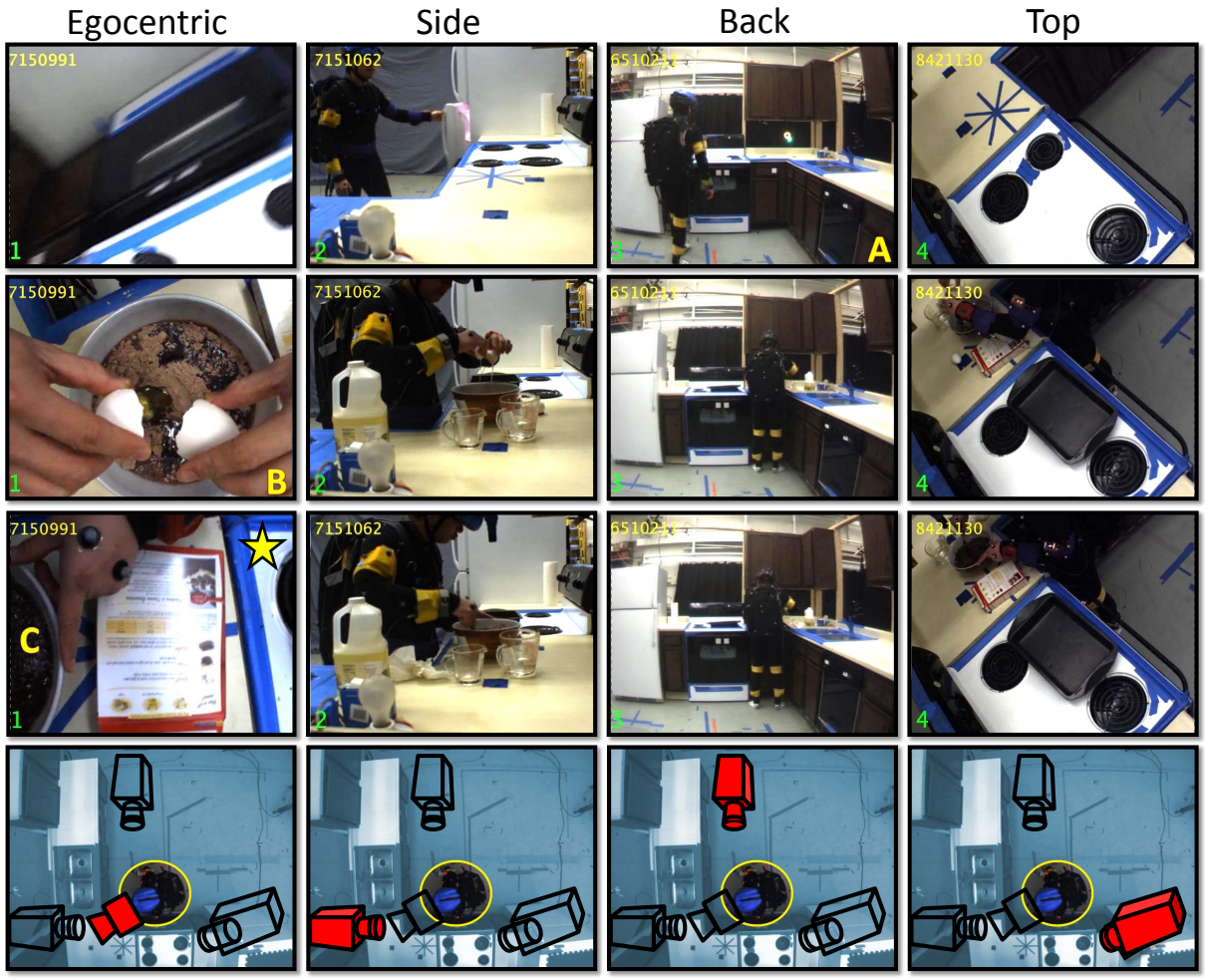


Figure 3.1: We study action recognition in the presence of a single egocentric camera and multiple static cameras. The bottom row in Figure 3.1 illustrates settings where videos of people making brownies have been recorded from an egocentric camera (first column) and three static cameras (columns 2, 3, 4). Actions like cracking an egg (image B) are better recognized using an egocentric camera where information about subtle movements and complex interaction is available. However, information about holistic body movements is typically missing in egocentric cameras. Actions like walking are better recognized from a static camera (image A). Further, people tend to look away when they perform actions that become procedural memories to them. For example the subject in image C looks at the recipe (instead of looking at the bowl) while stirring the brownie mix. This results in missing valuable action information in egocentric cameras. In this research, we show a model that can benefit from both egocentric and static cameras by reasoning about the importance of each camera for each action.

As a side product, our method enables automatic cinematography. In the presence of an egocentric and multiple static cameras, our method can automatically select a camera through which the action is better observed. By enforcing smooth transitions between cameras, we can automatically direct a scene with multiple cameras.

3.2 Related Work

Human activity recognition has attracted several researchers over the last decade. Comprehensive surveys are provided in [71][2]. Most relevant to our problem are the topics of multiple camera or multiple view action recognition, multi-modal action recognition, egocentric activity recognition, and also virtual cinematography.

Egocentric Action Recognition Egocentric action recognition typically refers to studying human actions from a camera that is mounted on the body (head to chest). This line of work has recently attracted many researchers. [18] proposes a weakly supervised bottom-up approach to segment the hands and objects using multiple instance learning. [74] uses a SIFT-based representation to address the challenges and characteristic constrains of egocentric action recognition. [19] uses hand-object interactions and proposes an object-based representation for action recognition that jointly models the objects and actions. [21] proposed a generative probabilistic model that recognizes actions while predicting gaze locations. [41] uses the hand motion and gaze information for first person activity recognition. [73] focuses on object recognition from egocentric videos, and analyzes the effect of figure-ground segmentation. [70] shows improved action recognition by explicitly modeling the change in the appearance of the objects in interaction on a novel activities-daily-living (ADL) dataset. The problem of egocentric video summarization is approached by modeling the interaction with objects in scenes [48]. [20] studies the problem of understanding simple social interactions like dialogues in egocentric settings and provides "first person social interactions dataset". [43] examines action recognition in sports videos using egocentric cameras. [65] uses eye movements and ego-motions to better recognize indoor activities. [83] utilizes egocentric action recognition for the purpose of contextual mapping. [82] studies the problem of

activity recognition using low resolution wearable cameras. The affect of gaze prediction in action recognition is explored by [51]. Actions can also be modeled through state transitions like suggested by [23]. From a different perspective, [77] handles a different the action recognition problem, where they try to understand the other person's interaction with the camera wearer, with respect to an egocentric camera.

Multi-modal/Multi-view Action Recognition: There is strong evidence that utilizing multiple modalities can improve action recognition [3]. Modalities might refer to different sources of videos or other sensors accompanying cameras. Multi-view action recognition has been approached by learning latent variables that encode the change in the appearance of actions or view points of actions [17, 94, 28], by joint learning of shared structures across multiple views [80], by hierarchical models of spatio-temporal features [93], by local partitioning and hierarchical classification of 3D HOG descriptors[91], by transfer learning [16, 15, 54], or by using spatio-temporal self-similarity descriptors [39]. Multiple datasets exists for multi modal action recognition: i3dpost [27], IXMAX [92, 91], or CMU-MMAC [11]. We use static and egocentric recordings of the CMU-MMAC dataset in this part of our research. [81, 25] have studied action recognition by combining egocentric cameras with IMU sensors using the CMU-MMAC dataset. [57, 99] used IMU sensors in the CMU-MMAC dataset to recognize actions. To the best of our knowledge, there is no method for activity recognition using an egocentric camera and multiple static cameras. Note that, approaches that require 3D reconstruction of the subject or visual hull are not directly applicable to our settings that uses egocentric cameras.

Virtual Cinematography: Cinematography is a complex subject with many constraints such as camera positioning, lighting coditions, camera selection and control constrained by aesthetic rules. A typical virtual camera control system contains egocentric point-of-view shots, switching the camera between predefined shots, and tracking a moving object by moving the camera [4]. The majority of the work has been focused on active and interactive cinematography where one has control over the position of the cameras and other parameters [53, 14, 30]. [52] uses simu-

lated videos to train a classifier to predict the parameters of cameras that best fit the scenes and present the content in coherent manner. We are mainly concerned with a passive case where multiple videos of a scene exist, and one needs to decide which camera to use for each frame. Virtual cinematography is not the main focus of this research, and our model does not address the principles of cinematography. This is a showcase to demonstrate that our model provides a level of understanding that enables camera selection.

3.3 Approach

Our task is to recognize human actions in the presence of a single egocentric and multiple static cameras. Our intuition is that each camera plays a different role in predicting an action and should be weighted according to its importance. Given the importance of all cameras for recognizing the action in a frame, the problem of action recognition reduces to a multi-modal classification problem. Unfortunately, the camera importance information is not available. We jointly learn the importance of cameras along with the action classifiers.

Learning: During training we are given videos from a single egocentric and multiple static cameras and action labels for each frame. At test time, the task is to assign an action label to each frame given the observations from all the cameras. To set up notation, let us assume that there are C cameras, N frames, M different actions and a frame i from camera j is represented by a d-dimensional feature vector x_i^j where $i \in \{1 : N\}$, and $j \in \{1 : C\}$. Each frame i is also labeled with the action label y_i where $y_i \in \{1 : M\}$. The main objective is to minimize the expected loss with respect to the conditional distribution of labels: $\min E_{p(\hat{Y}|X,A)}[\mathcal{L}(Y, \hat{Y})]$ where $\hat{Y} = \{\hat{y}_i | i = 1 : N\}$ is the set of predicted action label, $Y = \{y_i | i = 1 : N\}$ is the set of ground truth labels, $X = \{x_i^j | i = 1 : N, j = 1 : C\}$ is the set of observations from all the cameras, $\mathcal{L}$ is a form of loss function, and A is the vector of latent indicator variables α_i^j, which encode the importance of each camera j for understanding the correct action in a frame i. The α_i^j's (henceforth referred to collectively as α) are continuous variables with values in the interval $[0, 1]$.

The choice of α depends on both the observations from all cameras and the action of interest. For example, a model can learn that for walking it is better to use a static side camera, while for

cracking an egg an egocentric camera is preferred. Some actions may be partially encoded by different cameras. For example, for taking a pan out of an oven it might be better to start with a static side camera to see movement of the person toward the oven and then switch to an egocentric camera to better observe reaching for the pan. This can be encoded with a *camera selection* model that picks a camera for each frame in each action.

This procedure makes hard decisions about the importance of cameras. In practice, different cameras provide useful *orthogonal* information. For example, for the "pouring water into measuring cup" action, the side view camera encodes the movements and posture of the body while the egocentric camera captures the best view of the measuring cup and the hand movement for turning the water off. Our model takes the importance of each camera into account while making predictions about actions.

The best estimate of the camera importance variables is the one that maximizes the accuracy of action prediction. We adopt a max margin formulation where the importance of each camera is modeled by an element-wise product operator $\odot$ and A, a vector of latent camera importance variables. We stack all the observations across all cameras into an observation vector $\mathcal{X}_i = [x_i^1, ..., x_i^C]$ which is a $(C * d)$-dimensional representation of frame i, where C is the total number of cameras and x_i^j is a d dimensional vector of each frame i in camera j. To simplify the notation, we assume that the feature vectors have the same dimensionality across cameras. Each action classifier $\mathcal{W}_a$ is a $(C * d)$-dimensional classifier for action a. For each frame i, the latent camera importance variable A_i is also a $(C * d)$-dimensional vector, where $A_i = [A_i^1, A_i^2, ..., A_i^j, ..., A_i^C]$, $j \in \{1 : C\}$. For each camera j, $A_i^j = \alpha_i^j * \mathbf{1}^d$ is a d-dimensional indicator vector that ensures all the dimensions of the feature vector corresponding to a camera are weighted equally.

Our model searches for the best camera importance variables that maximize the action prediction accuracy by:

$$\min_{A,\mathcal{W},\xi} \sum_{a=1}^{m} \parallel \mathcal{W}_a \parallel_2^2 + \lambda \sum_{i=1}^{N} \xi_i^a \tag{3.1}$$

such that

$$Y_i(\max_a[(A_i^T \odot \mathcal{W}_a)^T(A_i \odot \mathcal{X}_i)]) > 1 - \xi_i^a \quad \forall i$$

$$A_i = [A_i^1, A_i^2, ..., A_i^C]$$

$$A_i^j = \alpha_i^j * \mathbf{1}^d \quad j \in \{1 : C\}$$

$$\sum_{j=1}^{C} \alpha_i^j = 1, \quad \alpha_i^j \in [0, 1], \quad \xi_i \geq 0 \quad \forall i,$$

where $\mathcal{W}$ is the matrix of all action classifiers across all cameras ($\mathcal{W} = [\mathcal{W}_1\mathcal{W}_2...\mathcal{W}_m]$), and ξ is the standard slack variable in max margin formulations.

To optimize for $A, \mathcal{W}, \xi$ we use block coordinate descent in the dual of optimization 3.1. This involves estimating $\mathcal{W}$ for fixed A and optimizing for A given fixed $\mathcal{W}$ in the dual. Optimizing for $\mathcal{W}$ given a fixed A reduces to a quadratic programming problem. We initialize $\mathcal{W}$ by independently trained classifiers for each action. We calibrate these classifiers using [78].

To encode higher order correlations in the feature space we also consider different combinations of cameras (each combination of cameras can be though of as a new dummy camera). Section 3.4 shows the benefits of considering such higher order correlations via camera combinations.

Inference: Our model requires estimates of A for test frames during inference. Optimizing 3.1 for A during inference is not possible because Y_i is not known. Instead, we train classifiers to predict A, using the optimum A values obtained from optimization 3.1 on the training set as labels. The most confident camera classifier that results in correct prediction of actions is used as the camera label for each frame. We use an RBF SVM to train these camera indicator classifiers. At test time, we use the camera indicator classifier responses as estimates of A for a given frame. We then use these A's to predict action labels using $(A^T * \mathcal{W}_a^*)^T(A * \mathcal{X})$. The action with the highest score claims the frame. To enforce temporal consistency at the test time, we use Viterbi smoothing with dynamic programming.

3.4 Experiments

We evaluate our model on how accurately it can predict actions in the settings where an egocentric camera plus multiple static cameras observe human activities. We compare our method with state of the art methods for multi-view activity recognition, such as Latent Dynamic CRF and Hidden Unit CRF, and several baselines that aim at evaluating different components of our model. To quantitatively evaluate the quality of the learned camera indicator variables (α), we utilize them in a virtual cinematography task.

In this part of our research, we are interested in learning to predict human actions in the presence of one egocentric camera plus multiple static cameras. Our experiments are mainly designed to support this task. However, our model is general and applicable to standard settings of multiple view static cameras. To show the capabilities of our model in the settings of multiple static cameras, we also test our method on standard and new IXMAS dataset.

Multiple Static and an Egocentric Camera **Dataset:** We chose the challenging CMU Multi-Modal Activity dataset (CMU-MMAC) [11] because it has multiple static and one egocentric videos of subjects performing kitchen activities. We use brownie making videos, because frame-level annotations of actions are provided. 11 out of 40 different actions of the dataset are unique to different videos. After discarding unique actions, "none" categories, and dropped frames, the remaining dataset consists of 28 different actions, for 5 different subjects, recorded from one egocentric and 3 static cameras (total of $\sim 42K$ frames, 1/30 sample rate). The final actions include close fridge, crack egg, open brownie bag, pour brownie bag into big bowl, pour oil into measuring cup small, twist on/off cap, stir, take fork, walk to fridge, switch on, open drawer and more.

Features: Similar to the creators of the CMU-MMAC dataset [81], we use the GIST [66] features for all the methods and baselines in our experiments on the CMU-MMAC dataset. Eight orientations per scale and four blocks per direction are used with PCA. The choice of GIST features is not perfect but they are suitable for making the comparisons of methods in this research, where emphasis is not on the feature engineering.

Experimental Setup: We use leave-one-video-out (each video belongs to one actor) as our experimental protocol and average accuracy (Avg. Acc.) as our evaluation metric. Our model achieves an average accuracy of 54.62.

Comparisons to State-of-the-Art Methods: We compare our method with state-of-the-art methods in multi-view, temporal classification. We select state of the art methods that are applicable to the settings of an egocentric and multiple static cameras. Note that methods that rely on 3D estimates of the visual hull of the subjects are not directly applicable to egocentric cameras.

Different versions of latent model CRFs have been successfully used in multi-view action recognition. In particular, we compare our method with Latent Dynamic CRF and Hidden Unit CRF. Latent Dynamic CRF [60] is a discriminative method with a strong track record for multi-view activity recognition. It models the sub-structure of action sequences by introducing hidden state variables. In our experiments the best results are obtained by using one hidden node per label. Table 3.1 shows that our model outperforms LDCRF on the challenging task of action recognition with one egocentric and multiple static cameras. We also compare our model with Hidden Unit CRF [88] where there are hidden nodes between action classes and features. Those hidden nodes can reveal the latent discriminative structure in the features. For each frame a hidden unit CRF can represent nonlinear dependencies. The best results in our experiments are obtained by using a total number of 100 hidden units. Table 3.1 shows that our method also outperforms Hidden Unit CRF.

Both LDCRF and Hidden Unit CRF approach the problem of action recognition by joint reasoning over time. In the case of combining egocentric and static cameras, coupling joint temporal reasoning with discovering the latent structure in the high-dimensional feature space makes the problem extremely challenging. We postulate that separating temporal reasoning from discovering the latent structure might result in more accurate estimates of the latent structure.

Baselines: To further analyze our model, we examine the effects of each component in our model with several baselines designed to challenge different components in our formulation. Except for baseline 1, which measures the effect of Viterbi smoothing, all other baselines use a final Viterbi smoothing stage.

Baseline 1: To examine the importance of encoding temporal information, we remove the

Approach	**Avg. Acc.**	**Avg. per Class Acc.**
Latent Dynamic CRF [60]	41.55	33.57
Hidden Unit CRF [88]	24.13	26.22
Our Method	**54.62**	**45.95**

Table 3.1: The comparison of our model with two of the state-of-the-art latent CRF models: Latent Dynamic CRF and Hidden Unit CRF. Our model outperforms both of these methods.

Viterbi temporal smoothing at inference and compare it with our full model. Table 3.2 shows that encoding temporal information helps action recognition in our setting.

Baseline 2: To verify the effects of binary versus continuous α (camera selection vs using all cameras with respect to their calculated importances), we train our model with the binary α constraint, where $\alpha_i^j \in \{0, 1\}$. This forces our model to pick only one camera combination. Table 3.2 shows results when binary α is used.

Baseline 3: To challenge the observation about encoding higher order correlations using camera combinations, we compare our method with a version that uses only four cameras (no combination). This baseline uses continuous α. Results in Table 3.2 show that leveraging higher order relations of cameras (camera combination) in our latent discriminative model improves action recognition.

Baseline 4: This baseline is similar to the previous one, with one modification: using binary α (camera selection). Table 3.2 shows that both higher order correlations of features (camera combination) and using all cameras according to their importance improves the recognition accuracy.

Baseline 5: Our optimization learns an α for each frame. One could reasonably worry about a large number of parameters to learn. An alternative is to search for an α for each action. This implies that the importance of the cameras are fixed for all frames during the course of an action. Baseline 4 corresponds to experiments with per-action α model. The results in Table 3.2 support our intuition that the importance of the cameras changes during an action.

Approach	**Avg. Acc.**	**APC Acc**
Baseline 1	41.80	44.05
Baseline 2	52.00	41.55
Baseline 3	47.93	42.45
Baseline 4	44.58	35.01
Baseline 5	48.83	45.89
Baseline 6	43.55	39.81
Baseline 7	35.04	36.97
Our Method	**54.62**	**45.95**

Table 3.2: We compare our method with several baselines. Removing temporal smoothing, considering binary alphas, not considering camera combinations, and encoding per-action α hurts the performance of our model. This supports our intuitions about different parts of our model.

Baseline 6 examines the need for latent variables while learning to recognize actions across multiple cameras. In this baseline, we fuse multiple cameras without the latent variable. This late fusion baseline uses action models trained independently for different cameras and fuses them by a second layer RBF SVM. This baseline uses higher level correlation of features(camera combination) and Viterbi smoothing. Our model differs from this baseline in using the latent α to explicitly encode the relationships across cameras in a discriminative manner. Table 3.2 shows the importance of our latent variable.

Baseline 7: Another way to combine multiple cameras is to combine all the observations at the feature level and expect the classifiers to discover complex relationships in this high-dimensional data. Baseline 7 corresponds to this early fusion model. The results in Table 3.2 imply that complex relationships cannot be reliably discovered by just fitting a classifier to the combination of features. This baseline also uses Viterbi smoothing.

As a sanity check, we also determine if there is apparent discriminative information in any single cameras that can bias the recognition performance on the CMU-MMAC dataset. To do

Camera Combinations	**Avg. Acc.**	**Acc. Drop %**
Ego + Side + Back + Top	44.58	0
No Side	42.97	3.6
No Top	40.56	9.0
No Back	43.54	2.3
No Ego	**27.99**	**37.2**

Table 3.3: To evaluate the importance of each camera in our formulation we remove a camera from our model (with binary α) one at a time, and report the drop in the accuracy. Egocentric camera is the most informative camera in this setting and the back static camera is the least useful one.

that, we train RBF-SVM action classifiers with the Viterbi algorithm using only one camera. Using egocentric, static back, static side, and static top cameras alone results in average accuracies of $37.92, 22.07, 21.26, 20.86$, respectively (compared to average accuracy of 54.62 using our full model).

To further explore the importance of each camera in our model, we remove one camera at a time from our model with binary α, and report the percentage drop in the performance numbers. Table 3.3 compares the accuracies and also the percentage drop in removing each of the cameras. The most informative camera in our setup is the egocentric one, and the least informative one is the back camera.

To qualitatively evaluate the learned α's, we depict the distribution of α for some frames belonging to the "crack egg" and "take baking pan" actions in Figure 3.2. When the subject is cracking the egg, our method assigns high weights to the egocentric camera and once the subject's head is turning (and the egocentric camera is not informative) then our method assigns more weights to the side static camera. For the "take baking pan" action our method assigns more weights to the back camera when the subject is moving toward the cabinet. Once the cabinet door is open and the subject starts searching for the baking pan, the egocentric camera claims more weight. Toward the end of this action, the side static camera becomes more informative, and our method assigns

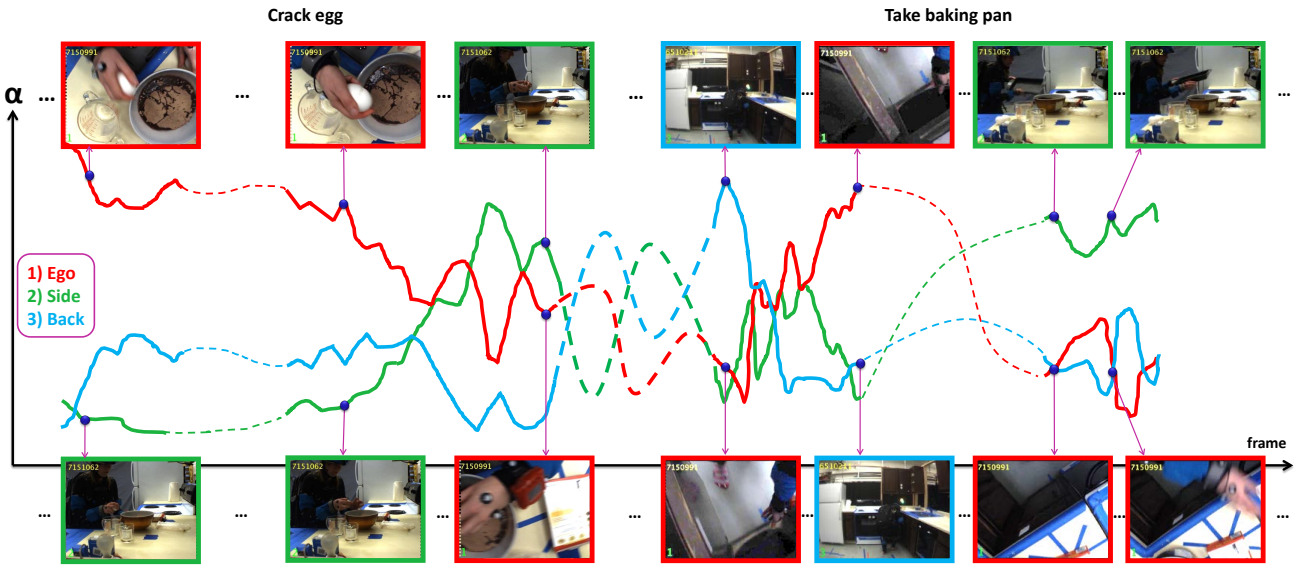


Figure 3.2: The distribution of the learned α for "crack egg" and "take baking pan" actions: Our model learns that the egocentric camera contains maximum information for cracking an egg (highest α) when the subject interacts with the egg. Toward the end of the action when the subject looks away from the action, our model assigns more weight to the side static camera that represent the action best. For the "take baking pan" action our model allocated more weight to the static back camera when the subject walks to the cabinet. After opening the cabinet door, our model assigns more weights to the egocentric camera. This is followed by putting more weights on the side camera when the subject is about to put the pan on the counter top.

more weight to that camera. Figure 3.3 shows an interesting behavior of the learned α. For the "stirring" action both the egocentric and the side static camera are highly informative. As a result, our method keeps switching between them, putting more weight on one after the other.

Multiple Static Cameras Although not exactly aligned with the focus of this research, we also show that our model is general enough that it can be applied to standard settings of multiple static cameras. We test our model on the well-studied IXMAS dataset[92], and a newer version of it, which we refer to as EPFL-IXMAS.

The IXMAS dataset has 11 actions performed 3 times by 10 people recorded by 5 five synchronized static cameras. The newer dataset, EPFL-IXMAS, contains the same action categories

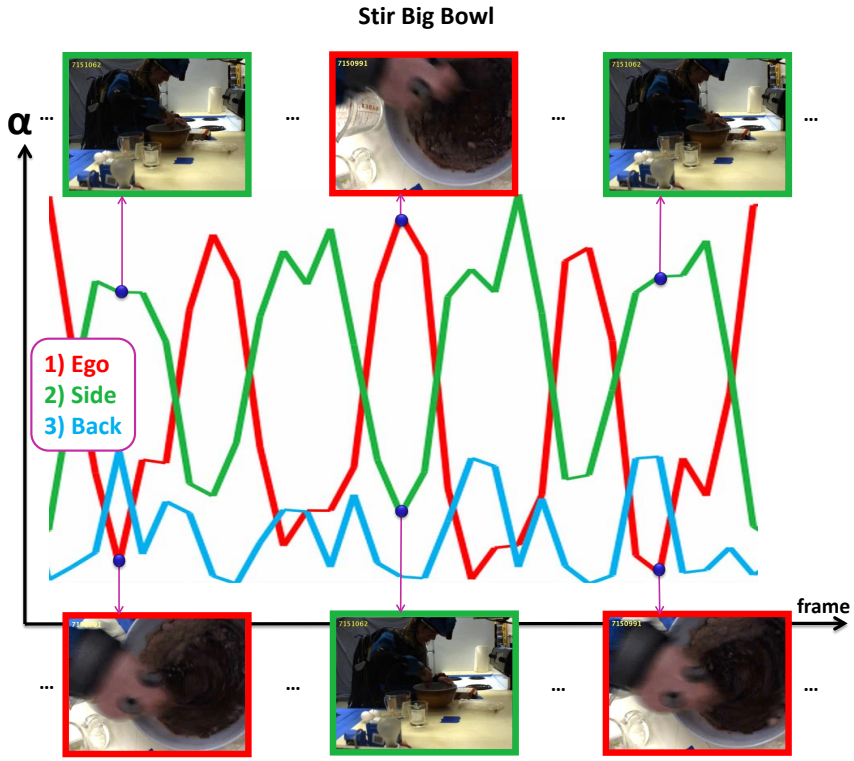


Figure 3.3: Distribution of α for some example frames of "stirring" action. This is an example of a case where our model is undecided about which camera to use because both egocentric and side cameras are almost equally informative. As a result, our model switches between the cameras periodically. This is aligned with the nature of this action. As far as the action recognition is concerned this will maximize the discrimination. To avoid frequent camera jumps in virtual cinematography we use segmented least squares to smooth camera transitions.

recorded by 5 different cameras having different viewpoints, with different actors, where 2/3 of the actions were recorded while actors were partially occluded by objects [91]. On both datasets we use the embedded 3DHOG descriptors described in [91].

Tables 3.4 and 3.5 compare the results of our method with other state-of-the-art methods reported on IXMAS and EPFL-IXMAS datasets.

Approach	**Avg. Acc.**
Turaga et al. [86]	98.8
Wang et al. [90]	93.6
Weinland et al. [92]	93.3
Pehlivan et al. [68]	90.9
Wu et al. [95]	88.2
Haq et al. [87]	83.7
Yan et al. [96]	78
Ours	**97**

Table 3.4: Comparison to state-of-the-art on IXMAS dataset. Note that some of the above methods use 3D features calculated from all the cameras at the same time ([86, 92, 68]), some fuse individual camera features ([95, 96]), while [90, 87] use features obtained from each view without fusion.

Approach	**No Occ.**	**Occ.**	**All**
Mahasseni et al. [56]	–	–	88.6
Weinland et al. [91]	86.3	76.7	–
Ours	**95.6**	**90.5**	**91.2**

Table 3.5: Comparison of state-of-the-art with our results on EPFL-IXMAS dataset. All numbers show Avg. Acc. [56] trained using 2/3 of the EPFL-IXMAS dataset for training, while [91] used original IXMAS recordings together with artificially occluded one. We follow [91] in a multi-view test setting. The "Occ." column shows the results only on the occluded sequences while the "No. Occ." column shows the results only on non-occluded sequences. Note that the first two rows use a single camera at test time.

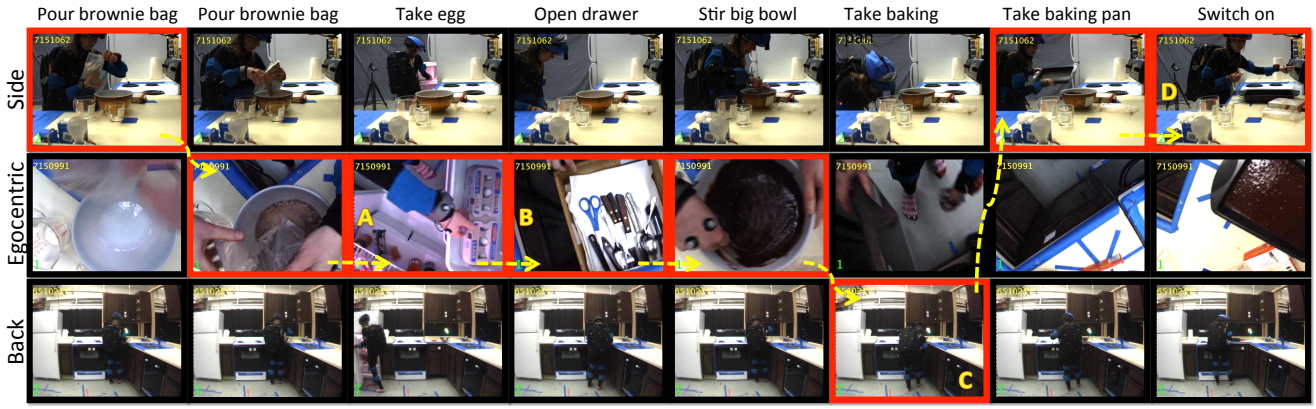


Figure 3.4: This figure depicts some frames across 3 cameras (side, ego, back) for eight different actions. Red boxes indicated the frames that our method selects for the virtual cinematography. It is interesting to see that for the "take egg" action our model chooses the egocentric camera where one can clearly see the eggs (Image A). For "taking a fork", our model also switches to the egocentric camera where one can see the items inside the drawer (Image B). When the subject walks to the cabinet our model switches to the back camera (Image C). When the subject is about to turn the oven on our model picks the side camera where the extended arm of the subject is visible.

Using α to Direct a Video Scene To qualitatively evaluate our inferred α, we use them to direct the scene of "making brownies" recorded from multiple static and an egocentric camera from CMU MMAC dataset. The task is to select which cameras to use for each frame. We use our learned α to select a camera per frame. Picking the camera with maximum α value results in undesirable frequent switches between cameras. To avoid that we use segmented least squares to smooth the camera transitions. Figure 3.4 shows examples of the frames across 3 cameras: static side, static back, and egocentric camera. Frames with red boxes are selected by our method. It is interesting to see that when the subject searches for the fork in a drawer, our method switches to the egocentric camera where all the items inside the drawer are visible (Image B in Figure 3.4). When the subject moves toward the cabinet to take a baking pan, our method switches to the back camera where human movements are clearly visible (Image C in Figure 3.4). When the subject switches on the oven, our method picks the side static camera where the extended arm of the subject is visible (Image D in Figure 3.4).

3.5 Discussion

Confusions in Action Prediction on the CMU-MMAC Dataset Figure 3.5 shows per-class action prediction accuracies and the most confusing action for all actions in the CMU-MMAC dataset. The most confusion comes from very similar actions like “take cup small” and “take cup big”, or “take pan” and “take baking pan”. The top three actions in terms of their recognition

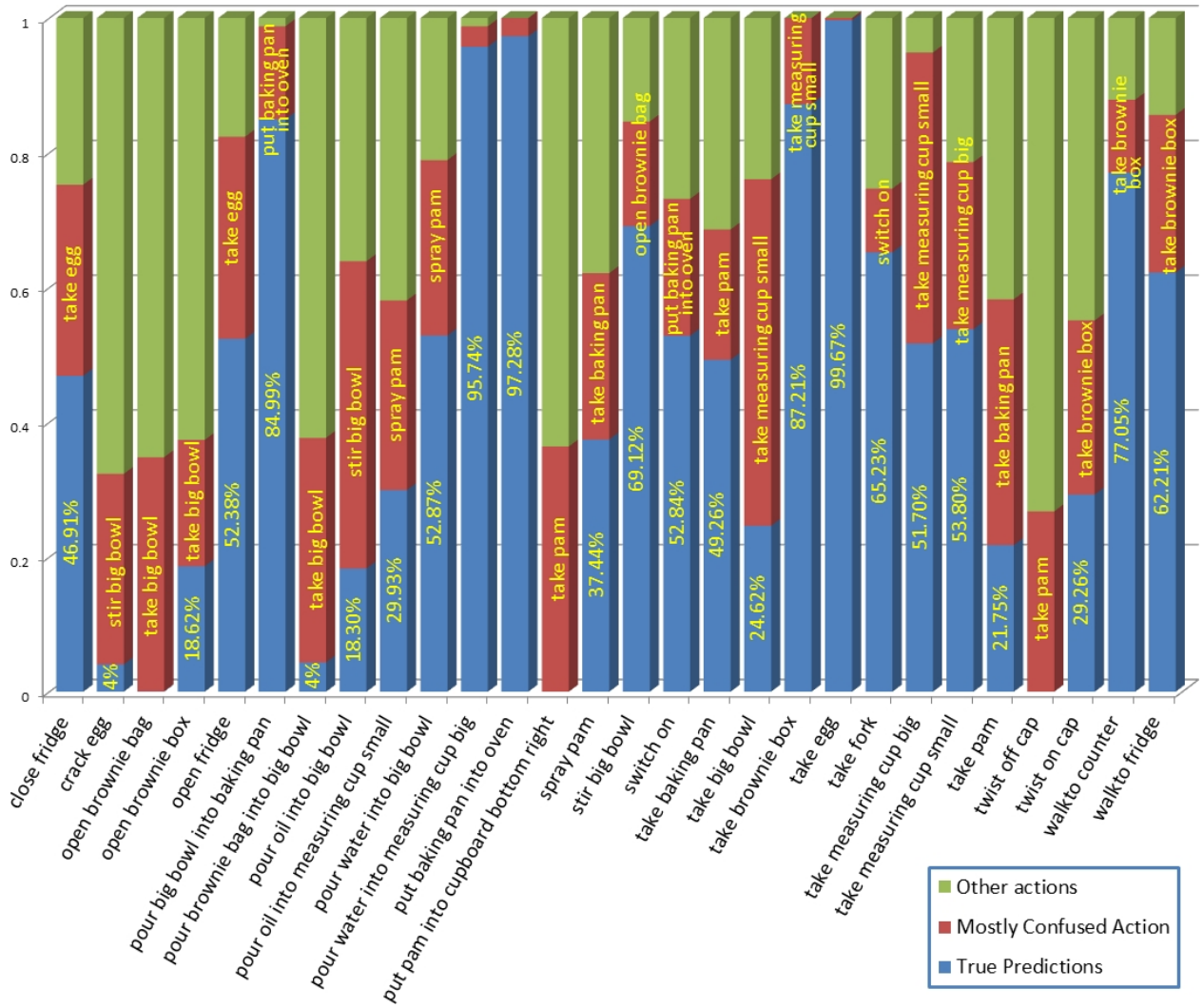


Figure 3.5: Analysis of the action prediction on the CMU-MMAC dataset.

accuracy are “take egg”, “put baking pan into oven”, and “pour water into big cup“. The three most difficult actions for our model are “twist off cap” and “put pan into cupboard bottom right”, and “open brownie bag”. Another source of confusion stems from the fact that some actions share very similar settings. For example, “open fridge” and “take egg” are frequently confused because the majority of the scene in the “take egg” action corresponds to the half-open door of the fridge. In addition, some actions share very similar body movements. For example, reaching for the same cabinet to take a PAM or a baking pan.

We use GIST features as suggested by the creators of the dataset. Some of the confusions can be resolved with a richer representation. Object-based representations and hand-segmentations may provide more information to reduce the aforementioned confusions.

Figure 3.6 shows the confusion matrix we obtained from our experiments on the CMU-MMAC dataset. The numbers are rounded and represent percentages.

Confusions in Action Prediction on the IXMAS Datasets Figures 3.7, 3.8, 3.9, 3.10 show the confusion matrices regarding our experiments on the two versions of the IXMAS datasets. All numbers are rounded and represent percentages. While Figure 3.7 represents the recognition results on the original IXMAS dataset, Figures 3.8, Figure 3.9, Figure 3.10 show the confusion matrices belonging to our experiments on the EPFL-IXMAS dataset.

Directing Scenes To avoid undesirable frequent camera switches while directing scenes, we use segmented least squares with 10-20 frames for smoothing. We do not switch between the cameras for inconsistent actions. Quantitative evaluation of scene direction is not trivial. To this end, with help from a professional cinematographer, we label each scene by the camera index that, we think, best represents the action in that frame. This is a tedious task that requires watching 4 videos at the same time; thus we only labeled videos corresponding to one subject. There is no unique answer to this problem and for some frames multiple cameras might be used. When compared with manual labels, our method achieves an accuracy of 54.91%.

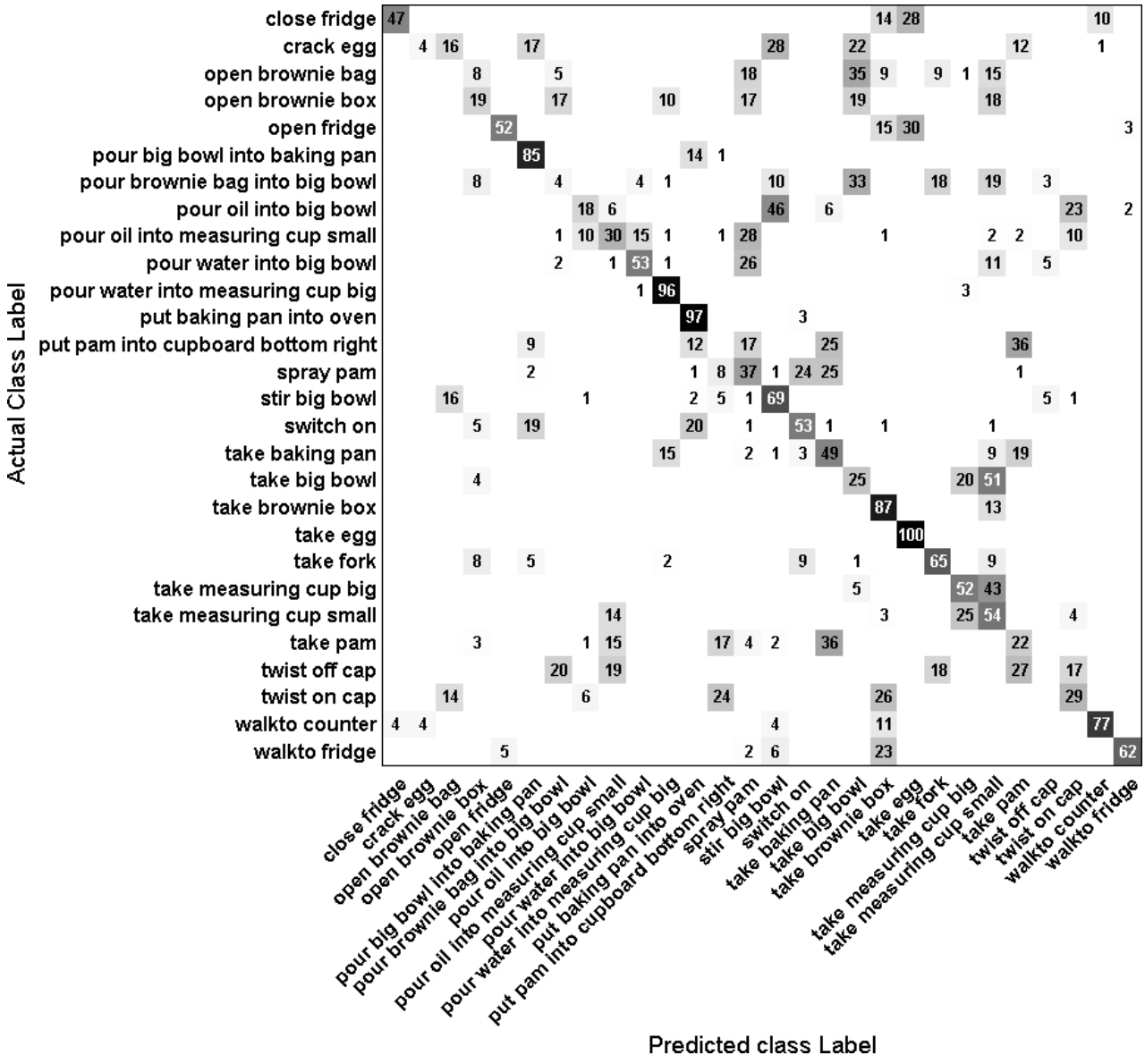


Figure 3.6: Confusion matrix resulting from the experiments on the CMU-MMAC dataset.

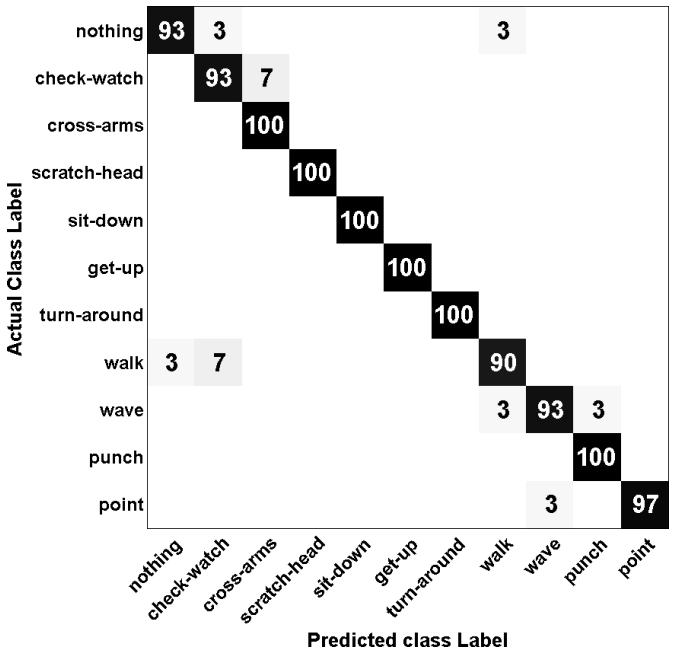


Figure 3.7: Confusion matrix resulting from the experiments on the original IXMAS dataset.

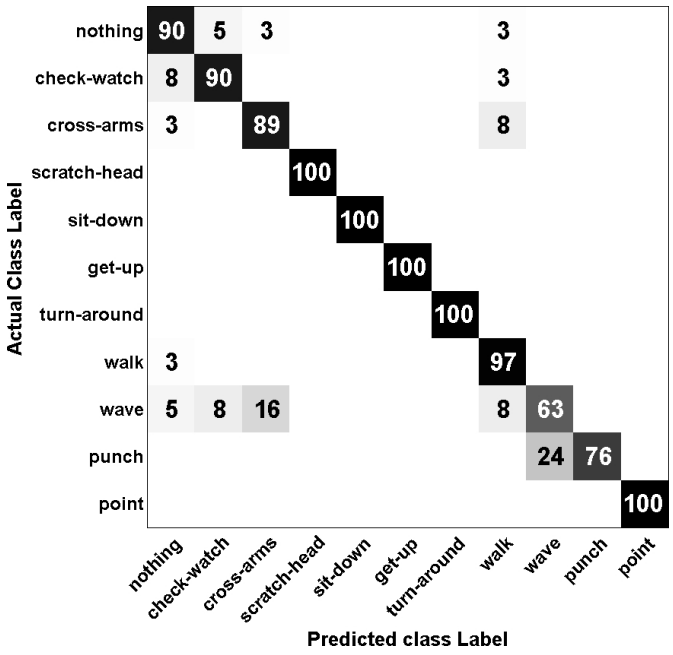


Figure 3.8: Confusion matrix resulting from the experiments on both the occluded and the non-occluded recordings of the EPFL-IXMAS dataset.

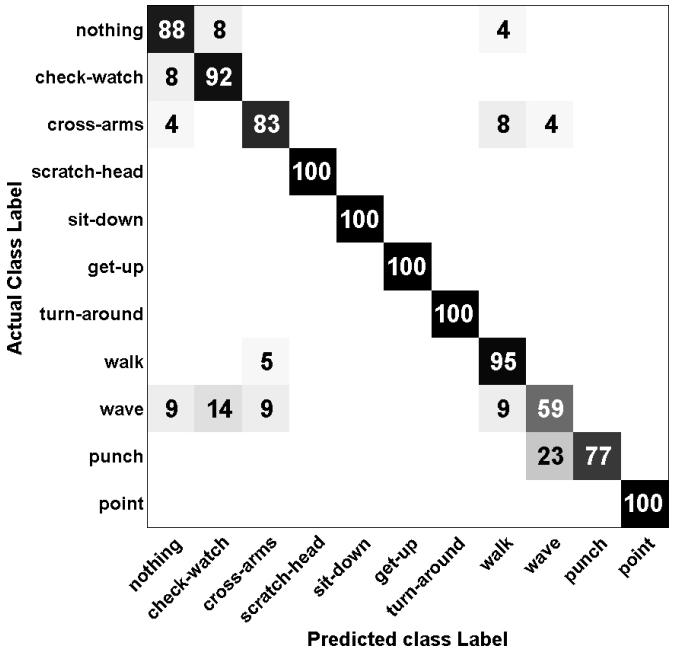


Figure 3.9: Confusion matrix resulting from the experiments on the occluded recordings of the EPFL-IXMAS dataset.

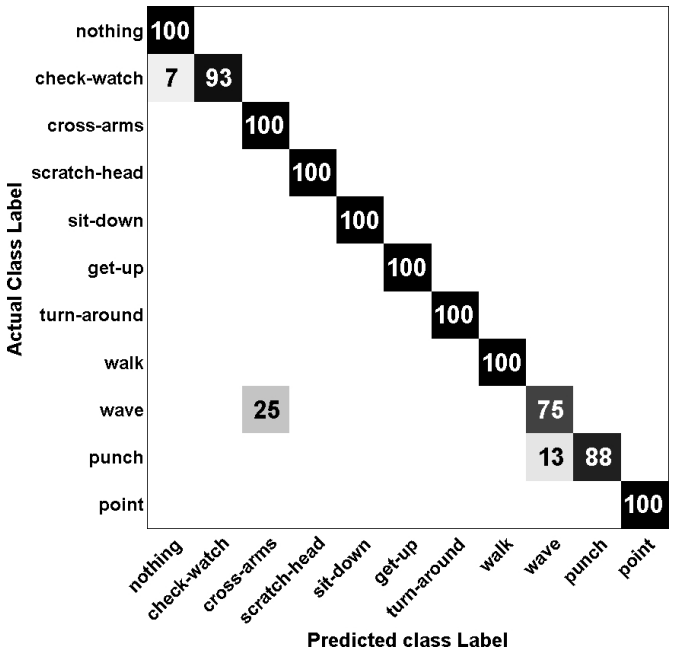


Figure 3.10: Confusion matrix resulting from the experiments on the non-occluded recordings of the EPFL-IXMAS dataset.

3.6 Summary

We introduced a model that, for each query frame, discriminatively predicts the importance of different cameras and fuses the information accordingly. We showed that our model outperforms state-of-the-art methods that jointly reason across time and actions. Our hypothesis is that joint reasoning across camera importance and actions followed by temporal smoothing is a more manageable learning problem than joint reasoning over time. By being more focused on frame-level discrimination, our model learns meaningful latent variables and can discriminatively suggest the importance of each camera for each frame. The next step involves extending our explicit latent variable formulation to also perform joint reasoning over time. Our learned camera indicator variable provides a level of understanding of the scene that enables meaningful camera selection for automatic cinematography. Our model does not take into account the principles of cinematography.

Our method shows very successful results on the challenging CMU-MMAC dataset by fusing the information from the egocentric and static cameras as needed. To the best of our knowledge, this is the first attempt in combining egocentric action recognition and conventional static camera action recognition. Moreover, our experiments show that our model is on par with the state of the art methods on two versions of the IXMAS datasets, with excellent recognition results on the occluded scenes of the EPFL-IXMAS.

Chapter 4

GENERATING NOTIFICATIONS FOR MISSING ACTIONS USING COUPLED ACTION PREDICTION AND SEGMENTATION

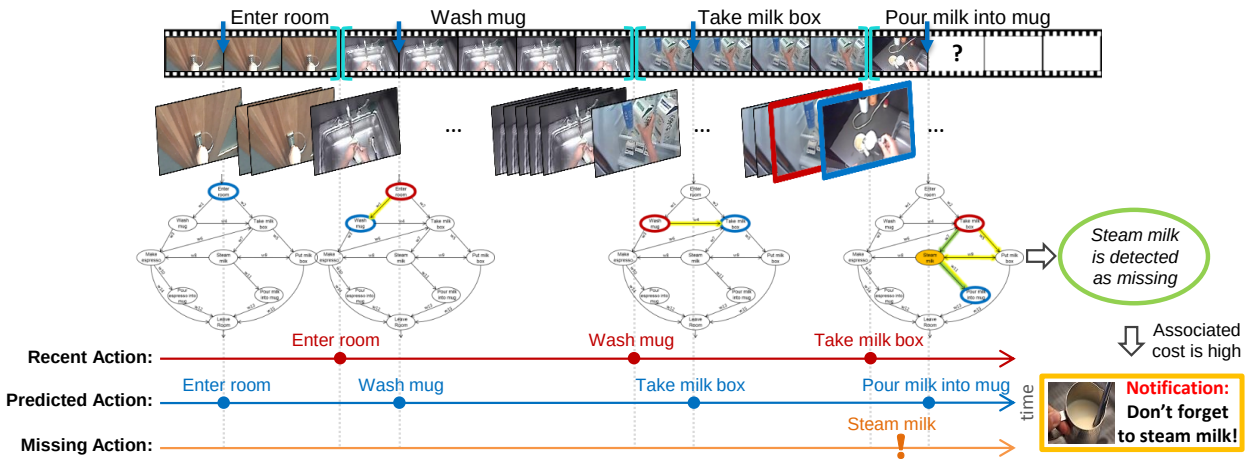


Figure 4.1: Our purpose is to issue notifications about missing actions given an unsegmented input stream of egocentric video. For the latte making sequence above, our system recognizes the actions that happened so far, predicts the ongoing action, reasons about missing actions and the associated cost, and generates notifications for the costly missing actions. In this figure, the brackets refer to segmented action boundaries, the blue arrows show the prediction points and the graphs below show the inter-action dependencies. The most recently completed action is marked in red, the predicted action is marked in blue, and the missing action is marked in orange. In this example, the actor is about to miss an important action: *steam milk*, and a reminder for that is given.

We all have experienced forgetting habitual actions among our daily activities. For example, we probably have forgotten to turn the lights off before leaving a room or turn the stove off after cooking. In this chapter, we propose a solution to the problem of issuing notifications on actions that may be missed. This involves learning about interdependencies between actions and being able to predict an ongoing action while segmenting the input video stream. In order to show a proof of concept, we collected a new egocentric dataset, in which people wear a camera while

making lattes [1]. We show promising results on the extremely challenging task of issuing correct and timely reminders. We also show that our model reliably segments the actions, while predicting the ongoing one when only a few frames from the beginning of the action are observed. The overall prediction accuracy is 45.9% when only 10 frames of an action are seen (2/3 of a sec). Moreover, the overall recognition and segmentation accuracy is shown to be 72.3% when the whole activity sequence is observed. Finally, the online prediction and segmentation accuracy is 67.9% when the prediction is made at every time step.

4.1 Statement of The Problem

We all have witnessed prospective memory failures in our daily activities. How often do we forget to turn the lights off before leaving a room? Or to turn the stove off after boiling water? Prospective memory failures are often devastating and sometime disastrous. Several researchers in cognitive sciences have explored the causes of prospective memory failures and shown effective medical practices [12]. One of the most effective medical practices is to use reminders [31] [61].

In this chapter, we study the problem of automatically issuing reminders about actions that users might forget. These reminders can potentially prevent catastrophic events. Such notifications can also be useful for individuals with memory and cognition problem such as Alzheimer's disease. We also envision that these reminders can eventually be helpful in completing complex procedures.

Egocentric cameras are suitable for these kinds of frameworks, as they move with the user. For a proof of concept, we collected a latte making dataset using an egocentric camera, in which subjects wore a head-mounted camera to record their actions during the latte making activity in their own style.

We propose a novel system, which notifies people when they forget to perform an action before their current action finishes. Such a system needs to understand the ongoing action before it completes and decide whether there is a missing action or not, and if it is necessary give a notification about this missing action. For that, it needs to extract inter-action dependencies and possess

[1] A latte is a coffee drink made with espresso and steamed milk [64].

a decision mechanism that takes the importance of the missing action into account. Furthermore, predicting the ongoing actions as early as possible is needed. For example, when a *leave room* action is predicted, the system can check whether it needs to remind the user for *turning the stove off* by checking whether *the stove is set to on* and decide how important that reminder is by analyzing the associated cost.

4.2 Related Work

Egocentric Vision: Recently, egocentric activity recognition is gaining a lot of interest among the computer vision community. There are many works analyzing either the camera wearers actions [19, 22, 19, 70, 43, 58] or the people surrounding the camera wearer [77, 76], please also see Section 3.2. For a review of egocentric vision please see [41].

Our notification system requires prediction of actions from partial observations. [76] uses the video recordings of a camera worn by a robot and *predicts the actions of the people around the robot*. To the best of our knowledge, we are the first to study action prediction in egocentric videos, where the *camera wearer's actions* are analyzed. This setting is especially suitable for daily living activity prediction.

Action Prediction: Action prediction is defined in [75] as a probabilistic process, which estimates the action in progress when only the beginning part of the action exists. [75] represents an action as an integral histogram of spatio-temporal features and uses a dynamic bag-of-words approach to predict. [76] predicts the human actions during human-robot interactions. They use previous actions as cues to infer the following ones, therefore requiring particular action pairings. It is different from our model in that, although there is a partial-order among the actions in our model, we do not require one action to be a predecessor of another; a person either steams the milk first and grinds the coffee next or vice-versa. [8] studies recognizing actions where the missing part of the action (temporal gap), can be anywhere. They used sparse coding to determine the likelihood that a certain type of activity is present in the partially observed video. [49, 50] propose an activity prediction framework; their purpose is to predict higher-level activities using actionlets, and the dependencies between them are represented by a probabilistic suffix tree.

A max-margin early event detector (MMED) is proposed by [32, 34] for training temporal event detectors for early detection of actions. Their model extends the structured output SVM (SOSVM) to accommodate sequential data. In general MMED is designed for *early event detection*, while for the problem we are tackling we need to *recognize the actions from partially observed videos*. [98] proposes a moving pose descriptor framework on depth sequences, which is also used for activity detection and low-latency recognition. After the successful application of a modified SOSVM model to an early event detection problem, the popularity of SOSVM-based models for action prediction has risen over the years. [44] proposed a multi-scale model for action prediction for already segmented videos. Their model uses both the global and the local history of features to learn temporal dynamics. Similarly, [45] used hierarchical "movemes" for describing human movements at multiple levels of granularities, ranging from atomic movements to the ones that exist in a larger temporal extent to predict actions in segmented videos. Both [44, 45] used modified versions of the SOSVM. [13] explored the trade-off between accuracy and observational latency by determining distinctive canonical poses of the subjects.

Activity forecasting aims to predict future actions rather than recognizing ongoing ones from partial observations. [42, 37] proposed methods for activity forecasting, which is out of the scope of this work.

Event detectors in general suffer from a major fundamental drawback of overlapping detections for different actions. In our approach, besides prediction we also address temporal segmentation in a coupled, supervised setting.

Joint Segmentation & Recognition: Most studies in the literature perform recognition/prediction on pre-segmented videos. There exist techniques for coupled segmentation/recognition, but most of them rely on generative models [26, 6, 47, 100]. [33] proposed a maximum margin temporal clustering, which determines the start and the end of each segment, and discriminates among temporal clusters by a multi-class SVM. [63] represented activities as temporal compositions of motion segments. To train their action classifier model, they used both temporal composition information and visual features of motion segments. [7] used spatio-temporal graphs of an activity class to parse a test video by matching its graph with the closest activity model. [69] achieved

online parsing of videos to its actions and sub-actions using specialized grammars, which define temporal structure and can be parsed with a finite-state-machine. [35] jointly performed video segmentation and action recognition using a method based on a discriminative temporal extension of the spatial bag-of-words model. The most similar to our work, [23] segmented actions by using a model that encodes the change in the states of the world.

Orderings in an Activity Sequence: An action is defined by [72] as a sequence of key poses of actors depicting key states. Their model assumes a partial ordering between the key frames, making it tolerant to action duration. [84] proposed a method to learn the partially ordered structure inherent in human everyday activities by analyzing the variability in the data. [79] represented each activity using partially ordered intervals and used Dynamic Bayesian Networks to represent the partial order. [29] extracted the storyline of a video using AND/OR graphs to represent the causal relationships among actions. [38] used co-occurrence graphs to represent the relations among low-level events. In our work we focus on the ordering among actions and propose to use directed graphs, where a relation between two nodes represents an action ordering but the relation defining the graph does not have to be antisymmetric as in the case of partial orders, allowing the inclusion of cycles.

4.3 Approach

Our end goal is to notify users when they forget to perform an important action before the ongoing one ends. In order to do this we need to: (1) recognize what the user has already done so far, (2) determine what the user is about to do, (3) identify the missing actions, (4) compute the corresponding cost of missing them, and (5) use this cost to generate reminders. In this framework, the first two tasks require performing action recognition and prediction together with segmentation, (3) and (4) require extracting inter-action dependencies, and (5) requires a notification decision mechanism.

Suppose we are given the segmentation of actions, and we have a function $F(t)$ that for each small window L before time t can give us the action category ($F(t) = a_j \in A = \{a_1, a_2, ..., a_M\}$, the set of all actions), and a function $\psi(t, F(t))$, that for each time t and the given action category

$F(t)$, provides the state of the progression of the action:

$$\psi(t, F(t)) = \begin{cases} -1 & if\ t\ \in B_{F(t)} \\ 0 & if\ t\ \in M_{F(t)} \\ 1 & if\ t\ \in E_{F(t)} \end{cases}$$

where B, M, E, represents the action beginning, middle and end states. Let t_+ be the next time step after t; $t_+ = t + L$. Then we can formulate a scoring function $\mathcal{Z}$ for the utility of issuing a reminder at time t_+ as:

$$\mathcal{Z}(t_+) = \Delta(t) * [\mathcal{C}(F(t_+), \mathcal{N}(F(t), F(t_+))) + \lambda * \alpha] \tag{4.1}$$
$$\Delta(t) = -\psi(t_+, F(t_+)) * \psi(t, F(t))$$

where $\mathcal{C}(a_j, a_q)$ is the cost of performing action a_q after a_j, $\mathcal{N}(a_i, a_j)$ returns the first missing action between the last completed action a_i and the predicted one a_j, $a_i, a_j, a_q \in A$. $\Delta(t)$ is a function that specifies the candidate times for issuing a reminder; these are the times a prediction is made after a completed action. α is a constant penalty for a reminder about an unnecessary action or for missing a required reminder, and λ is a trade off factor. The value of $\mathcal{Z}(t_+)$ at time t_+ determines if a notification should be given.

In order to obtain a list of missing actions and compute $\mathcal{N}$ and $\mathcal{C}$, we need to extract the dependencies between actions and calculate the cost of performing one action after another. We use action orderings to model the inter-action dependencies.

4.3.1 Extracting Dependencies Between Actions:

Every individual might follow a different order of actions while completing an activity. For example, one person might prefer to add milk after adding espresso, while the other might prefer to do the opposite. We represent the space of all dependencies between actions as a possibly cyclic, directed graph [2], which we call a *flexible ordered graph* [3].

[2]Different from partially ordered graphs, which do not allow cycles.

[3]To make the solution simpler, we do not allow self-loops although our model can be easily adjusted to handle them.

A flexible ordered graph is a directed graph $G = (V, E)$, in which the vertices V represent actions and the weighted directed edges E represent ordering constraints (Figure 4.2). The flexible ordered graph is transitive, possibly cyclic but not reflexive. Having cycles does not hurt the process but rather gives the flexibility to complete some action sequences multiple times. For example, a person might decide to add more milk after having added some.

Flexible Ordered Graph Construction: Assume that we have a set of videos, $S = \{S_1, S_2, ..., S_K\}$. To construct a flexible ordered graph, we use the *complete set of actions*, which is assumed to contain possible action orderings when the activity is completed *correctly*. Over this set, a flexible ordered graph is constructed by computing the transition probabilities, T, between every pair of actions (a_i, a_j). $T(a_i, a_j)$ shows the transition probability from action a_i to a_j. For each action pair (a_i, a_j), if a_j follows a_i, an edge $e_{ij} \in E$ is created from v_i to v_j; $v_i, v_j \in V$. The weight of each edge $w(v_i, v_j) = w(e_{ij}) = 1/T(a_i, a_j)$.

For the latte making activity, the flexible ordered graph has fixed entrance and exit states, namely, `entering the room` and `leaving the room`. An example flexible ordered graph is given in Figure 4.2.

Cost of performing one action after another ($\mathcal{C}$): We calculate the cost of taking action a_q after a_j, $\mathcal{C}(a_j, a_q)$ as the minimum weighted geodesic distance from a_j to a_q on the flexible ordered graph G:

$$\mathcal{C}_q^j = \mathcal{C}(a_j, a_q) = min_P(\sum_{e \in P} w(e))$$

where $P(a_j, a_q)$ is *any* path from a_j to a_q, and $e \in P$. The path corresponding to $\mathcal{C}(a_j, a_q)$ is represented as P_{jq}^*. Since the weights on the edges are always positive, taking circular paths repeatedly can never produce a value less than taking the direct path once.

Determining missing actions ($\mathcal{N}$): Assume that the most recently completed action is a_i, and our method predicted that the user is about to perform action a_j. We compute $\mathcal{C}(a_j, a_q)$, where a_q is the first action on the path P_{ij}^*. If the value of $\mathcal{C}_q^j$ is high enough, we report the actions on

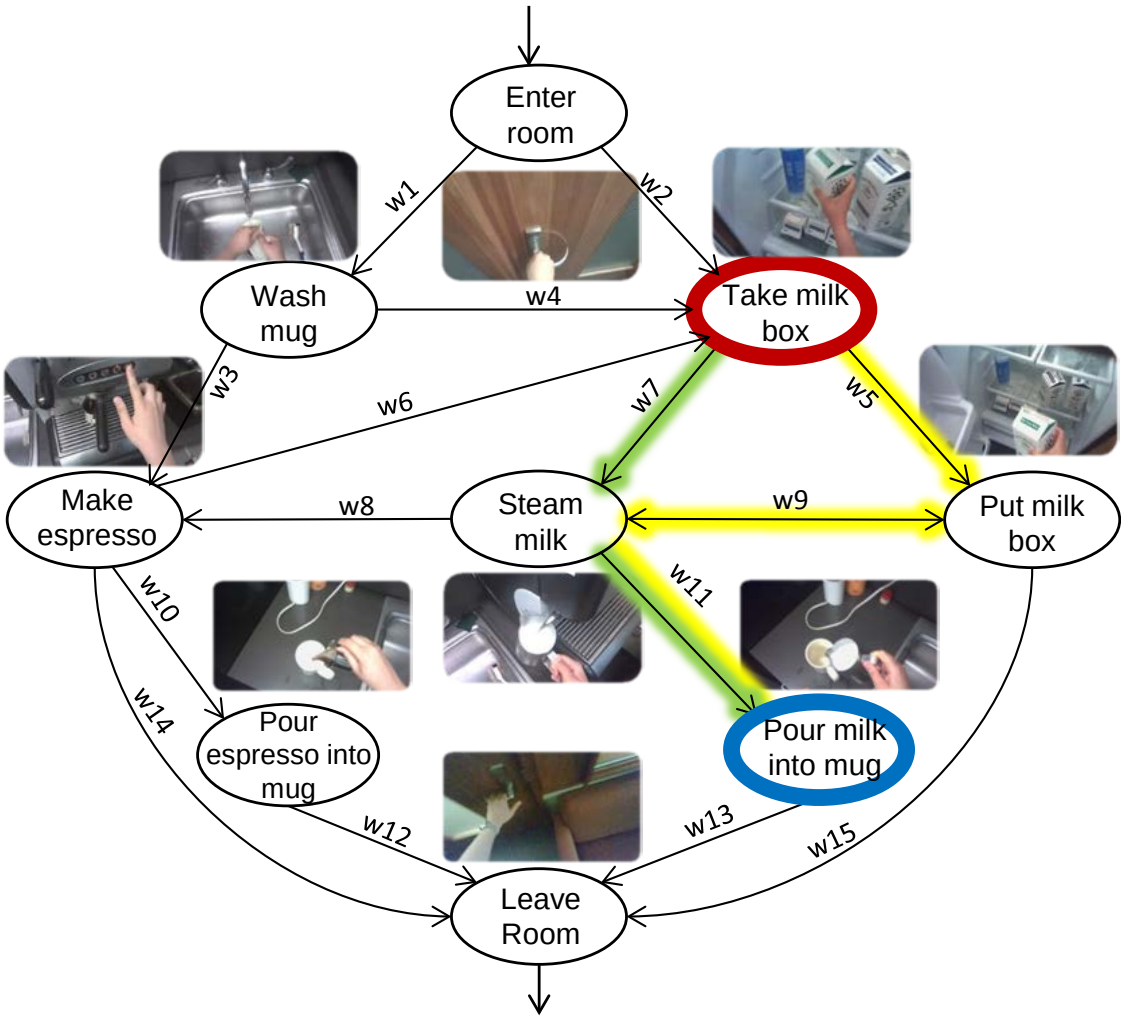


Figure 4.2: An example flexible ordered graph, where the most recent action is `take milk box`, marked in red, and the predicted action is `pour milk into mug`. The possible paths from the most recent action to the predicted one are marked in green and yellow.

P^*_{ij} as missing. For example in Figure 4.2, after `take milk box`, our system predicted that the user is about to `pour milk into mug`. The action `steam milk` is then reported as a missing action, because it is on the minimum cost path between `take milk box` and `pour milk into mug` and has a high cost of being missing.

4.3.2 Coupled Prediction and Segmentation Module

Calculation of $\mathcal{Z}$, in Equation 4.1, requires F and ψ, which in turn requires action segmentation, recognition and, prediction available. So far, these were assumed. In this section we describe a model for predicting what the user is about to do from a video stream. This requires knowing the boundary of the previous action and predicting the ongoing action from the very first few frames.

Segmentation Using Action Part Classifiers: For a streaming unsegmented video sample, we designed a discriminative Hidden Markov Model (HMM), which infers the past (recognition) and the current (prediction) action categories, while segmenting the activity sequence. In addition, it provides the current progress of the predicted action: beginning, end or middle.

In this model, the states are different action parts belonging to all action categories. These actions are connected according to their reachability. Actions parts can be in one of the three categories: action beginning (corresponding to the first L frames of an action), action end (corresponding to the last L frames of an action) and action middle (corresponding to each L consecutive frames between the action beginning and end parts).

For a particular action a_i, the probability of reaching to action middle (M_{a_i}) and action end (E_{a_i}) states from the action beginning (B_{a_i}) state depends on the length of the action. In addition to in-class reachability, the end state of every action class can reach to the beginning states of other action classes according to the transition probabilities obtained from the full training dataset, which consists of both complete and incomplete training samples. Figure 4.3 shows the states and the allowable transitions in the model, where the observation probabilities come from the action beginning, middle and end classifiers described in the next section.

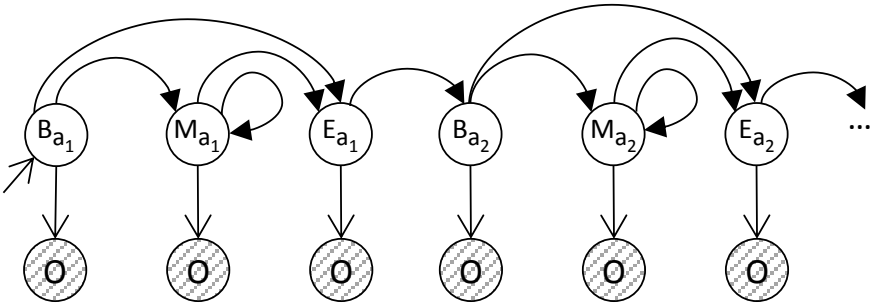


Figure 4.3: Our coupled HMM model for segmentation and prediction. The states are beginning, middle and end, and the observations come from action part classifiers.

Action Part Classifiers: For training *action part* classifiers, we first partition the action samples in the training data into parts: beginning, middle and end. Figure 4.4 shows this partition for different action part classifiers. We only use the first L frames of actions for training action beginning classifiers. For training the action end classifiers, we use the last L frames of each action, and we use all remaining L frames in each action as a separate sample for training the action middle classifiers. We use L2 regularized logistic regression for the training of all action part classifiers.

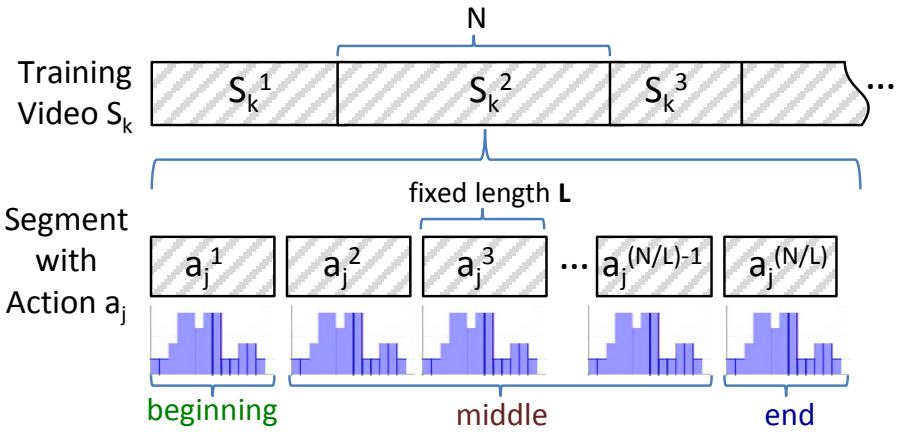


Figure 4.4: Different parts of training data are used for training different action part classifiers: beginning, middle, end.

Inference: During the inference process, we need to solve recognition, segmentation and prediction at the same time. A given input video S_k^{test} is divided into non-overlapping parts of length L and fed into the part classifiers. Given our model with the observations from the part classifiers, this problem is solved using the Viterbi algorithm. This procedure not only finds the action boundaries and recognizes past actions, but also predicts the ongoing action from partial observations. Figure 4.5 illustrates this procedure.

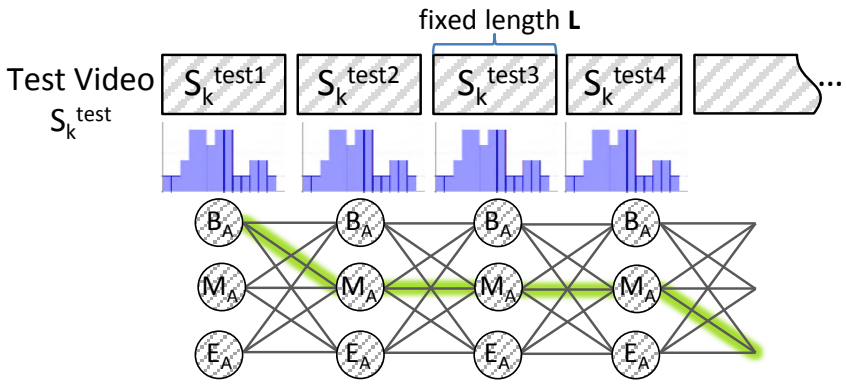


Figure 4.5: Inference: Each test video is divided into fixed length parts and feed to part classifiers. In the Viterbi trellis B_A, M_A, E_A nodes represent 29 class action beginning, middle and end classifiers, respectively, and every edge represents 29x29 connections. An example output path is highlighted in green.

4.4 Experiments

Our purpose is to give proper reminders to users when they forget to perform some actions. For this purpose we design a model, which has two main functionalities: (1) segmenting, recognizing previous and predicting the current action at the same time, (2) making a notification decision based on the segmentation, recognition and prediction.

Espresso Room Dataset: To the best of our knowledge, there is no existing dataset, which is designed for notification purposes. Egocentric settings are more suitable for understanding daily

Figure 4.6: Example frames from the egocentric dataset of latte making activity.

activities as the camera can move around with the person. For this reason, we collected a new dataset using a Looxcie HD camera mounted on a cap, which can record 1920x1080 resolution in $\sim 30fps$. We have recorded 41 videos of about 20 subjects, some of whom made lattes multiple times in different days. The total number of frames in the dataset is $344,173$. During data collection, all subjects were asked to make lattes according to their own preferences; therefore the action order and duration, and the actions in each video vary. Our only request to the subjects was to look at the action of interest while performing the action, in order to avoid irrelevant frames. However, in many cases the subjects still looked around, making the dataset more challenging. Another challenge in the dataset is simultaneous actions; we noticed that people tend to perform multiple actions at the same time. For instance, while making espresso, they may froth milk at the same time. Simultaneous action recognition/prediction is out of scope of this research, and we labeled those parts of videos with only one of the simultaneous actions. Example frames from the collected dataset is given in Figure 4.6, and the full list of actions is given in Table 4.1. In our experiments, we downscaled the videos to 480x270 resolution and sampled one out of every two frames, resulting in a total of 108,159 frames after irrelevant ones were removed.

clean portafilter	clean espresso pitcher
clean nozzle	clean milk pitcher
wash mug	close fridge
flatten milk box	froth milk
enter room	grind coffee
leave room	make espresso
open fridge	pour coffee into mug
take coffee portafilter	pour milk into mug
pour milk into pitcher	put jacket
put keys	put milk box back to fridge
throw milk box	put milk pitcher
take jacket	take keys
take milk box	take milk pitcher
turn off lights	turn on lights
stir	

Table 4.1: Actions of the latte making activity.

Metrics: We use a variety of metrics to evaluate the proposed notification system, and to analyze different parts and alternative approaches. Our goal is to be as precise as possible for each user. Therefore, our performance numbers are the average over the scores of each sample unless otherwise stated. Moreover, since recognizing each action is very important for giving proper notifications, we believe average per class accuracy is more important than the average accuracy for the evaluation purposes (despite the fact that our model produces higher results for average accuracy). Note that, this is a multi-class problem and accuracy is calculated over all classes simultaneously.

Experimental Setup: In all of our experiments we use 24 complete samples, where there is no missing action as our fixed part of training data. We then use 17 incomplete samples with missing actions for testing in a leave-one-out-cross-validation fashion. We use our training data such that every sample of an action has exactly one action beginning part, exactly one action end part, and zero or more action middle parts, whose lengths are equal to 10 frames. During the inference time, a test video sequence is divided into fixed length parts of 10 frames.

Features: The visual data obtained from egocentric videos are drastically different from the data obtained from static cameras. Although the high camera movement causes jitter and requires camera motion handling and perspective distortions, there is an advantage to the egocentric cameras: the scene changes with the action as the camera moves. Therefore, the scene-based representations such as GIST [66] features are useful. In our experiments we use GIST as our feature representation and extend it to video part representation using the bag of words (BoW) paradigm, with 500 words. To show the benefit of GIST, we also experiment with space-time-interest-point features (STIP) [46]. For comparison purposes, we use segmented videos and represent each action using BoW with a dictionary size of 500 words. Table 4.2 shows comparisons between GIST and STIP in action recognition on our dataset using discriminative HMM. Dense Motion Trajectory [89] features in general are not suitable in egocentric settings because of continuous drastic camera movement.

	Avg. Acc.	Avg. per Class Acc.
STIP	57.70	56.56
GIST	70.14	68.61

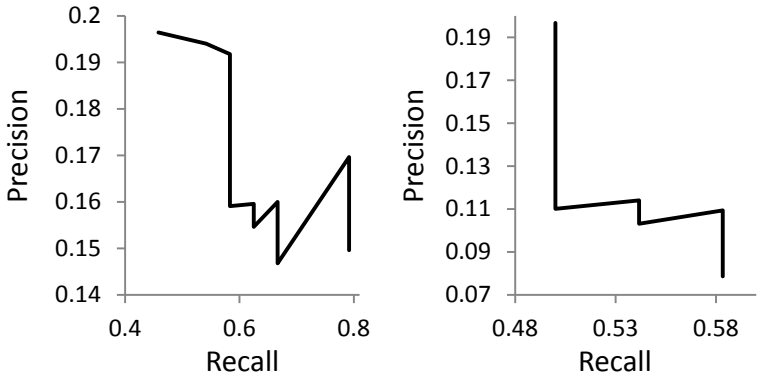
Table 4.2: Accuracies of the STIP vs the GIST features in recognizing actions using a Discriminative HMM.

4.4.1 Evaluation

We evaluated the proposed system in two tasks: 1) evaluation of the notification module. 2) evaluation of the prediction and segmentation model.

Notification Module: Our system is designed to give appropriate notifications, while the users continue their activities. Since the activity of latte making always ends with the `leave room` action, we also check for the required actions that are not completed until a `leave room` action is detected. Some of these reminders can be previously given when a missing is action detected. The cost of any required action after performing the `leave room` action is set to infinity, always resulting in a notification when it is missed. Required actions are defined in two parts; the first part consists of the minimum set of actions, $A^{min} \subset A$, that are manually determined as required for latte making activity to be complete, and the second part is updated online at every new prediction, according to automatically calculated co-occurrences with other actions. (For example, an `open fridge` action always co-occurs with a `close fridge` action. When an `open fridge` action is detected, a `close fridge` action is added to the set of required actions.). For the calculation of co-occurrences, we use the complete set defined above. When a required action is completed, we remove it from the set.

The accuracy of our notification system can be measured in 3 ways: 1) Existence of a reminder: We measure if we give a reminder for the samples that need one regardless of time and action type. Our system gives a notification whenever it is required, precision and recall are equal to 1 for all



(a) Alarm correctness (b) Alarm correctness & timeliness

Figure 4.7: Evaluation of the notification module: (a) shows the precision-recall curve for notification correctness evaluation regardless of time. (b) shows the precision-recall curve for giving the correct notification at the right time.

samples. 2) Correctness of the type of the reminder: We measure if our method give the proper notification regardless of timeliness. 3) Timeliness of reminder: We measure both the correctness and timeliness of the notification. The timeliness of a notification is measured in a δ neighborhood. This means that if a reminder is given in the δ neighborhood of the groundtruth time, it is counted as correct. In our experiments, δ is set as 15 frames (1.5 parts, when $L = 10$). Figure 4.7 represents the precision-recall curves for the last two evaluations, when $\alpha = 0.5$, $\lambda = 1$. Note that, issuing the correct and timely reminders is extremely challenging. In fact, the chance probability of having the correct reminder at the right time t_+ is equal to $1/(\frac{\#frames}{L})^{(M+1)}$, where L is equal to the length of a part, and we make a decision after seeing every part. In a 30 class case (including *no reminder* class), with $\#frames$ about $108K$ and $L = 10$, this number reduces to $\frac{1}{10{,}800^{30}}$. If we incorporate δ to this equation, it roughly becomes $\frac{2^{(2*1.5)}}{10{,}800^{30}} = \frac{8}{10{,}800^{30}}$, still a very small chance.

Segmentation and Prediction Model: Table 4.3 shows the performance of our model when the whole sequence is already observed using different metrics, both in partwise and framewise. When we evaluated partwise, we assigned the groundtruth label of a part with max voting. The first col-

	Avg. Over Samples	Overall Avg.
Accuracy	72.3	73.56
Avg. per Class Acc.	63.4	64.36
Framewise Acc.	71.84	73.13
Framewise Avg. per Class Acc.	61.97	63.5

Table 4.3: Accuracies of our prediction and segmentation model when L=10 and the whole sequence is observed.

umn shows the average accuracy of all samples. The second column shows the accuracy calculated by concatenating the recognition labels and comparing with the concatenated groundtruth. This evaluation is biased towards the results of the longer sequences. The first and second rows give average accuracy and average per class accuracy respectively, over parts. The third and fourth rows give the framewise average accuracy and framewise average per class accuracy, respectively.

Baselines: Tables 4.4, 4.5, 4.6 show different baselines. To explore our model in different settings, we first calculate the prediction accuracy. For this purpose, we use the groundtruth locations of action beginning states for deciding prediction times. Figure 4.8 shows an example scenario, in which the prediction times are represented by arrows, brackets show groundtruth action boundaries, and the red boxes are the parts on which a prediction is made. The first line of Table 4.4 shows the results of the action beginning classifier, which does not use temporal information. The second line shows the results when we use an ordinary discriminative HMM, where the observations come from the action beginning classifier. In both of these experiments the segmentation is provided.

Row 3 of Table 4.4 shows the results for coupled prediction and segmentation. Prediction times are the same as the first two experiments and the red boxes in Figure 4.8 show the corresponding

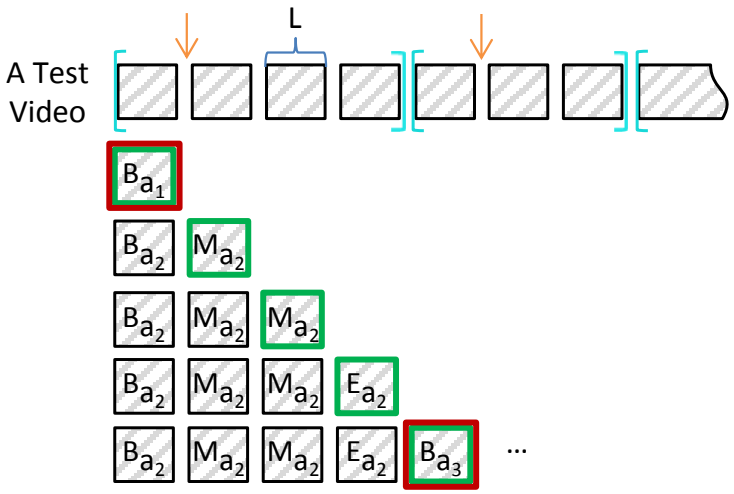


Figure 4.8: Prediction is evaluated on the action beginning parts, shown by red boxes. Online prediction evaluation is done over the predicted labels of parts shown by green boxes. The labels in the boxes show the predictions at each time step. Arrows show the times prediction is evaluated. Brackets show groundtruth action boundaries.

parts. Row 4 of Table 4.4 shows the results of online prediction. We measure the online prediction accuracies for the labels predicted at each time step. As more frames are observed, the earlier predictions might be updated, but the accuracies are computed only on the current predictions; the green boxes in Figure 4.8 show the corresponding parts. This is the evaluation suitable for a real life scenario, since we make decisions as we observe frames. As another experiment, we use the whole sequence to recognize and segment actions. The last row in Table 4.4 shows the corresponding results.

We evaluated our part classifiers to see how well each represented its class. Table 4.5 show their accuracies, when we assume the segmentation is given and the test input is from the related part of the action. The values in the last row are equal to the action prediction accuracies, when the segmentation is given and temporal information is not used (Table 4.4, row 1).

Our model is designed to predict actions for unsegmented videos. We compared our model with max-margin early event detectors (MMED) [32, 34], which is a modified SOSVM model, detecting the actions in an unsegmented video before they complete. At each step [32] detects

	Avg. Acc.	Avg. per Class Acc.
Prediction, not temporal *(segmentation is given)*	51.07	48.26
Prediction, temporal *(segmentation is given)*	71.49	69.38
Prediction, temporal *(segmentation is unavailable)*	45.87	43.42
Online Prediction, temporal *(segmentation is unavailable)*	67.87	56.2
Recognition, temporal *(segmentation is unavailable)*	72.3	63.4

Table 4.4: Accuracies of prediction models, in different settings. For all models, the prediction scores represent when only the first L=10 frames of actions are observed. Online prediction scores use the predictions after every part of L=10 frames.

	Avg. Acc.	Avg. per Class Acc.
Action End Classifier	48.81	46.17
Action Middle Classifier	58.83	46.73
Action Beginning Classifier (Prediction)	51.07	48.26

Table 4.5: Accuracies of Part Classifiers, when the segmentation is given, L=10.

	Precision	Recall	F measure
MMED	0.33	0.25	0.25
Our Model	0.61	0.57	0.57

Table 4.6: Our model vs MMED. We use L=10 when evaluating our model.

only the first instance of an action. In order to detect multiple instances of the same class in a sample, we input the remaining video after every detection. We used the same features as our method. The major drawback of the early event detection approach over coupled prediction and segmentation is that every detection of action is done separately for each class; therefore, there can be overlapping frames in the detected action locations. We measured the framewise F value of each class, and took the average over classes, then compared with that of our method. Our method showed superior performance in these experiments. The results are given in Table 4.6.

4.5 Discussion

This chapter gives detailed results for the experiments with our model. Figure 4.9 and Figure 4.10 show recognition and online prediction results respectively, of our model when the sequence is also automatically segmented simultaneously. Figure 4.11 and Figure 4.12 show the accuracies of our model in different settings. Finally, Figure 4.13 and Figure 4.14 represent the confusion matrices regarding to recognition and online prediction.

Figure 4.9 shows the recognition and segmentation results of our model when L=10, and the whole sequence is observed. The first row of each sample represents the groundtruth segmentation, and the second row shows the recognition and segmentation. The Y axis shows the sample ID's, and the X axis shows the time as frame numbers. This figure also shows that there is a quite large variance in the sample length. However, our model always predicts actions from a fixed number of frames at the action beginning (unlike some other approaches that use a percentage of cropped video.) This makes our model's prediction quality independent of the duration of the action. This property can be seen in Figure 4.9, as the recognition of the action category is not affected by the

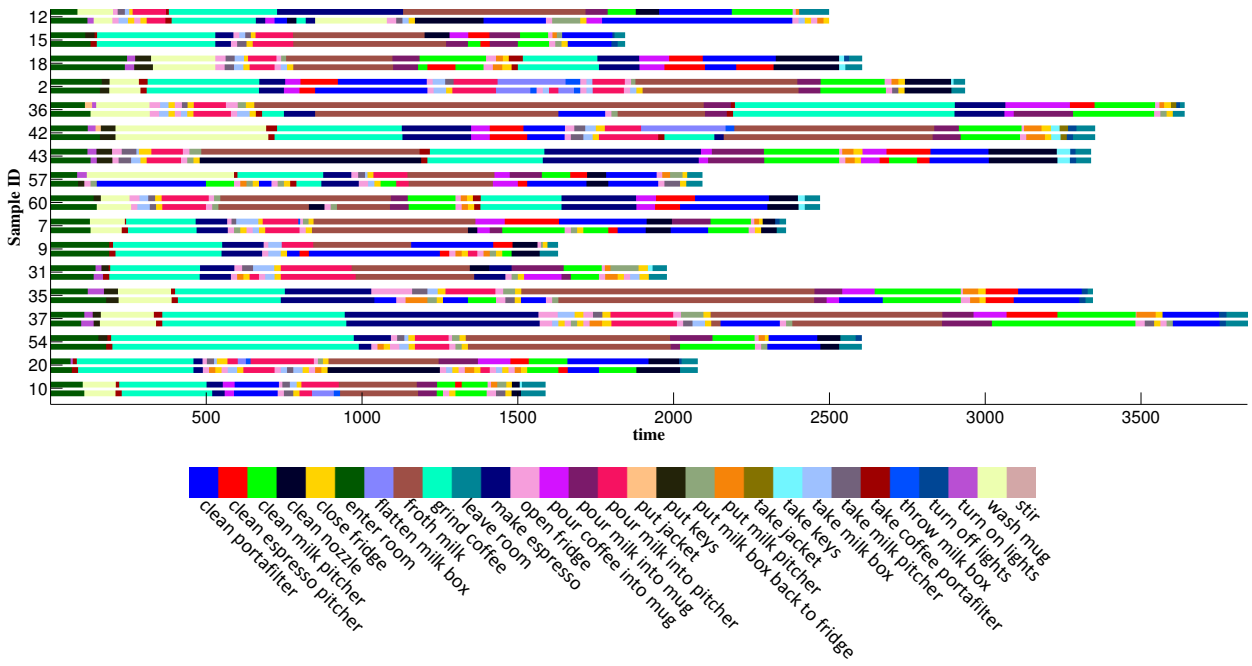


Figure 4.9: Recognition and segmentation results of our model. L=10. The whole sequence is observed.

action length. (For example compare the `enter room` action for samples 10 and 18.)

Figure 4.10 shows the online prediction and segmentation results of our model when L=10, and a prediction is made at every time step. This is the suitable setting for a real time system, since the time step is negligibly small. Compared to the results when the whole sequence is observed (see Figure 4.9), the prediction and segmentation results have more fluctuations (small chunks of frames are mis-segmented/mis-predicted).

Figure 4.11 shows the per sample accuracies of our model in recognition and segmentation when L=10, and the whole sequence is observed. Here we can see that the framewise and partwise accuracies are very close to each other because of the small value of L. When we measure partwise accuracies, we use the label with the highest frequency in a part as the groundtruth part label.

Figure 4.12 shows the per sample accuracies for online prediction when L=10, and a prediction is made at every time step. Here we can see that although the accuracies are in general lower than

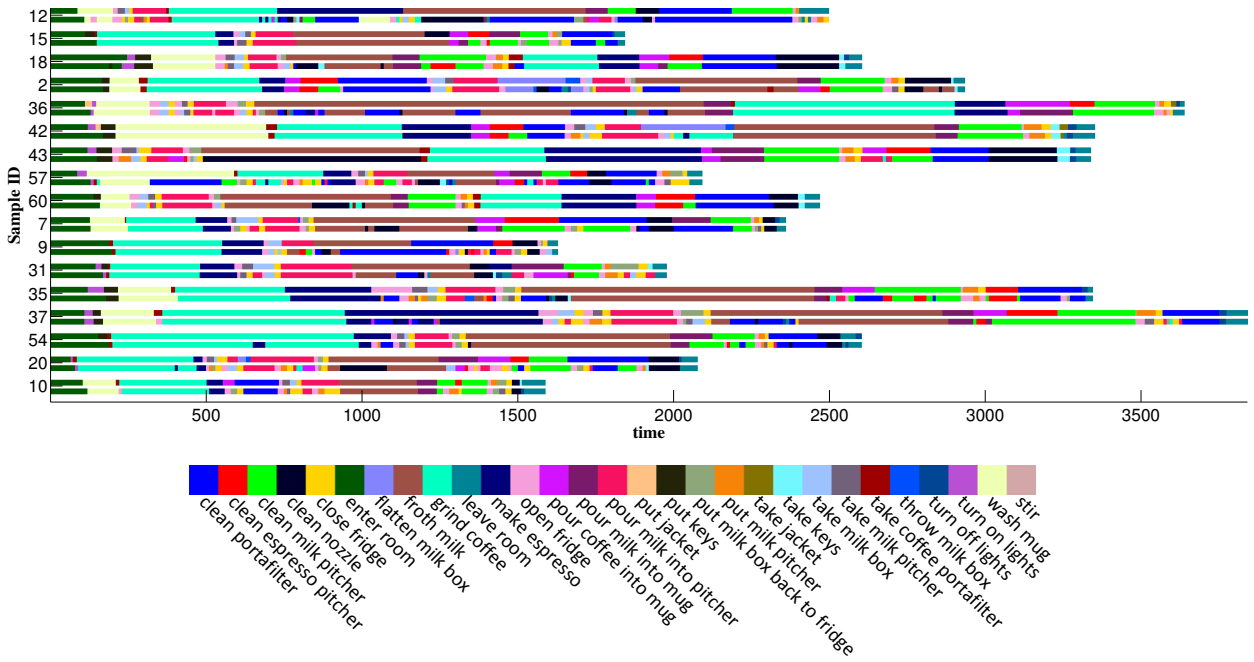


Figure 4.10: Online prediction and segmentation results of our model. L=10.

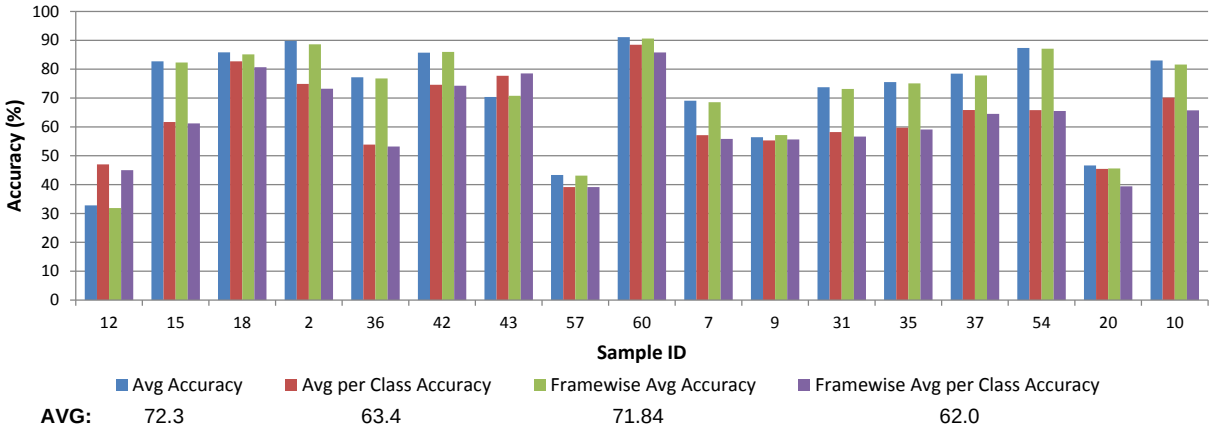


Figure 4.11: Per sample accuracies for recognition and segmentation, L=10.

the case when the whole sequence is observed, the difference is very minor, showing the power of our model in making predictions on incomplete sequences.

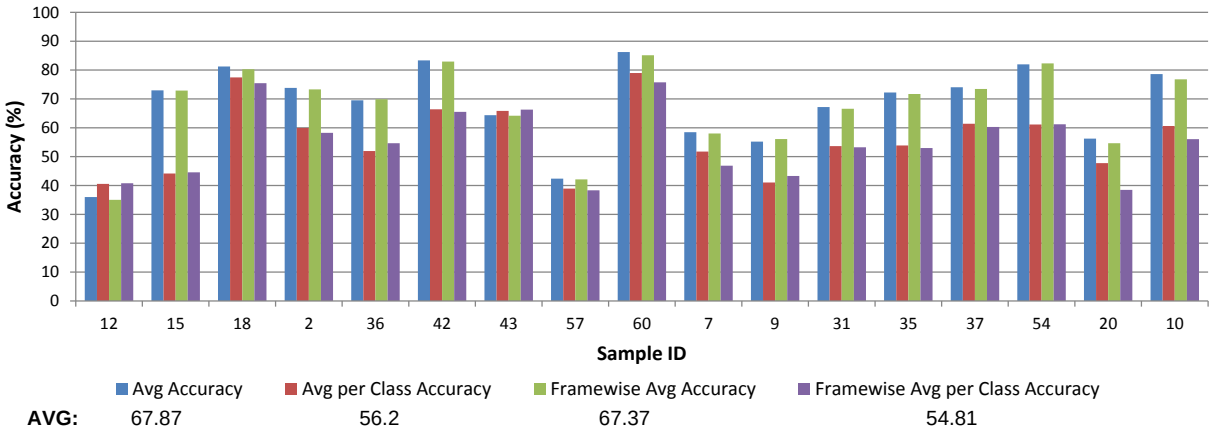


Figure 4.12: Per sample accuracies for online prediction and segmentation, L=10.

Figure 4.13 shows the confusion matrix for recognition and segmentation when L=10, and the whole sequence is observed. The highest confusion is observed between the actions `put jacket` and `enter room`, which are usually consecutive actions in the latte making activity sequences of our dataset.

Figure 4.14 shows the confusion matrix for online prediction and segmentation when L=10, and a prediction is made at every time step. Similar to Figure 4.13, the highest confusion is observed between the actions `put jacket` and `enter room`, but another high confusion is also observed between `turn on lights` and `enter room`, which are also very common consecutive actions in our dataset. When the recognition is made after the whole sequence is observed, the mispredictions in `turn on lights` that exist in online prediction are fixed by the Viterbi algorithm resulting in a smaller confusion in Figure 4.13.

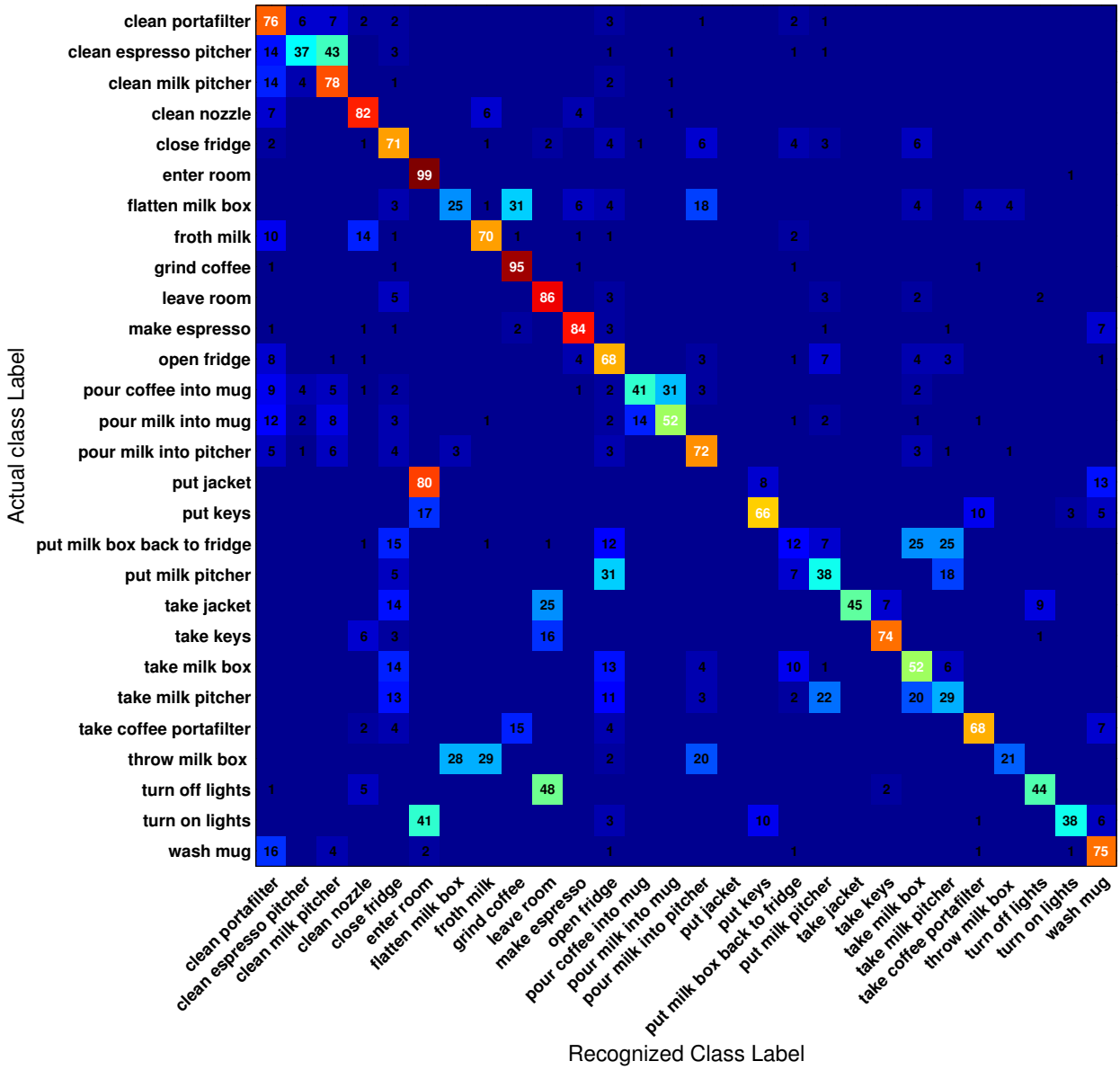


Figure 4.13: Confusion matrix for recognition and segmentation, L=10. Framewise, overall, average accuracy is 73.13%.

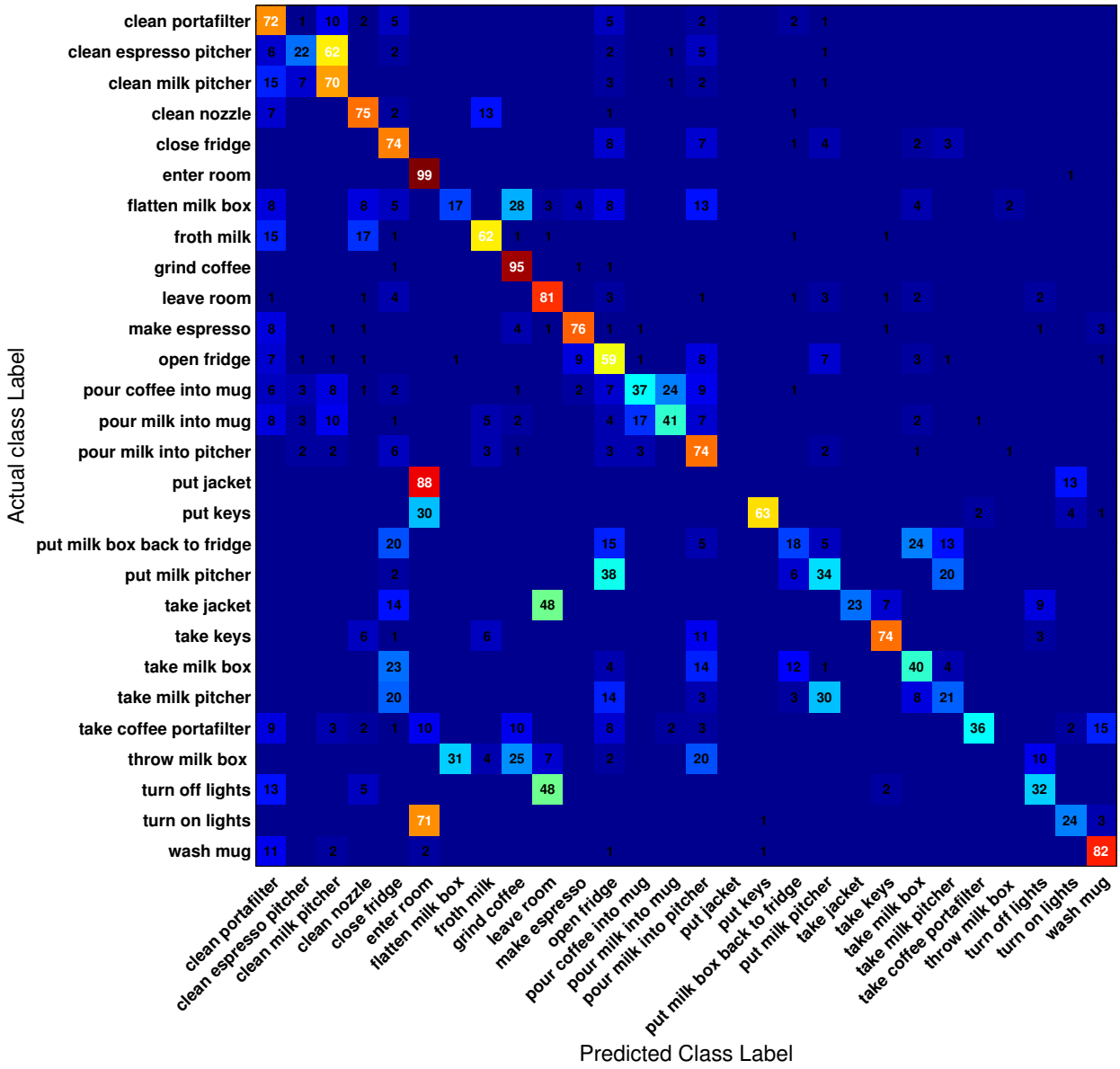


Figure 4.14: Confusion matrix for online prediction and segmentation, L=10. Framewise, overall, average accuracy is 68.39%.

4.6 Summary

Prospective memory failures exist in all of our lives. Forgetting an action might cause serious and expensive consequences. In this chapter, we introduced a framework that issues notifications on the actions that may be missed during a sequence of an activity. To show a proof of concept, we collected an egocentric dataset of people making lattes and showed that our method can produce action reminders. We evaluated our notification model by means of correctness and timeliness and showed promising results. To further analyze our model, we also evaluated our recognition, prediction and segmentation models and showed comparisons to state-of-the-art approaches.

Producing accurate and timely reminders is an important but extremely challenging problem and encourages new research directions. To reason about missing actions and their associated costs, we used temporal dependencies between actions represented by our flexible ordered graph. Our approach does not take into account semantics of dependencies between activities in terms of causalities, preconditions and effect. Future work involves deeper understanding of inter-action dependencies and discovering these challenging dependencies. Furthermore, producing action reminders requires solving segmentation, recognition, and prediction at the same time. We proposed one solution to this problem that couples these tasks at a higher level. Tighter coupling at lower levels is another promising direction to be explored. Finally, generating action reminders requires a complex decision making component. Our solution formulates this problem with a scoring function that trades off between the cost of missing action and the penalty for issuing a reminder. Devising suitable reward functions for more complex decision mechanisms (such as MDPs) that can take into account future possibilities is the next big step.

Chapter 5

CONCLUSION

The purpose of this research was to explore the possible ways of improving people's lives using information from static or egocentric cameras. In this dissertation, we described multiple methods for recognizing human actions in different scenarios and evaluated their performance. The first method was designed for recognizing hand tremors from videos recorded by a static camera. For this purpose a new dataset was collected and the developed system achieved 95.4% accuracy in LOOCV over this dataset of 90 simulated videos and 83 normal cases.

The second method used an egocentric and multiple static cameras to understand the actions of people while they are making brownies, using videos of the CMU-MMAC dataset. The system fuses information from both modalities according to the information they carry at a particular moment in time. Our model achieved 54.62% of accuracy, while the random chance is 3.5% (1/28). We also compared our method with state-of-the-art methods and our method showed a much better performance. Moreover, we tested the same system on a multiple static camera setting of IXMAS and EPFL-IXMAS datasets, while people are performing other actions like kicking, punching, checking watch, crossing arms, scratching head. Our method outperformed the state-of the-art on the EPFL-IXMAS dataset, while showing an accuracy on par with the state-of-the-art on the IXMAS dataset.

Finally, we introduced a missing action notification problem, and proposed a solution and to show a proof of concept we collected a new dataset using an egocentric camera. The proposed solution gives necessary reminders to the user when the user forgets to perform an action during the course of a latte-making activity. In order to give proper on time notifications, our solution extracts possible action orderings, predicts the actions before they complete, recognizes the past activities while segmenting the observed sequence. The proposed model makes a prediction when

only the very first few frames of an action is observed (10 frames = 2/3 of sec). While the overall prediction accuracy is 45.9%, the overall recognition and segmentation accuracy is shown to be 72.3% when the whole activity sequence is observed. Moreover, the online prediction and segmentation accuracy is 67.9% when the prediction is made at every observation time step. Finally, the proposed model shows promising results on the extremely challenging task of issuing correct and timely reminders.

This work has shown the utility of camera-based monitoring for medical or life-assisting applications. Future work involves investigating the benefit of multi-modal camera systems composed of different type of cameras in improving the timely notification accuracy and action understanding (prediction/recognition/forecasting). Moreover, improving the timely notification accuracy on egocentric cameras alone, by improving on both the discovery of the inter-action dependencies, and early prediction of actions with simultaneous segmentation and recognition is also another research direction. Finally, we foresee that both notification decision mechanism and action prediction can benefit from other information such as action forecasting estimations. Static or depth cameras can also be used for generating general reminders to the audience visible, but this might require understanding of simultaneous actions of multiple people in order to be suitable for real-world scenarios.

BIBLIOGRAPHY

[1] A. Ghali, et al. Object and event recognition for stroke rehabilitation. In *VCIP*, 2003.

[2] J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Comput. Surv.*, 2011.

[3] Pradeep K. Atrey, M. Anwar Hossain, Abdulmotaleb El-Saddik, and Mohan S. Kankanhalli. Multimodal fusion for multimedia analysis: a survey. *Multimedia Syst.*, 2010.

[4] William H. Bares, Somying Thainimit, and Scott Mcdermott. A model for constraint-based camera planning. In *In Smart Graphics. Papers from the 2000 AAAI Spring Symposium*, 2000.

[5] G. Bradski. The OpenCV Library, 2000.

[6] Matthew Brand and Vera Kettnaker. Discovery and Segmentation of Activities in Video. *PAMI*, 2000.

[7] W. Brendel and S. Todorovic. Learning spatiotemporal graphs of human activities. In *ICCV*, 2011.

[8] Y. Cao, D. Barrett, A. Barbu, N. Siddharth, H. Yu, A. Michaux, Y. Lin, S. Dickinson, J. M. Siskind, and S Wang. Recognizing human activities from partially observed videos. In *CVPR*, 2013.

[9] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2011.

[10] Fu Chang, Chun-Jen Chen, and Chi-Jen Lu. A linear-time component-labeling algorithm using contour tracing technique. *Comput. Vis. Image Underst.*, 2004.

[11] Fernando De la Torre, Jessica Hodgins, Javier Montano, Sergio Valcarcel, and Justin Macey. Guide to the carnegie mellon university multimodal activity (cmu-mmac) database. Technical report, CMU, RI, 2009.

[12] R. Key Dismukes. Prospective Memory in Workplace and Everyday Situations. *Curr. Dir. Psychol.*, 2012.

[13] Chris Ellis, Syed Zain Masood, Marshall F. Tappen, Joseph J. Laviola, Jr., and Rahul Sukthankar. Exploring the trade-off between accuracy and observational latency in action recognition. *IJCV*, 2013.

[14] David K. Elson and Mark O. Riedl. A lightweight intelligent virtual cinematography system for machinima production. In *AIIDE*, 2007.

[15] A. Farhadi, D. Forsyth, and R. White. Transfer learning in sign language. In *CVPR*, 2007.

[16] Ali Farhadi and Mostafa Kamali Tabrizi. Learning to recognize activities from the wrong view point. In *ECCV*, 2008.

[17] Ali Farhadi, Mostafa Kamali Tabrizi, Ian Endres, and David A. Forsyth. A latent model of discriminative aspect. In *ICCV*, 2009.

[18] A. Fathi, Xiaofeng Ren, and J.M. Rehg. Learning to recognize objects in egocentric activities. In *CVPR*, 2011.

[19] Alireza Fathi, Ali Farhadi, and James M. Rehg. Understanding egocentric activities. In *ICCV*, 2011.

[20] Alireza Fathi, Jessica K. Hodgins, and James M. Rehg. Social interactions: A first-person perspective. In *CVPR*, 2012.

[21] Alireza Fathi, Yin Li, and James M. Rehg. Learning to recognize daily actions using gaze. In *ECCV*, 2012.

[22] Alireza Fathi, Yin Li, and James M. Rehg. Learning to recognize daily actions using gaze. 2012.

[23] Alireza Fathi and James M. Rehg. Modeling actions through state changes. In *CVPR*, 2013.

[24] Usama M. Fayyad and Keki B. Irani. Multi-interval discretization of continuous valued attributes for classification learning. In *13th IJCAI*, 1993.

[25] R. Fisher and P. Reddy. Supervised multi-modalaction classification. In *Technical report, Carnegie MellonUniversity*, 2011.

[26] E. Fox, E.B. Sudderth, M.I. Jordan, and A. Willsky. Bayesian nonparametric inference of switching dynamic linear models. *IEEE Signal Processing*, 2011.

[27] Nikolaos Gkalelis, Hansung Kim, Adrian Hilton, Nikos Nikolaidis, and Ioannis Pitas. The i3dpost multi-view and 3d human action/interaction database. In *CVMP*, 2009.

[28] Chunhui Gu and Xiaofeng Ren. Discriminative mixture-of-templates for viewpoint classification. In *ECCV*, 2010.

[29] A. Gupta, P. Srinivasan, Jianbo Shi, and L.S. Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009.

[30] Li-wei He, Michael F. Cohen, and David H. Salesin. The virtual cinematographer: a paradigm for automatic real-time camera control and directing. In *Computer graphics and interactive techniques*, 1996.

[31] Julie D Henry, Peter G Rendell, Louise H Phillips, Leigh Dunlop, and Matthias Kliegel. Prospective memory reminders: A laboratory investigation of initiation source and age effects. *The Quarterly Journal of Experimental Psychology*, 2012.

[32] Minh Hoai and Fernando De la Torre. Max-margin early event detectors. In *CVPR*, 2012.

[33] Minh Hoai and Fernando De la Torre. Maximum margin temporal clustering. In *AISTATS*, 2012.

[34] Minh Hoai and Fernando De la Torre. Max-margin early event detectors. *IJCV*, 2014.

[35] Minh Hoai, Zhen-Zhong Lan, and Fernando De la Torre. Joint segmentation and classification of human actions in video. In *CVPR*, 2011.

[36] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A Practical Guide to Support Vector Classification, 2000.

[37] De-An Huang and Kris M Kitani. Action-reaction: Forecasting the dynamics of human interaction. In *ECCV*, 2014.

[38] Hamid Izadinia and Mubarak Shah. Recognizing complex events using large margin joint low-level event model. In *ECCV*, 2012.

[39] I.N. Junejo, E. Dexter, I. Laptev, and P. Perez. View-independent action recognition from temporal self-similarities. *PAMI*, 2011.

[40] K. Cuppens, et al. Detection of epileptic seizures using video data. In *Proc of the 6th Intern Conf on IE*, 2010.

[41] Takeo Kanade and Martial Hebert. First-person vision. *Proceedings of the IEEE*, 2012.

[42] Kris M Kitani, Brian D. Ziebart, J. Andrew (Drew) Bagnell, and Martial Hebert. Activity forecasting. In *ECCV*, 2012.

[43] Kris Makoto Kitani. Ego-action analysis for first-person sports videos. *IEEE Pervasive Computing*, 2012.

[44] Yu Kong, Dmitry Kit, and Yun Fu. A discriminative model with multiple temporal scales for action prediction. In *ECCV*, 2014.

[45] Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. A hierarchical representation for future action prediction. In *ECCV*, 2014.

[46] Ivan Laptev. On space-time interest points. *IJCV*, 2005.

[47] Benjamin Laxton, Jongwoo Lim, and David Kriegman. Leveraging temporal, contextual and ordering constraints for recognizing complex activities in video. In *CVPR*, 2007.

[48] Yong Jae Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012.

[49] Kang Li and Yun Fu. Prediction of human activity by discovering temporal sequence patterns. *PAMI*, 2014.

[50] Kang Li, Jie Hu, and Yun Fu. Modeling complex temporal composition of actionlets for activity prediction. In *ECCV*, 2012.

[51] Yin Li Li, Alireza Fathi, and James M. Rehg. Learning to predict gaze in egocentric video. In *ICCV*, 2013.

[52] Edirlei E. Soares de Lima, Cesar T. Pozzer, Marcos C. d'Ornellas, Angelo E. M. Ciarlini, Bruno Feijó, and Antonio L. Furtado. Support vector machines for cinematography real-time camera control in storytelling environments. In *Symposium on Games and Digital Entertainment*, 2009.

[53] Christophe Lino, Marc Christie, Roberto Ranon, and William Bares. A smart assistant for shooting virtual cinematography with motion-tracked cameras. In *ACMM*, 2011.

[54] Jingen Liu, M. Shah, B. Kuipers, and S. Savarese. Cross-view action recognition via view knowledge transfer. In *CVPR*, 2011.

[55] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *7th IJCAI*, 1981.

[56] Behrooz Mahasseni and Sinisa Todorovic. Latent multitask learning for view-invariant action recognition. In *ICCV*, 2013.

[57] Corey McCall, Kishore K. Reddy, and Mubarak Shah. Macro-class selection for hierarchical k-nn classification of inertial sensor data. In *PECCS*, 2012.

[58] Tomas McCandless and Kristen Grauman. Object-centric spatio-temporal pyramids for egocentric activity recognition. In *BMVC*, 2013.

[59] Sushmita Mitra and Tinku Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man and Cybernetics*, 2007.

[60] L. Morency, A. Quattoni, and T. Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *CVPR*, 2007.

[61] TAISUKE MORITA. Reminders supporting spontaneous remembering in prospective memory tasks1. *Japanese Psychological Research*, 2006.

[62] G. Murthy and R. Jadon. A review of vision based hand gestures recognition. *IJITKM*, 2009.

[63] Juan Carlos Niebles, Chih-Wei Chen, and Li Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV*, 2010.

[64] Definition of Latte. http://www.oxforddictionaries.com/definition/english/latte.

[65] Keisuke Ogaki, Kris Makoto Kitani, Yusuke Sugano, and Yoichi Sato. Coupling eye-motion and ego-motion features for first-person activity recognition. In *CVPR Workshops*, 2012.

[66] Aude Oliva and Antonio Torralba. Building the gist of a scene: the role of global image features in recognition. In *Progress in Brain Research*, 2006.

[67] P. Garg, et al. Vision based hand gesture recognition. *Engineering and Technology*, 2009.

[68] Selen Pehlivan and Pinar Duygulu. A new pose-based representation for recognizing actions from multiple cameras. *CVIU*, 2011.

[69] H. Pirsiavash and D. Ramanan. Parsing videos of actions with segmental grammars. In *CVPR*, 2014.

[70] Hamed Pirsiavash and Deva Ramanan. Detecting activities of daily living in first-person camera views. In *CVPR*, 2012.

[71] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 2010.

[72] M. Raptis and L. Sigal. Poselet key-framing: A model for human activity recognition. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 2013.

[73] Xiaofeng Ren and Chunhui Gu. Figure-ground segmentation improves handled object recognition in egocentric video. In *CVPR*, 2010.

[74] Xiaofeng Ren and M. Philipose. Egocentric recognition of handled objects: Benchmark and analysis. 2009.

[75] M. S. Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *ICCV*, 2011.

[76] M. S. Ryoo, Thomas J. Fuchs, Lu Xia, J.K. Aggarwal, and Larry Matthies. Robot-centric activity prediction from first-person videos: What will they do to me? In *HRI*, 2015.

[77] M. S. Ryoo and Larry Matthies. First-person activity recognition: What are they doing to me? In *CVPR*, 2013.

[78] Walter J. Scheirer, Anderson Rocha, Ross Michaels, and Terrance E. Boult. Meta-recognition: The theory and practice of recognition score analysis. *PAMIs*, 2011.

[79] Yifan Shi, Yan Huang, D. Minnen, A. Bobick, and I. Essa. Propagation networks for recognition of partially ordered sequential action. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004.

[80] Yale Song, Louis-Philippe Morency, and Randall Davis. Multi-view latent variable discriminative models for action recognition. In *CVPR*, 2012.

[81] Ekaterina H. Spriggs, Fernando De la Torre, and Martial Hebert. Temporal segmentation and activity classification from first-person sensing. In *IEEE Workshop on Egocentric Vision*, 2009.

[82] Sudeep Sundaram and Walterio Mayol-Cuevas. High level activity recognition using low resolution wearable vision. In *CVPR*, 2009.

[83] Sudeep Sundaram and Walterio Mayol-Cuevas. What are we doing here? egocentric activity recognition on the move for contextual mapping. In *ICRA*, 2012.

[84] Moritz Tenorth, Fernando De la Torre, and Michael Beetz. Learning probability distributions over partially-ordered human everyday activities. In *ICRA*, 2013.

[85] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 2008.

[86] Pavan Turaga, Ashok Veeraraghavan, and Rama Chellappa. Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In *CVPR*, 2008.

[87] Anwaar ul Haq, I. Gondal, and M. Murshed. On dynamic scene geometry for view-invariant action matching. In *CVPR*, 2011.

[88] L. J. P. van der Maaten, M. Welling, and Saul. Hidden-Unit Conditional Random Fields. 2011.

[89] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action Recognition by Dense Trajectories. In *CVPR*, 2011.

[90] Heng Wang, Alexander Klser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 2013.

[91] Daniel Weinland, Mustafa Özuysal, and Pascal Fua. Making action recognition robust to occlusions and viewpoint changes. In *ECCV*, 2010.

[92] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Free viewpoint action recognition using motion history volumes. *CVIU*, 2006.

[93] Chen Wu, Amir Hossein Khalili, and Hamid Aghajan. Multiview activity recognition in smart homes with spatio-temporal features. In *EEE International Conference on Distributed Smart Cameras*, 2010.

[94] Xinxiao Wu and Yunde Jia. View-invariant action recognition using latent kernelized structural svm. In *ECCV*, 2012.

[95] Xinxiao Wu, Dong Xu, Lixin Duan, and Jiebo Luo. Action recognition using context and appearance distribution features. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 2011.

[96] Pingkun Yan, Saad M. Khan, and Mubarak Shah. Learning 4d action feature models for arbitrary view action recognition. In *CVPR*, 2008.

[97] Z. Uhrikova, et al. Action tremor analysis from ordinary video sequence. *Conf Proc IEEE Eng Med Biol Soc*, 2009.

[98] M. Zanfir, M. Leordeanu, and C. Sminchisescu. The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection. In *ICCV*, 2013.

[99] Liyue Zhao, Xi Wang, Gita Sukthankar, and Rahul Sukthankar. Motif discovery and feature selection for crf-based activity recognition. In *ICPR*, 2010.

[100] Feng Zhou, Fernando De la Torre, and Jessica K. Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *PAMI*, 2013.

VITA

Bilge Soran is a PhD student at the University of Washington, Computer Science and Engineering Department. She earned a Master of Science degree from the University of Washington, in 2010, where the thesis topic was *Parcellation of Human Inferior Parietal Lobule Based on Diffusion MRI*. She earned another Master of Science degree from the Gebze Institute of Technology, in 2007, where the thesis topic was *Event Driven Molecular Dynamics Simulation*. Before that, she obtained her Bachelors in Science degree from the Middle East Technical University, in 2002. She worked in The Scientific and Technological Research Council of Turkey between 2003-2009, as a senior researcher on avionics and simulation systems. Before that she had 2 years of industry experience.

She welcomes your comments to `bilge@cs.washington.edu`.