Human Activity and Behavior Analysis

Advances in Computer Vision and Sensors

Volume 2

Edited by

Md Atiqur Rahman Ahad
Sozo Inoue
Guillaume Lopez
Tahera Hossain

CRC Press
Taylor & Francis Group

Human Activity and Behavior Analysis

Human activity and behavior analysis relates to the field of vision and sensor-based human action or activity and behavior analysis and recognition. This book includes a series of methodologies, surveys, relevant datasets, challenging applications, ideas, and future prospects.

This book discusses the topics such as action recognition, action understanding, gait analysis, gesture recognition, behavior analysis, emotion and affective computing, and related areas. This volume focuses on two main subject areas: Movement and Sensors, and Sports Activity Analysis.

The editors are experts in these arenas, and the contributing authors are drawn from high-impact research groups around the world. This book will be of great interest to academics, students, and professionals working and researching in the field of human activity and behavior analysis.

Ubiquitous Computing, Healthcare and Well-being

Human activity recognition has been researched in thousands of papers, with mobile/environmental sensors in ubiquitous/pervasive domains, and with cameras in vision domains. Human behavior analysis is also explored for long-term health care, rehabilitation, emotion recognition, human interaction, and so on. However, many research challenges remain for realistic settings, such as complex and ambiguous activities/behavior, optimal sensor combinations, (deep) machine learning, data collection, platform systems, and applications.

The ***Ubiquitous Computing, Healthcare and Well-being*** series provides a forum for capturing the latest advances and setting the course for future research in these areas. Books in the series will typically be presented by leading researchers globally and will cover, among other things: original methods; exploration of new applications; excellent survey papers; presentations on relevant datasets; challenging applications; ideas and future scopes; guidelines.

Series Editors:
Sozo Inoue, *Professor and Director of the Care XDX Center, Kyushu Institute of Technology, Japan*
Md Atiqur Rahman Ahad, *Associate Professor of Artificial Intelligence and Machine Learning, Department of Computer Science and Digital Technology, University of East London, UK*

Human Activity and Behavior Analysis: Advances in Computer Vision and Sensors: Volume 1
Edited by Md Atiqur Rahman Ahad, Sozo Inoue, Guillaume Lopez, Tahera Hossain

Human Activity and Behavior Analysis: Advances in Computer Vision and Sensors: Volume 2
Edited by Md Atiqur Rahman Ahad, Sozo Inoue, Guillaume Lopez, Tahera Hossain

For more information about this series please visit:
https://www.routledge.com/UbiquitousComputingHealthcareandWell-being/book-series/UCHAWB

Human Activity and Behavior Analysis

Advances in Computer Vision and Sensors: Volume 2

Edited by

Md Atiqur Rahman Ahad, Sozo Inoue,
Guillaume Lopez, and Tahera Hossain

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

Designed cover image: Shutterstock

First edition published 2024
by CRC Press
6000 Broken Sound Parkway NW, Suite 300, Boca Raton, FL 33487-2742

and by CRC Press
4 Park Square, Milton Park, Abingdon, Oxon, OX14 4RN

CRC Press is an imprint of Taylor & Francis Group, LLC

ISBN: 9781032598765 (hbk)
ISBN: 9781032636030 (pbk)
ISBN: 9781032636054 (ebk)

DOI: 10.1201/9781032636054

Typeset in Minion
by codeMantra

Contents

Preface

Sensors and cameras are exploited for the analysis and recognition of human activity and behavior. In the book *Human Activity and Behavior Analysis: Advances in Computer Vision and Sensors*, we have divided across two volumes, 40 wonderful chapters under five parts, Part 1: Healthcare and Emotion (Chapters 1–7), Part 2: Mental Health (Chapters 8–14), Part 3: Nurse Care Records (Chapters 15–26), Part 4: Movement and Sensors (Chapters 27–36), and Part 5: Sports Activity Analysis (Chapters 37–40). These chapters were developed from the *4th International Conference on Activity and Behavior Computing* (ABC 2022, https://abc-research.github.io/), held at University of East London, UK. These chapters were selected after a rigorous review process by related top experts and strict review rebuttal process. We believe the chapters will enrich the research community in the field of Activity and Behavior Computing (ABC). We hope that this book will ignite several exciting research directions. We cordially thank all authors, reviewers, and chairs for their great efforts.

Best regards,
Md Atiqur Rahman Ahad, *University of East London, UK*
Sozo Inoue, *Kyushu Institute of Technology, Japan*
Guillaume Lopez, *Aoyama Gakuin University, Japan*
Tahera Hossain, *Aoyama Gakuin University, Japan*

Editors

Md Atiqur Rahman Ahad, PhD, is Associate Professor of AI and M at the University of East London, UK.

Sozo Inoue, PhD, is Professor at the Kyushu Institute of Technology, Japan.

Guillaume Lopez, PhD, is Professor at Aoyama Gakuin University, Japan.

Tahera Hossain, PhD, is Assistant Professor (Project), Aoyama Gakuin University, Japan.

Contributors

Md Atiqur Rahman Ahad
Department of Computer Science & Digital Technology
University of East London
London, UK

Hikari Aoyagi
Department of Integrated Information Technology
Aoyama Gakuin University
Tokyo, Japan

Yutaka Arakawa
Department of Life Science and System Engineering
Kyushu Institute of Technology
Kitakyushu, Japan

Reza Asvadi
Cognitive Telecommunication Research Group
Department of Electrical Engineering
Shahid Beheshti University G. C.
Tehran, Iran

Yu Enokibori
Graduate School of Informatics
Nagoya University
Nagoya, Japan

Ryota Fukumoto
Department of Computer Science
The University of Electro-Communications
Chofu, Japan

Seyed Ali Ghorashi
Department of Computer Science & Digital Technologies
School of Architecture, Computing, and Engineering
University of East London
London, United Kingdom

Yoshinori Hanada
Department of Integrated Information Technology
Aoyama Gakuin University
Tokyo, Japan

Yuichi Hattori
Department of Life Science and System Engineering
Kyushu Institute of Technology
Kitakyushu, Japan

Masayuki Hirafuji
Graduate School of Agricultural & Life Sciences
The University of Tokyo
Tokyo, Japan

Tahera Hossain
Graduate School of Science and Engineering
Aoyama Gakuin University
Tokyo, Japan

Kazuki Imura
Department of Integrated Information Technology
Aoyama Gakuin University
Tokyo, Japan

Sozo Inoue
Department of Human Intelligence Systems
Graduate School of Life Science and Systems Engineering
Kyushu Institute of Technology
Kitakyushu, Japan

Shun Ishii
Department of Integrated Information Technology
Aoyama Gakuin University
Tokyo, Japan

Shogo Ishikawa
Graduate School of Information Science and Technology
Shizuoka University
Shizuoka, Japan

Md Baharul Islam
Department of Computer Engineering
Bahcesehir University
Istanbul, Turkey

Ami Jinno
Department of Science and Engineering
Aoyama Gakuin University
Tokyo, Japan

Katsuhiko Kaji
Department of Information Sciences
Aichi Institute of Technology
Toyota, Japan

Shinji Kawakura
Graduate School of Engineering Mechanical & Physical Engineering
Osaka Metropolitan University
Osaka, Japan

Shinya Kiriyama
Graduate School of Information Science and Technology
Shizuoka University
Shizuoka , Japan

Pascal Krenckel
Department of Computer Science
Otto von Guericke University
Magdeburg, Germany

Claudia Krull
Department of Computer Science
Otto von Guericke University
Magdeburg, Germany

Akinori Kunishima
Graduate School of Information Science and Technology
Shizuoka University
Shizuoka, Japan

Guillaume Lopez
Department of Integrated Information Technology
Aoyama Gakuin University
Tokyo, Japan

Parisa Fard Moshiri
Cognitive Telecommunication Research Group
Department of Electrical Engineering
Shahid Beheshti University G.C.
Tehran, Iran

Nishiki Motokawa
Department of Science and Engineering
Aoyama Gakuin University
Tokyo, Japan

Lauro Fialho Müller
Department of Computer Science
Otto von Guericke University
Magdeburg, Germany

Kazuya Murao
College of Information Science and Engineering
Ritsumeikan University
Kyoto, Japan

Mohammad Nabati
Cognitive Telecommunication Research Group
Department of Electrical Engineering
Shahid Beheshti University G. C.
Tehran, Iran

Nazmun Nahid
Department of Life Science and System Engineering
Kyushu Institute of Technology
Kitakyushu, Japan

Masayuki Numao
Department of Computer Science
The University of Electro-Communications
Chofu, Japan

Yuki Ogane
Graduate School of Management Information Science
Aichi Institute of Technology
Toyota, Japan

Masahiro Okamoto
College of Information Science and Engineering
Ritsumeikan University
Kyoto, Japan

Atsushi Omata
Graduate School of Information Science and Technology
Shizuoka University
Shizuoka, Japan

Arezoo Sadeghzadeh
Department of Computer Engineering
Bahcesehir University
Istanbul, Turkey

Keisuke Sato
Department of Science and Engineering
Aoyama Gakuin University
Tokyo, Japan

Hossein Shahverdi
Self-Employed Researcher
Tehran, Iran

Ryosuke Shibasaki
Graduate School of Engineering
University of Tokyo
Tokyo, Japan

Koki Suzuki
Graduate School of Information Science and Technology
Shizuoka University
Shizuoka, Japan

Min Xinyi
Department of Life Science and System Engineering
Kyushu Institute of Technology
Kitakyushu, Japan

Anna Yokokubo
Department of Integrated Information Technology
Aoyama Gakuin University
Tokyo, Japan

I

Movement and Sensors

CHAPTER 1

Testing the Applicability of Virtual Stochastic Sensors in Human Activity Recognition

Claudia Krull, Pascal Krenckel, and Lauro Fialho Müller

Otto von Guericke University Magdeburg

1.1 INTRODUCTION

Activity recognition has been a hot research topic for some years now. One of its main applications is ambient assisted living, so that people in need of assistance might stay in their own environment longer than previously possible. There is a wide range of sensors of different types and applications being tested in this domain, ranging from ambient binary motion sensors and pressure mats to wearable gyroscopes or sensors monitoring body functions. This is also due to the many different levels of surveillance tested by the researchers. Where ambient sensors can usually only detect whether a certain device is in use or area is occupied not distinguishing residents, wearables can precisely monitor the current status of a particular person. Consequently, there is also a large variety of methods employed for activity recognition, such as conditional random fields (CRF), support vector machines (SVM) or hidden Markov models (HMM). Furthermore, the applications differ in the type of features used and the offline or online activity recognition [3].

Virtual stochastic sensors (VSS) have been developed for behavior reconstruction of partially observable stochastic systems and to enable solving backward problems in the realm of stochastic modeling and simulation [11]. VSS provide different types of models for discrete and hybrid doubly stochastic systems and the respective analysis methods to perform behavior reconstruction. The models are based on the ideas of HMM, but extend these by arbitrary non-Markovian distribution functions for multiple concurrent processes. The reconstruction methods used stem from state space-based simulation using supplementary variables to extend the discrete state definition [11,14]. Discretizing the time domain results in a simple algorithmic

DOI: 10.1201/9781032636054-2

analysis method, instead of solving partial differential equations. This enables VSS to perform system behavior reconstruction for a wide variety of real-world problems. VSS have for example been applied successfully to load contribution reconstruction in non-intrusive appliance load monitoring [13], as well as in gesture recognition with a time component in human–computer interaction [8]. The similarity of VSS and HMM, as well as the advantages VSS have shown when including the dynamic time information, make them a good candidate for activity reconstruction from sensor readings.

This chapter demonstrates the application of VSS in ambient assistent living by using the publicly available CASAS Aruba 2010 data-set [7] to reconstruct residents activities based on binary sensor readings. This data-set is taken from a single resident private home with environmental sensors and contains annotated data. This first study shows that VSS can compete with existing approaches in activity recognition and paves the way to more detailed analysis. It is based on the Master thesis of one of the authors [16].

1.2 RELATED WORK

This section points to some research work in ambient assisted living that is similar in approach or goal to this paper. Furthermore, some relevant basics for virtual stochastic sensors (VSS) are presented.

1.2.1 Activity Recognition

Baghezza et al. show the wide range of research topics in activity recognition, including sensor types, data handling, algorithms, and their performance. First, we will concentrate on some of these papers that relate to our own research either by using similar data or similar methods, showing which accuracy VSS have to achieve to be competitive [3].

Aicha et al. use binary motion sensors and HMM to detect the presence of more than one occupant in a normally single-resident home achieving an accuracy of 85% [1]. Kasteren et al. use a wide range of sensor readings and compare raw data, change point and last firing features for activity monitoring using CRF and HMM, achieving 95% and 91% accuracy, respectively [20]. Samarah et al. use binary sensor data and compare CRF, HMM and Naive Bayes Classifiers (NBC) achieving accuracies between 71% and 79%. The paper also addresses privacy issues when monitoring home environments and presents an approach to enhance privacy protection [18]. Tsai et al. also use a CASAS data-set and pattern learning on per-day activity distribution as feature, achieving an accuracy of up to 80% for activity recognition [19].

Some further papers also using CASAS data-sets for activity recognition are listed below. How their results compare to the ones using VSS will be discussed in Section 1.4.4. Chen et al. use feature selection on a CASAS data-set and compare the recognition accuracy of Bayesian belief networks (BBN), artificial neural networks (ANN), SVN, and an ensemble method, achieving at most 90% accuracy [5]. Cook trains generalized CRF, HMM and NBC models using 11 CASAS data-sets and depending on the setup achieves 75% accuracy or when using semi-supervised learning up to 100% for some data-sets [6]. Fatima et al. achieve a recognition accuracy of over 92% using SVM kernels and an optimization approach [9]. Oukrich et al. use neural networks with an auto encoder and supervised back-propagation, achieving 90% accuracy [17]. Aminikhanghahi and Cook test sequential activity segmentation and can achieve an accuracy of up to 80% in an online segmentation and recognition setting [2]. Liciotti et al. use LSTM as a deep learning approach to find a generalized model for different data sets and achieve up to 94% F1 measure [15]. From these papers, we conclude that our method needs to achieve an accuracy of at least 80% to be considered competitive.

1.2.2 Virtual Stochastic Sensors

Virtual stochastic sensors (VSS) represent a framework for analyzing partially observable stochastic systems, including different modeling paradigms and solution methods. They can solve inverse problems in modeling and simulation, where the output of a system is known, or observable, and one is looking for the cause of that observation. VSS can outperform other analysis methods using black box models, when information about the hidden systems structure is available. The modeling paradigms stem from non-Markovian stochastic Petri nets and can therefore incorporate such information to accurately represent dynamic system behavior and its relationship with the observable output [11].

The modeling paradigms include hidden non-Markovian models (HnMM) [12] and conversive HnMM (CHnMM) [4], which use augmented stochastic Petri nets (ASPN) [4, 11] as user model that can contain multiple concurrent non-Markovian transitions. ASPN can produce observable output by the firing of transitions or by being dependent on the discrete system state. The system output is collected in a protocol with time stamps, since in contrast to HMM, the model is defined in continuous time and can produce output at arbitrary points in time. A formal definition of ASPN and HnMM can be found in Ref. [11].

The reconstruction algorithm used for VSS is based on the Proxel method [10, 14]. Proxels are probability elements that extend the discrete state space of a system by incorporating age information of the non-Markovian transitions. Each Proxel is processed to determine the possible follow-up system states in a discrete time step. This way the reachable state space of the system is explored in discrete time steps, on the fly calculating the probabilities of all reachable states. This approach avoids having to solve differential equations when including non-Markovian distributions. However, the method suffers from state space explosion, when the discrete state space is too large or the time step too small. To manage state space explosion, Proxels with very low probability are discarded, which introduces an error. Therefore, the time step size and the number of Proxels retained both present method parameters that can be used to trade off between accuracy and efficiency [14].

The Proxel-based analysis method has been modified to use an ASPN output protocol and follow, in discrete time steps, possible system development paths that could have produced this protocol. On the fly, the probability of each path is computed. The method results in a list of possible system development paths that could have produced an observed output protocol, complete with their probabilities. This result can then be analyzed to reconstruct possible system behavior to have caused the observed output. Thus VSS enable behavior reconstruction based on observed system output for a wide range of stochastic systems [11].

The similarity of HMM and HnMM, as well as the successful application of VSS to non-intrusive appliance load monitoring (NIALM) make them a viable candidate for activity recognition in the ambient assisted living domain.

1.3 MODELING AND RECONSTRUCTION USING VSS

This section will describe the approach taken in this paper to test VSS for activity recognition on the CASAS Aruba 2010 data-set.

1.3.1 Data Preparation

The CASAS Aruba 2010 data-set provides annotated data for the daily activities of a single resident over a period of 220 days. The sensors used include 31 binary motion sensors, 5 temperature sensors, and 4 binary door sensors. The data-set contains more than 1.7 million sensor readings, more than 90% from the motion sensors. In this paper, we focus on the binary sensors, due to their simpler nature, and thus neglect the temperature sensors, removing 6.7% of the sensor readings. In order to improve the data quality, we also removed duplicate readings, incorrect sensor labels and states, malformed states, and non-alternating binary sensor readings, removing 200 additional entries.

The activities labeled in the data-set are listed in Table 1.1. The data-set is highly imbalanced in the distribution of the activities. Due to the low number of data points, we will leave out the *respirate* activity. Furthermore, since entering and leaving home do not correspond to activities that last a significant amount of time, but rather state changes, we will also neglect them here, resulting in a total of eight activities.

In the pre-processing phase, the activity labels were used to calculate the currently ongoing activity over time. As there were no parallel activities in the data-set, this provides a unique label for each time slice. Furthermore, the individual activities duration was computed, as well as the time between consecutive activities. Some activities have interruptions, 60% of these with a duration under 5 minutes and 78% with less than 10 minute duration. We considered 5 minutes as a viable threshold to avoid fragmentation and merged consecutive occurrences of the same activity, when the interruption was below this threshold and did not hold another activity.

TABLE 1.1 Labeled Activities and Number of Occurrences in the CASAS Aruba 2010 Data-Set, Activities Omitted Later Are Emphasized

Activity	**Frequency**
Relax	2910
Meal preparation	1606
Enter home	431
Leave home	431
Sleeping	401
Eating	257
Work	171
Bed to toilet	157
Wash dishes	65
Housekeeping	33
Respirate	6

1.3.2 Activity Modeling

Using the pre-processed data, the actual model can be constructed. VSS are not trained as SVN classifiers are, for example, but are constructed and parameterized using available structural information about the system.

There are two possible approaches for designing the activity models for reconstruction. First, a single ASPN can be built incorporating all activities, their duration and the interactions between them. All this can be derived from the available input data. This option is similar to existing HMM activity recognition approaches [21] and has the advantage of being able to capture not just individual activity characteristics but also their interdependencies. A reconstructed path would then show the sequence of likely activities matching the observation sequence.

The second option is to build one ASPN for each activity, only capturing the relationship of the sensor readings and the respective activity, neglecting possible interactions with other activities but enabling a more detailed representation of individual activities. This approach is similar to the one taken when using VSS for NIALM [13]. The reconstruction would then yield a likelihood for each activity and the overall activity sequence would have to be determined in a post-processing step, comparing the individual values.

In this paper, we have chosen the second option for the following reasons. The state space-based analysis method used for reconstruction suffers from state space explosion and can therefore work more efficiently with smaller discrete state spaces. Secondly, using individual activity models is much more versatile, as activities can easily be removed or added in the analysis without affecting the other models, whereas a single model would have to be completely restructured. Using individual models also prevents over-fitting to a single household setting and can be more easily automated for future applications.

For the individual models, we have chosen to use a generic ASPN for most activities, as shown in Figure 1.1a. The *sleeping* activity was closely tied to the

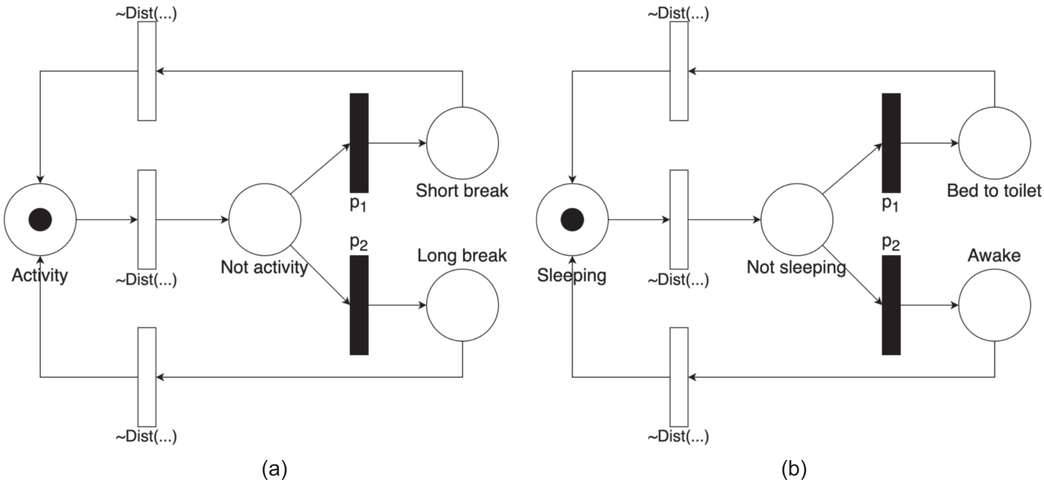

Figure 1.1 Generic activity ASPN (a) and sleeping activity ASPN (b).

bed to toilet activity, so these were merged into one ASPN, shown in Figure 1.1b. The *work* and *eating* activities showed distinctly more complex break behavior, with weekends and nights being distinct from long and short breaks; therefore, these were also structured individually. The activity duration distributions were determined based on the individual duration samples from the pre-processing step.

Augmenting the ASPN with the symbol emission probabilities results in a model structure, where each state representing an activity or break is associated with the individual sensor output probabilities, as seen in Figure 1.2. The output probabilities were estimated using the assumption of the sensors being mutually independent and then determining the probability that a specific sensor was activated during a specific activity.

A fully specified ASPN of the *sleeping* activity from one instance in the cross-validation process is described below, for the notation details please refer to [11,16]. The output probabilities were omitted for reasons of brevity.

$$
\begin{aligned}
S &: \{s_1 = \text{Sleeping}, s_2 = \text{Bed } \textit{to} \text{ toilet}, s_3 = \text{Awake}\} \\
V &: \{v_1, v_2, \ldots, v_m\} \\
\text{TR} &: \{\text{TR}_1, \text{TR}_2, \text{TR}_3\} \\
\text{TR}_1 &: (\sim \text{Normal}(252.47, 137.71), \text{false}) \\
\text{TR}_2 &: (\sim \text{StudT}(4.26), \text{false}) \\
\text{TR}_3 &: (\sim \text{StudT}(72204.24), \text{false}) \\
A &: \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} \emptyset & \{\text{TR}_1\} & \{\text{TR}_1\} \\ \{\text{TR}_2\} & \emptyset & \emptyset \\ \{\text{TR}_3\} & \emptyset & \emptyset \end{bmatrix} \\
B &: \{b_{11}, b_{12}, \ldots b_{1m}, b_{21}, \ldots b_{3m}\} \\
\Pi &: (1, 0, 0)
\end{aligned}
$$

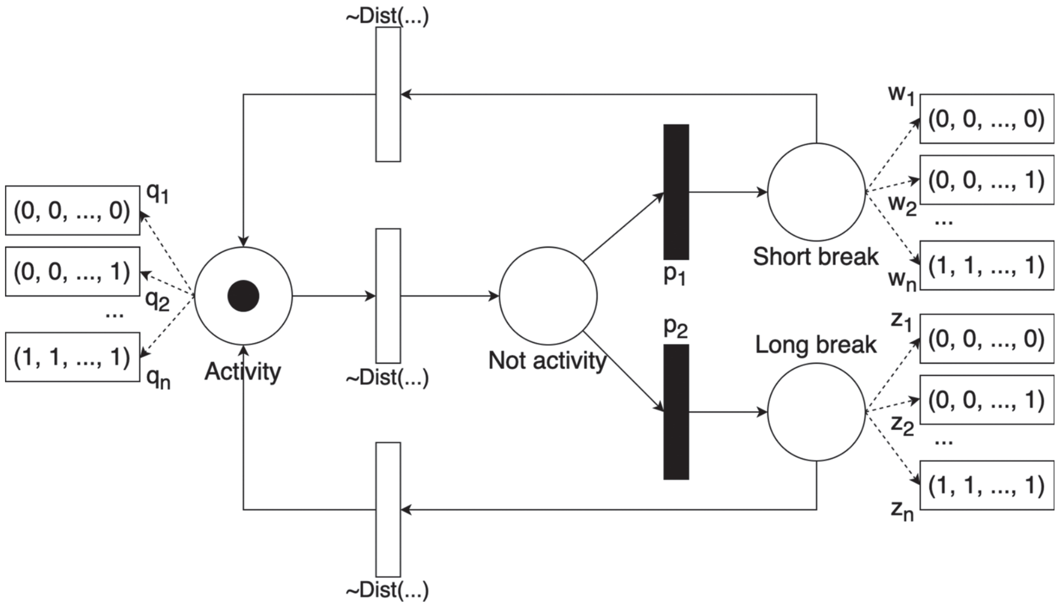

Figure 1.2 Generic activity ASPN with sensor output probabilities.

1.3.3 Reconstruction Approach

Using the now fully specified ASPN of the activities, we can adapt the existing reconstruction approach for VSS. The path reconstruction for each individual model can be done using the generic approach described in Ref. [11]. The system is initialized in one state, and the algorithm explores the possible system developments in discrete time steps, simultaneously keeping track of the probability of a specific development and whether the development matches the observed sequence. The time step is a crucial parameter in the solution method and determines both the accuracy and the efficiency, where a good compromise has to be found between these. In Section 1.4.2.1 we present an experiment testing which time step value represents a good trade-off for activity recognition in the given example.

The reconstruction results of the individual models contain, for each time slice of the observation sequence, the respective probabilities of the individual activities. As a post-processing step, a simple decision engine is used, that picks the most likely current activity from the reconstruction results for each time slice. An example is shown in Figure 1.3. This results in every time slice being tagged with one distinct activity.

Since some of the reconstructed probabilities are very low, a reasonable choice would also be to tag a time slice as *None*, when all activities are below a certain probability threshold (see Figure 1.4). The experiment in Section 1.4.2.3 shows how such a threshold affects the recognition accuracy.

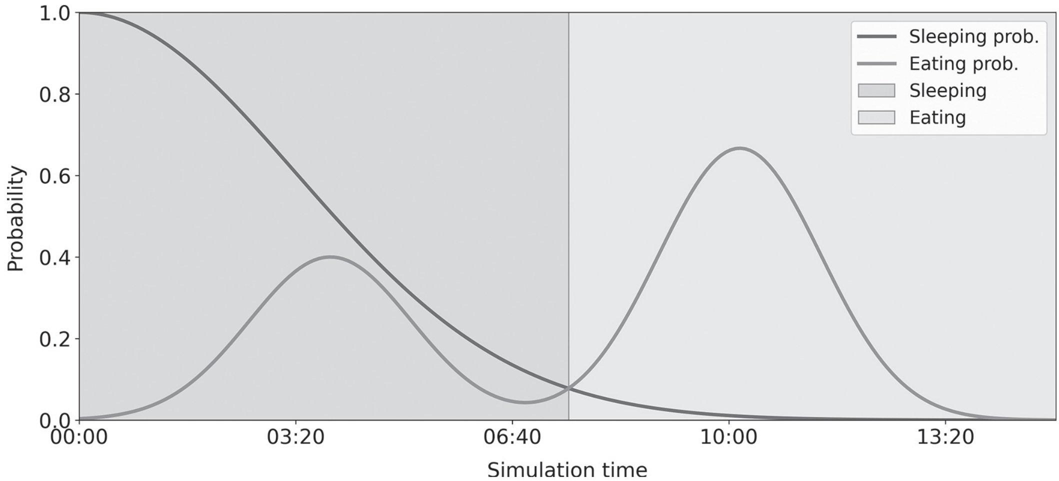

Figure 1.3 Decision engine logic example.

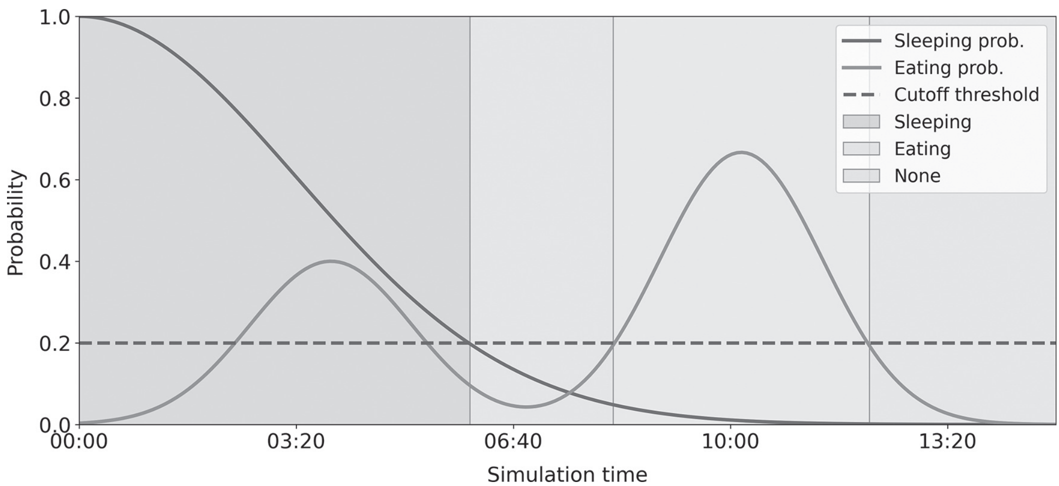

Figure 1.4 Decision engine logic example with cutoff probability.

1.4 EXPERIMENTS

To evaluate the applicability of virtual stochastic sensors (VSS) to human activity recognition (HAR), we conducted three different experiments which vary different parameters of the solution method. The computer system used was a MacBook Pro 2019 (Intel i7-9750H, 16GB DDR4-2666). The following list motivates each experiment.

- Experiment 1: Time steps size: The computational complexity of the Proxel Method is highly dependent on the time step Δt. A smaller time step leads to a smaller discretization error and thus to a more accurate model behavior. However, a smaller time step increases the computational requirements.

With this experiment, we try to evaluate the influence of different time steps and find a good trade-off.

- Experiment 2: Proxels retained per time step: Another important method parameter is the number of Proxels retained per time step. To avoid exponential run-time complexity, the number of Proxels per time step is kept at a specific maximum. However, the larger the number, the more accurate is the solution. Therefore, with this experiment we try to evaluate the influence of the number of Proxels retained.

- Experiment 3: Performance with *None*-label: In the data, there are time slices when none of the labeled activities are performed. The decision engine always picks the activity with the highest probability. In these cases, the reconstructed activity will always be wrong. To counter this, a *None*-Label is introduced. If the highest probability for the activities is smaller than a threshold, then the time slice is labeled as *None*. We compare the method performance with and without this *None* label.

1.4.1 Evaluation Criteria

To evaluate the performance of our recognition method, we calculated the following metrics.

- Accuracy: The accuracy is the number of correctly classified samples divided by the total number of samples.

- Precision: The precision is the number of correctly as positive classified samples divided by the total number of positive classified samples.

- Recall: The recall is the number of correctly as positive classified samples divided by the total number of positive samples.

- $F_1 - score$: The F_1-score is the harmonic mean of precision and recall. $F_1 = 2\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}$

Precision and recall are measures for binary classifiers. The data-set used in this chapter, however, contains multiple classes. The two measures are calculated for each class individually, where positive means that the sample is part of the class and negative means it's not. To combine these values into a measure over all classes, a weighted-average is used. The weights are determined by the class size. In this case, the *average recall* is the same as the *accuracy*. One problem of the weighted-average is that small classes have little influence on the performance. Therefore, we also compute the macro-average, which uses the same weight for all classes despite their imbalance.

To avoid overfitting, we use k-fold cross-validation. The data-set is divided into k parts, where $k-1$ parts are used for training and the remaining part is used for testing. The cross-validation process is then repeated for every part of the data-set.

Finally the average over all subsets is calculated to receive a single estimate of the performance metric. Since the CASAS Aruba 2010 data-set contains 8 months of data we used 8-fold cross-validation with each subset containing one month of data.

1.4.2 Experiment Results

In this section, we present the results of the base model and our three experiments. The base model has the following specifications:

- time step Δt =60 s, which is used in most cases according to [3].
- Proxels retained per iteration: 200
- Decision engine: highest probability, no cutoff threshold

Table 1.2 shows the performance of the base model for every activity and the weighted average over all classes. It shows that the base model performs well for majority classes. The weighted-average for the precision is 0.8759. The accuracy is slightly lower with 0.8531. However, the algorithm underperforms for minority classes. Therefore, the macro-average is 0.5638 (Precision) and 0.5030 (Recall) which is significantly less then the weighted-average. As mentioned in Section 1.2.1, the accuracy in previous works were between 71% and 95%, so our approach achieves acceptable results with 85%.

1.4.2.1 Experiment 1: Varying Time Steps Δt

For the first experiment, we varied the discretization step size as follows: $\Delta t \in \{60, 90, 120, 240, 300, 600, 1200$ s$\}$. The other parameters remained the same as the base model's. Figure 1.5 shows the overall performance for the different step sizes. Up to a Δt of 120 s, the model performs as well as the base model. Larger step sizes, as expected, have a negative impact on the classification performance. On the other hand, larger step sizes take less time to compute. Although the wall time was

TABLE 1.2 Performance of the Base Model Per Activity, Macro and Weighted Average (Accuracy Emphasized)

Activity	**Precision**	**Recall**	**F1-Score**
Bed to toilet	0.2378	0.2844	0.2590
Eating	0.7176	0.5280	0.6084
Housekeeping	0.0297	0.1390	0.0489
Meal preparation	0.8374	0.7002	0.7627
Relax	0.9510	0.7739	0.8534
Sleeping	0.8199	0.9830	0.8940
Wash dishes	0.0064	0.0129	0.0086
Work	0.9106	0.6026	0.7253
Macro-average	0.5638	0.5030	0.5200
Weighted-average	0.8759	*0.8531*	0.8560

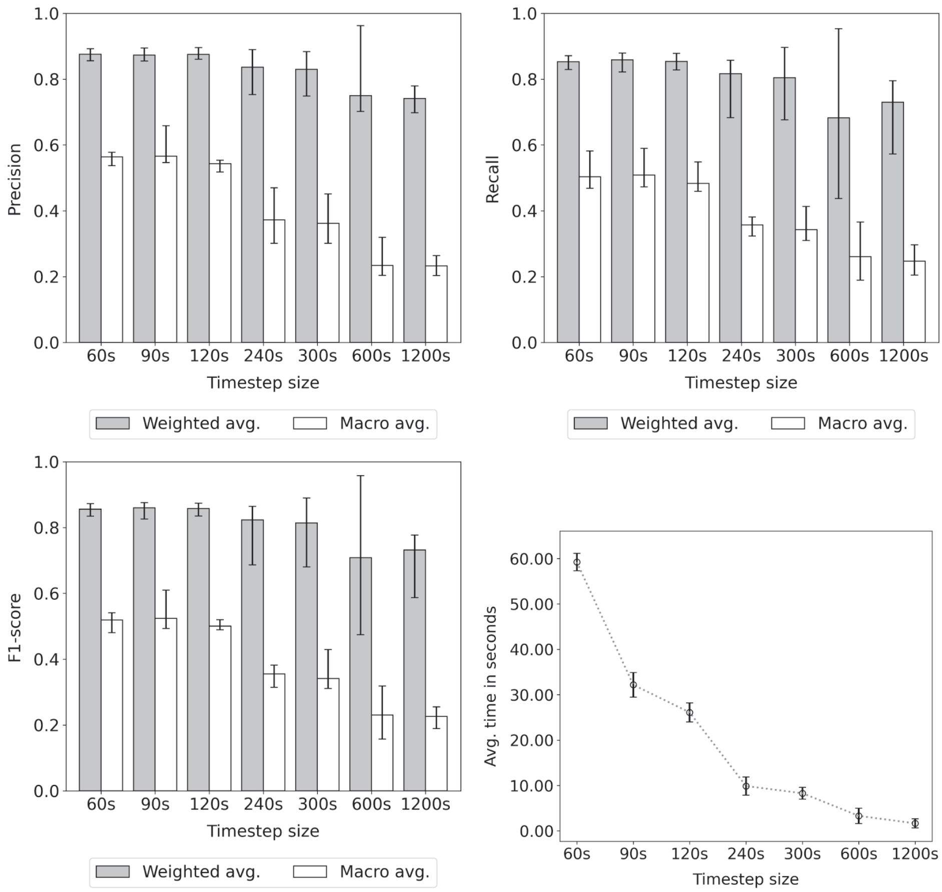

Figure 1.5 Average, minimum and maximum classification performance and run-time for varying Δt, base model $= 60$ s.

used, it still gives an indication about the run-time differences for different time step sizes.

1.4.2.2 Experiment 2: Varying Amount of Proxels Retained

The base model was run with a limit of 200 Proxels. A Proxel contains the expanded system state and the probability to be in this state. For every time step, only the 200 most likely Proxels were retained. This method tackles the state space's exponential growth, since every active transition leads to a new state space element and thus a new Proxel. However, it is possible that many of these Proxels have no influence on the models overall behavior. The number of Proxels needed is highly system dependent. As in experiment 1 all other parameters are kept the same, only varying $\#_{\text{Proxels retained}} \in \{10, 20, 50, 100, 200, 500, 1000, 2000\}$.

Figure 1.6 shows the results of Experiment 2. There are no significant differences between the different number of Proxels retained. Surprisingly, even 10 Proxels

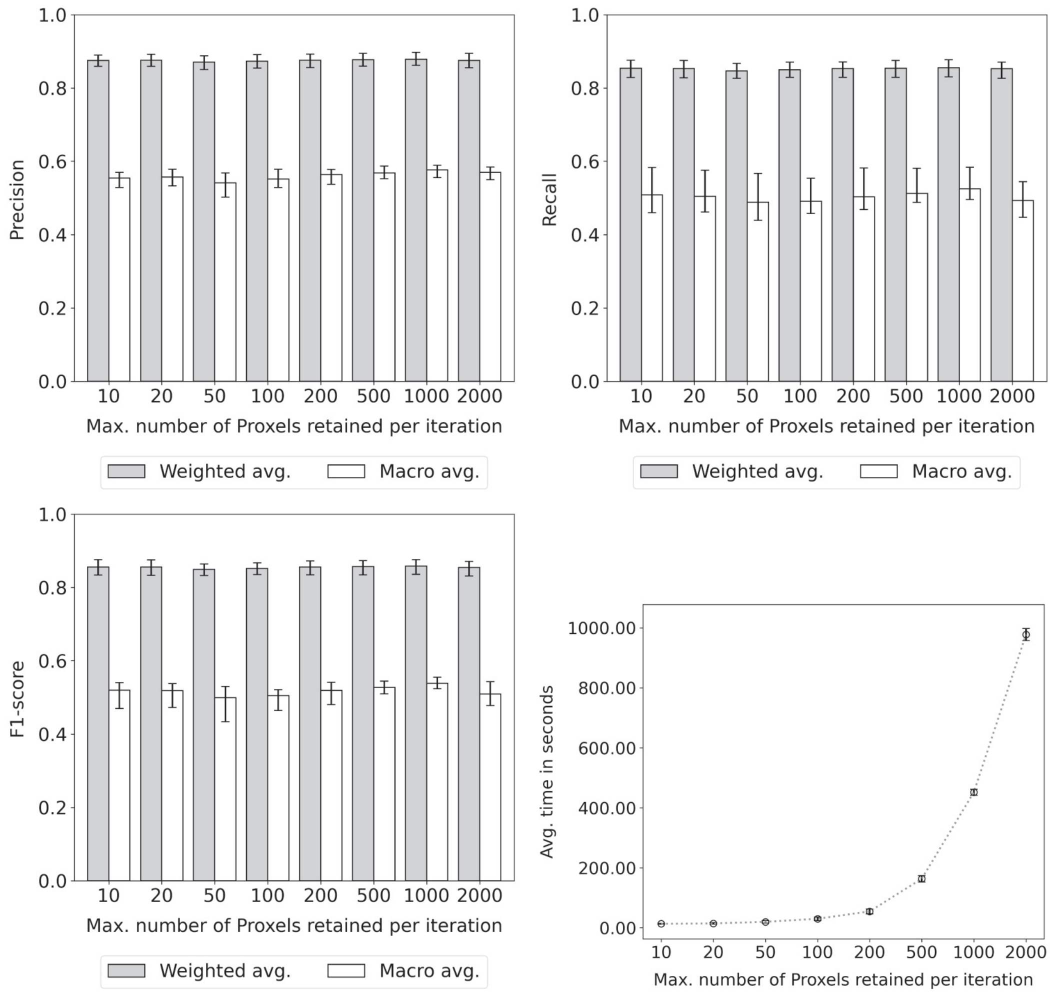

Figure 1.6 Average, minimum and maximum classification performance and run-time for different number of Proxel retained, base model= 200 Proxel.

retained are not significantly worse than 2000. In contrast, the number of Proxels retained does have significant influence on the run-time of the algorithm. The wall-time of the algorithm for 10 Proxel was 12s, whereas the base model with 200 Proxel took 56s.

1.4.2.3 *Experiment 3: Performance with* None-*Label*

One limitation of the base model is not being able to output the *None*-label. Some time slices do not contain any of the labeled activities. In these cases, the original data points are labeled as *None*. To counter this problem, instead of always classifying a data point based on the highest probability, we output a *None*-label if the maximum probability is less then a threshold $\in \{0.00, 0.01, 0.02, 0.05, 0.10, 0.15, 0.20\}$.

Figure 1.7 shows the results when including a cutoff threshold and *None* label. A slight increase in precision and recall can be observed for both the weighted-average and the macro-average. The reason for this increase is that the *None*-label is not included in the average. In the base model, only the mentioned activities were regarded as classes. For example for the base model the class precision for *None* can't be calculated and the recall is always zero. Although we include the *None*-label we do not include *None* as an additional class. Therefore, data points labeled as *None* are not included in our calculations at all. Only those that are wrongly classified as *None* are included in the calculations of the recall for the class they belong to. Since the data points classified as *None* are data points whose classification is usually uncertain, the classification performance improves slightly.

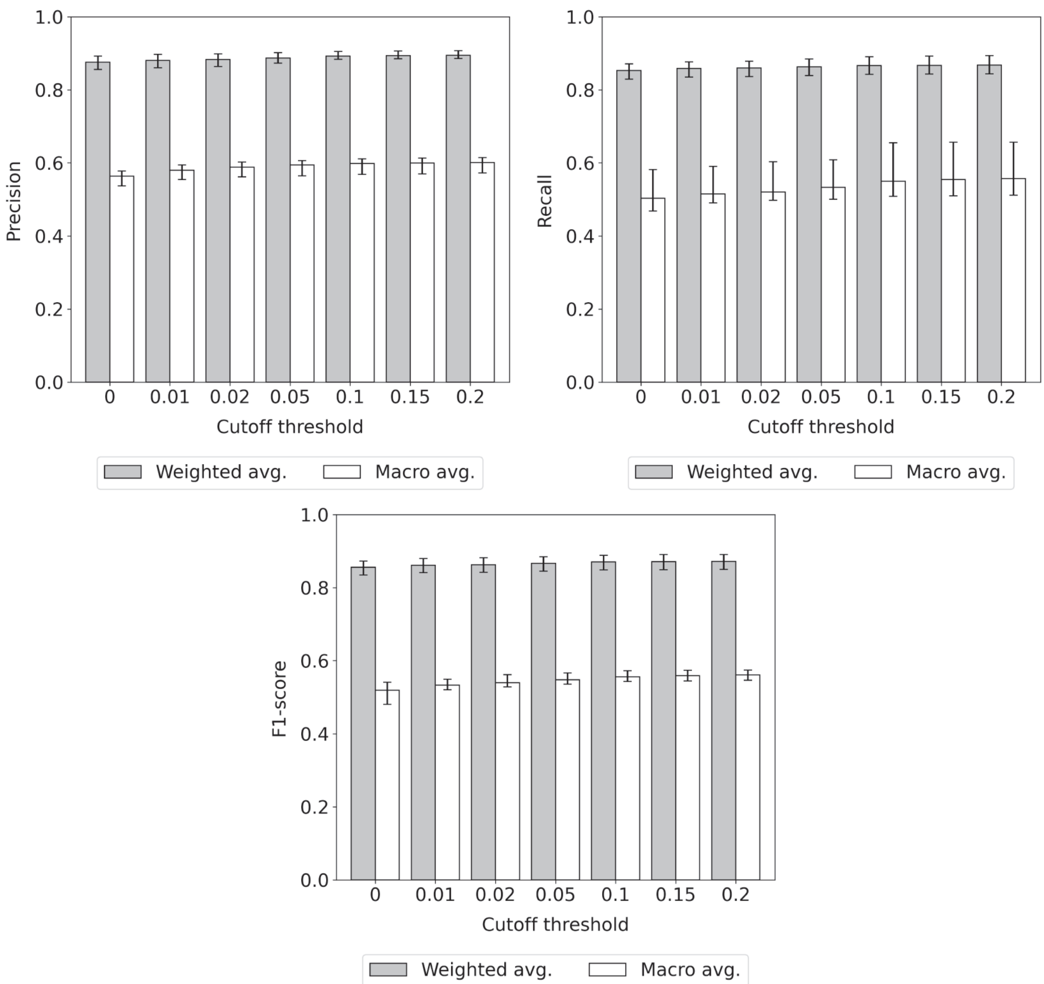

Figure 1.7 Average, minimum and maximum classification performance for varying cutoff threshold excluding the *None*-class from the accuracy calculation.

Therefore, we also did the same experiment including the *None*-class in our average calculations. Figure 1.8 shows the results. As expected, the performance drops sharply, compared to the base model. The accuracy drops from originally 0.85 to 0.57. The weighted average of the precision drops even more, from 0.88 to 0.41. With a larger cutoff threshold the precision also increases until a threshold of 0.1. From 0.1 to 0.2 it only increases slightly. But even with a cutoff threshold of 0.2 the precision of 0.6897 is still worse than the precision of the base model. This increase is also visible in the marco-average of the precision. In contrast, the recall increases only slightly with larger cutoff thresholds. The macro-average behaves similarly to the weighted-average.

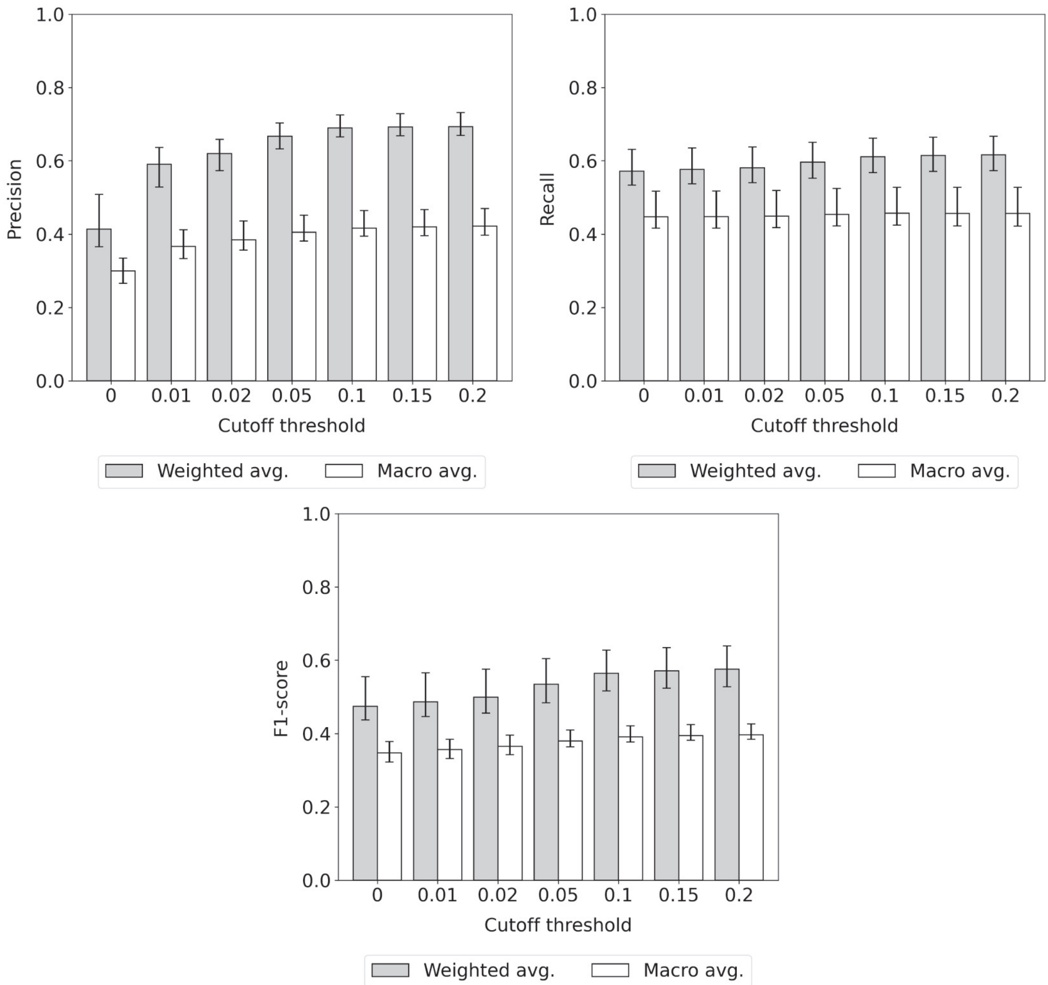

Figure 1.8 Average, minimum and maximum classification performance for varying cutoff thresholds including the *None*-class in the accuracy calculation.