MDPI

Article

A Smart Home Digital Twin to Support the Recognition of Activities of Daily Living

Damien Bouchabou [1,‡*], Juliette Grosset [2,‡], Sao Mai Nguyen [3], Christophe Lohr[4], and Xavier Puig[5]

[1] IMT Atlantique and U2IS, Ensta Paris; damien.bouchabou@gmail.com
[2] IMT Atlantique and ECAM Rennes; juliette.grosset@ecam-rennes.fr
[3] IMT Atlantique and U2IS, ENSTA Paris; nguyensmai@gmail.com
[4] IMT Atlantique; christophe.lohr@imt-atlantique.fr
[5] FAIR; xavierpuig@meta.com
* Correspondence: damien.bouchabou@gmail.fr
‡ These authors contributed equally to this work.

Abstract: One of the challenges in the field of human activity recognition in smart homes based on IoT sensors is the variability in the recorded data. This variability arises from differences in home configurations, sensor network setups, and the number and habits of inhabitants, resulting in a lack of data that accurately represents the application environment. Although simulators have been proposed in the literature to generate data, they fail to bridge the gap between training and field data or produce diverse datasets. In this article, we propose a solution to address this issue by leveraging the concept of digital twins to reduce the disparity between training and real-world data and generate more varied datasets. We introduce the Virtual Smart Home, a simulator specifically designed for modeling daily life activities in smart homes, which is adapted from the Virtual Home simulator. To assess its realism, we compare a set of activity data recorded in a real-life smart apartment with its replication in the Virtual Smart Home simulator. Additionally, we demonstrate that an activity recognition algorithm trained on the data generated by the Virtual Smart Home simulator can be successfully validated using real-life field data.

Keywords: Smart-Home, machine learning, home automation, simulator, database, digital twin, transfer learning

Citation: Bouchabou, D.; Grosset, J.; Nguyen, S.M.; Lohr, C.; Puig, X. A Smart Home Digital Twin to Support the Recognition of Activities of Daily Living. *Sensors* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

1. Introduction

Over the past few decades, there has been a significant increase in the adoption of smart homes and real-world testbeds, driven by the proliferation of Internet of Things (IoT) devices. These devices enable the detection of various aspects within homes, such as door openings, room luminosity, temperature, humidity, and more. Human Activity Recognition (HAR) algorithms in smart homes have become crucial for classifying streams of data from IoT sensor networks into Activities of Daily Living (ADLs). These algorithms enable smart homes to provide adaptive services, including minimizing power consumption, improving healthcare, and enhancing overall well-being.

Despite the notable advancements in machine learning techniques and the improved performance of HAR algorithms, their practical application to real-world test cases continues to encounter challenges. These challenges primarily stem from the variability and sparsity of sensor data, leading to a significant mismatch between the training and test sets.

1.1. A Variable and Sparse Unevenly Sampled Time Series

While HAR based on video data has made significant strides in performance [1], HAR in smart homes continues to encounter specific challenges, as highlighted in the survey by Bouchabou et al. [2]. Recent advances in HAR algorithms, such as

convolutional neural networks [3] and fully connected networks [4], along with sequence learning methods like long-short term memory [5], have contributed to advancements in the field. However, the task of recognizing ADLs in a smart home environment remains inherently challenging, primarily due to several contributing factors:

- Partial observability and sparsity of the data: The input data in HAR consists of traces captured by a variety of sensors, including motion sensors, door sensors, temperature sensors, and more, integrated into the environment or objects within the house [6]. However, each sensor has a limited field of view, resulting in most of the residents' movements going unobserved by the sensor network in a typical smart home setup. Unlike HAR in videos, where the context of human actions, such as objects of interest or the position of obstacles, can be captured in the images, the sparsity of ambient sensors in HAR does not provide information beyond their field of view. Each sensor activation alone provides limited information about the current activity. For example, the activation of a motion sensor in the kitchen could indicate activities such as "cooking," "washing dishes," or "housekeeping." Therefore, the information from multiple sensors needs to be combined to infer the current activity accurately. Additionally, each sensor activation provides only a partial piece of information about the activity and the state of the environment, unlike videos where both the agent performing the activity and the environment state are visible. Consequently, the time series of sensor activity traces cannot be approximated as a Markov chain. Instead, estimating the context or current state of the environment relies on past information and the relationship with other sensors.
- Variability of the data: Activity traces between different households exhibit significant variations. The variability arises from differences in house structures, layouts, and equipment. House layouts can vary in terms of apartments, houses with gardens, houses with multiple floors, the presence of bathrooms and bedrooms, open-plan or separate kitchens, and more. The number and types of sensors can also differ significantly between homes. For instance, datasets like MIT [7] use 77-84 sensors for each apartment, while the Kasteren dataset [8] uses 14-21 sensors. The ARAS dataset [9] includes apartments with 20 sensors, while the Orange4Home dataset [10] is based on an apartment equipped with 236 sensors. All these factors, including home topography, sensor count, and their placement, can result in radical differences in activity traces. The second cause of variability stems from household composition and residents' living habits. ADLs vary depending on the residents' habits, hobbies, and daily routines, leading to different class balances among ADLs. For example, the typical day of a student, a healthy adult, or an elderly person with frailty will exhibit distinct patterns. Furthermore, the more residents there are, the more the sensor activation traces corresponding to each resident's activities become intertwined, leading to complex scenarios involving composite actions, concurrent activities, and interleaved activities.

Therefore, algorithms need to analyze sparse and irregular time series data to generalize across various house configurations, equipment, households, and habits. Training machine learning algorithms to be deployed in such diverse scenarios necessitates training data that encompasses this wide variability.

1.2. Digital Twins for Generating Similar Data

To bridge the gap between training data and real-world usage data, data generation can be a potential solution, particularly through the concept of digital twins. A digital twin refers to a virtual representation that serves as a real-time digital counterpart of a physical object or process [11,12]. In the context of HAR, a digital twin could be a virtual replica of a target house, complete with the same installed sensors. Within this digital environment, one or multiple avatars can simulate ADLs by modeling the behaviors of residents. This way, the digital twin can be used to fine-tune algorithms before their deployment in the actual target house.

Moreover, digital twins have the potential to generate data representing a vast range of house configurations, household habits, and resident behaviors, thereby accelerating simulations, facilitating automatic labeling, and eliminating the cost of physical sensors. This extensive dataset can then be utilized for pre-training machine learning models. Furthermore, a digital twin can aid in evaluating the correct positioning and selection of sensors to recognize a predefined list of activities.

Digital twin models have gained significant interest in various application domains, such as manufacturing, aerospace, healthcare, and medicine [13]. While digital twins for smart homes are relatively less explored, digital twins for buildings have been studied extensively. Ngah Nasaruddin et al. [14] define a digital twin of a building as the interaction between the interior environment of a real building and a realistic virtual representation model of the building environment. This digital twin enables real-time monitoring and data acquisition. For example, digital twins of buildings have been utilized in [15] to determine the strategic locations of sensors for efficient data collection.

1.3. Contributions

The gap between training and testing data in HAR for smart homes presents significant challenges due to the variability and sparsity of activity traces. In this study, we address this issue by exploring the possibility of generating data suitable for deployment scenarios.

Our contributions are as follows:

- We propose a novel approach that paves the way for digital twins in the context of smart homes.
- We enhance the Virtual Home [48] video-based data simulator to support sensor-based data simulation for smart homes, which we refer to as VirtualSmartHome.
- We demonstrate, through an illustrative example, that we can replicate a real apartment to generate data for training an ADL classification algorithm.
- Our study validates the effectiveness of our approach in generating data that closely resembles real-life scenarios and enables the training of an ADL recognition algorithm.
- We outline a tool and methodology for creating digital twins for smart homes, encompassing a simulator for ADLs in smart homes and a replicable approach for modeling real-life apartments and scenarios.
- The proposed tool and methodology can be utilized to develop more effective ADL classification algorithms and enhance the overall performance of smart home systems.

In the next section (Section 2), we provide a comprehensive review of the state-of-the-art approaches in HAR algorithms, ADL datasets, and home simulators. Subsequently, in Section 3, we introduce the VirtualSmartHome simulator that we have developed, along with our methodology for replicating real apartments and human activities. Moving forward, in Section 4, we present an evaluation of our simulator, comparing the synthetic data produced by the VirtualSmartHome simulator with real data from a smart apartment. We also demonstrate the potential of our approach by employing the generated datasets for a HAR algorithm.

2. Related Work

While recent HAR algorithms have demonstrated improved recognition rates when trained and tested on the same households, their generalizability across different households remains limited. The existing ADL datasets also have their own limitations, prompting the exploration of smart home simulators to generate relevant test data. In this section, we discuss the limitations of current HAR algorithms and ADL datasets, and review the available home simulators.

2.1. Machine Learning Algorithms for Activity Recognition based on Smart Home IoT Data

Numerous methods and algorithms have been studied for HAR in the smart home domain. Early approaches utilized machine learning techniques such as Support Vector Machines (SVM), Naive Bayes networks, or Hidden Markov Models (HMM), as reviewed in [16]. However, these models lack generalization and adaptability, as they are designed for specific contexts and rely on hand-crafted features, which are time-consuming to produce and limit the models' generalization and adaptability.

More recently, deep learning techniques have emerged as a promising approach due to their ability to serve as end-to-end models, simultaneously extracting features and classifying activities. These models are predominantly based on Convolutional Neural Networks (CNN) or Long Short-Term Memory (LSTM).

CNN structures excel at feature extraction and pattern recognition. They have two key advantages for HAR. Firstly, they can capture local dependencies, meaning they consider the significance of nearby observations that are correlated with the current event. Secondly, they are scale-invariant, capable of handling differences in step frequency or event occurrence. For example, Gochoo et al. [17] transformed activity sequences into binary images to leverage 2D CNN-based structures. Singh et al. [18] applied a 1D CNN structure to raw data sequences, demonstrating their high feature extraction capability. Their experiments demonstrated that the CNN 1D architecture yields comparable results to LSTM-based models while being more computationally efficient. However, LSTM-based models still outperform the CNN 1D architecture..

LSTM models are specifically designed to handle time sequences and effectively capture both long and short-term dependencies. In the context of HAR in smart homes, Liciotti et al. [5] extensively investigated various LSTM structures and demonstrated that LSTM surpasses traditional HAR approaches in terms of classification scores without the need for handcrafted features. This superiority can be attributed to LSTM's ability to generate features that encode temporal patterns, as highlighted in [19] when compared to conventional machine learning techniques. As a result, LSTM-based structures have emerged as the leading models for tackling the challenges of HAR in the smart home domain.

2.2. Activities of Daily Living Datasets

Unfortunately, in order for a deep learning model to achieve sufficient generalization, a large amount of high-quality field data is required.

Several public real home datasets [7–10,20] are available, and an overview of sensor-based datasets used in HAR for smart homes is provided by De-La-Hoz-Franco *et al.* [21]. However, these datasets have limitations in terms of the number of activities and occupants considered, sensor usage, and the type of residents.

The currently available public datasets are insufficient to cover the probability distribution of all possible HAR data and enable algorithms to generalize to application test data.

Table 1. Cost of CASAS components: "smart home in a box" [20]

Components	Server	Light & Motion Sensors	Door Sensors	Relay	Temperature Sensors	**Total Cost**
Unit Price	$350	$85	$75	$75	$75	
Quantity	1	24	1	2	2	
Total Price	$350	$2,040	$75	$150	$150	**$2,765**

Moreover, collecting new datasets is a costly and challenging task. For instance, Cook *et al.* [20] developed a lightweight and easy-to-install smart home design called "a smart home in a box," aimed at efficiently collecting data once installed in the environment. However, the cost of components for the "smart home in a box" project [20], as shown in Table 1, amounted to $2,765 in 2013. Although the cost of sensors has

decreased over time, there are still challenges in accessing various types of houses and inhabitants. Additionally, collecting data from real inhabitants is time-consuming, and the recording cannot be accelerated like in a simulation. Furthermore, the data requires ground truth, including data stream segmentation (start and end time) and class labels, which necessitates significant user investment and is prone to errors due to manual annotations, as described in [2].

Considering these challenges, researchers suggest generating data using simulation environments.

The objective of simulators in the context of smart homes is to generate simulated data from domestic sensors that accurately reflect real-world scenarios. The production of synthetic data through simulation offers several advantages, including the ability to: 1) collect perfectly controlled ground truth data, and 2) introduce diversity in terms of environments and living habits.

2.3. Existing Home Simulators

In the field of HAR, the collection of data and the creation of physical environments with sensors pose significant challenges, including sensor positioning, installation, and configuration. Additionally, the collection of datasets is often limited by ethical protocols and user participation.

Table 2. Comparison of simulation environments : table indicating for each simulator : its approach (model-based, interactive or hybrid), its development environment, the language of the API, the number or name of the apartment recorded, its apartment designer / editor, its application and its output (videos or sensor logs); whether it is open source, is multi-agent, records objects, uses activity scripts, provides a 3D visualisation; and whether it records IoT Sensors and their numnert

Simulators	Open	Approach	Multi	Environment	API	Apartment	Objects	Scripts	IoT Sensors	Designer/Editor	Visual.	Application	Output
AI2Thor [22]	Yes	Model	Yes	Unity	Python	17	609	No	No	Yes	3D	Robot Interaction	Videos
iGibson [23]	Yes	Model	Yes	Bullet Python	No	15	570	No	1	Yes	None	Robot Interaction	Videos
Sims4Action [24]	No	Model	Yes	Sims 4	No	None	NA	No	No	Game Interface	3D	Human Activity	Videos
Ai Habitat [25]	Yes	Model	Yes	C++	Python	None	NA	No	No	Yes	None	Human Activity	Sens. Log
Open SHS [26]	Yes	Hybrid	No	Blender	Python	None	NA	No	29 (B)	With Blender	3D	Human Activity	Sens. Log
SESim [27]	Yes	Model	No	Unity	NA	NA	Yes	Yes	5	Yes	3D	Human Activity	Sens. Log
Persim 3D [28]	No	Model	No	Unity	C#	Gator Tech	Yes	No	Yes (B)	Yes	3D	Human Activity	Sens. Log
IE Sim [29]	No	Hybrid	No	NA	NA	NA	Yes	No	Yes (B)	Yes	2D	Human Activity	Sens. Log
SIMACT [30]	Yes	Model	No	JME	Java	3D kitchen	Yes	Yes	Yes (B)	With Sketchup	3D	Human Activity	Sens. Log
Park et al [31]	No	Interactive	No	Unity	NA	1	Yes	No	NA	With Unity	3D	Human Activity	Sens. Log
Francillette et al [32]	Yes	Hybrid	Yes	Unity	NA	NA	Yes	Yes	8 (B and A)	With Unity	3D	Human Activity	Sens. Log
Buchmayr et al [33]	No	Interactive	No	NA	NA	NA	Yes	No	Yes (B)	NA	3D	Human Activity	Sens. Log
Armac et al. [34]	No	Interactive	Yes	NA	NA	None	Yes	No	Yes (B)	Yes	2D	Human Activity	Sens. Log
VirtualHome [35]	Yes	Hybrid	Yes	Unity	Python	7	308	Yes	No	Yes	3D	Human Activity	Videos

Simulation platforms have been widely used in various domains, and there has been a recent surge of interest in developing simulators to represent indoor environments, as highlighted by Golestan *et al.* [36], see Table 2. Simulation tools offer scalability, flexibility, and extensibility, making them suitable for a wide range of contexts and large-scale applications. They enable rapid, repeatable, and cost-effective prototyping of applications. For example, Bruneau *et al.* [38] demonstrated that users with basic software development skills can simulate a simple indoor environment in just one hour on average, whereas a similar task would require much more time in the real world.

Some research focuses on providing photorealistic representations of environments to train computer vision models [25,39,40]. Other simulators incorporate actionable objects, allowing agents to learn manipulation and interaction with objects. These advancements have generated growing interest in studying embedded intelligence in home environments and building robots capable of performing household tasks, following instructions, or collaborating with humans at home [41–43].

Designing and creating indoor simulation environments is not a straightforward task. Numerous parameters and variabilities that impact daily life must be considered, such as the type and structure of the house and external events like changes in temperature or lighting throughout the day or across seasons. The development of such simulation environments is expensive and time-consuming. As a result, recent research has explored the use of video game platforms that offer sophisticated and successful

simulation environments. For instance, Roitberg *et al.* [24], Cao *et al.* [44] utilized these environments to generate video data of human activities. However, the use of such environments is limited due to their closed and proprietary nature, making it challenging to incorporate additional functionalities.

Other works have introduced specialized simulation platforms focused on human modeling, providing better control over activities [45]. According to Synnott *et al.* [46], there are two approaches to designing smart home environment simulators: interaction-based and model-based approaches.

Interactive approaches involve a virtual environment (2D or 3D scenario) where users can act as avatars, interacting with the environment. Most papers in the field adopt this approach for agent modeling because it offers greater variation in agent traces compared to model-driven approaches, especially when a sufficient number of users interact with the simulators.

For example, Park et al. [31] proposed CASS, a simulator that helps designers detect inconsistencies in a defined set of rules related to sensor readings, occupants' locations, and actuators. Users interact with the simulator through an interface to manipulate the simulated environment.

Buchmayr et al. [33] presented a simulator that models binary sensors (e.g., contact switches, motion and pressure sensors, temperature sensors) and incorporates faulty sensor behaviors by introducing noise signals to the readings. Users can generate agent traces by interacting with any sensor through a user interface.

Armac and Retkowitz [34] developed a simulator for residential buildings to generate synthetic ADL datasets. The simulator requires designers to define accessible and inaccessible areas (obstacles) and place devices in an indoor environment. Users can interact with virtual agents to

produce agent traces by interacting with environmental objects.

Interactive approaches offer accurate and realistic simulations since each action or movement is performed by a real human. However, generating large amounts of data through interactive approaches can be costly and requires significant effort from users. Thus, these approaches are typically suitable for testing single activities or short runs.

On the other hand, model-based approaches involve specifying a reference model for the simulation. The level of abstraction in defining activities determines the precision of the modeled behavior. Kamara et al. [47] argued that this method is sufficient for generating both ADL traces in residential settings and office routines in public buildings.

Bouchard et al. [30] proposed SIMACT, a simulator that allows third-party components to connect to the simulator's database to receive and store real-time sensor readings. The simulator includes a set of pre-recorded scenarios (agent traces) to ensure data consistency. Users can also define their own scenarios using an XML file. However, SIMACT does not provide a multi-agent environment.

Ho et al. [27] introduced Smart Environment Simulation (SESim), a simulator designed to generate synthetic sensor datasets. The simulator underwent three validation phases: the creation of a smart environment, analysis of the generated data, and training of an activity recognition algorithm using a multi-layer neural network. The algorithm achieved an average recognition accuracy of 83.08% and F1-score of 66.17%. However, the evaluation was conducted solely on synthetic data generated by the simulator.

Lee et al. [28] introduced Persim-3D, a context-driven simulator implemented in Unity 3D. The simulator represents agent activities as sequences of actions and incorporates contexts that determine the conditions under which specific activities can be performed. To assess the external validity of synthetic datasets, the authors compared the data generated by the simulator with real-world data collected from the Gator Tech Smart House (GTSH) [**?**] and reported an 81% similarity. However, the authors did not evaluate the performance of HAR algorithms using this simulator. Additionally, the current version of the simulator only supports simulation of a single user activity.

More recently, hybrid approaches have emerged, combining both model-based and interactive approaches in a single simulator [26,29,32,48]. These approaches offer the advantages of both methods.

Alshammari et al. [26] proposed OpenSHS, a simulator for ADL dataset generation. Designers can use Blender 3D to create the space and deploy devices, and users can control an agent with a first-person view to generate agent traces. The simulator records sensor readings and states based on user interactions. It also supports script-based actions in the environment. However, the project does not appear to be actively updated or used.

Francillette et al. [32] developed a simulation tool capable of modeling the behavior of individuals with Mild Cognitive Impairment (MCI) or Alzheimer's Disease (AD). The simulator allows manual control or modeling of an agent based on a behavior tree model with error probabilities for each action. The authors demonstrated that their simulator accurately emulates individuals with MCI or AD when actions have different error probabilities.

Synnott et al. [29] introduced IE Sim, a simulator capable of generating datasets associated with normal and hazardous scenarios. Users can interact with the simulator through a virtual agent to perform activities. The simulator provides an object toolbox with a wide range of indoor objects and sensors, allowing users to create new objects as well. IE Sim collects sensor readings throughout the simulation. The authors demonstrated that the simulator's data can be used to detect hazardous activities and overlapping activities. IE Sim combines interactive and agent modeling approaches.

Puig et al. [48,49] proposed the VirtualHome simulator, a multi-agent platform for simulating activities in a home. Humanoid avatars represent the agents, which can interact with the environment using high-level instructions. Users can also control agents in a first-person view to interact with the environment. This simulator supports video playback of human activities and enables agent training for complex tasks. It includes a knowledge base that provides instructions for a wide range of activities.

The VirtualHome simulator aligns with our requirements for recognizing activities in a house. Although some common human actions are not yet implemented, such as hovering or eating, the extensible programming of the simulator allows for modifications. Furthermore, the simulator facilitates the reproduction of human activity scenarios, retrieval of sensor states, and replication of a real smart apartment for a digital twin. It is an ongoing project with an active community.

3. Virtual Smart Home: The Simulator

We present Virtual Smart Home, a simulator designed for modeling activities of daily living in smart homes by adapting the VirtualHome simulator [48] to log sensor activations in a smart home environment. To assess its realism, we compare the simulated activities with a multi-user dataset recorded in a real-life living lab.

3.1. Design of Virtual Smart Home

After reviewing the available simulators discussed in Section 2.3, we have selected VirtualHome [48] as the foundation for our smart home simulator. Originally developed for computer vision algorithms, VirtualHome is a multi-agent platform designed to simulate activities in a house or apartment. It utilizes humanoid avatars that can interact with their environment and perform activities based on high-level instructions. The simulator incorporates a knowledge base that enables the creation of videos depicting human activities, as well as training agents to perform complex tasks. Additionally, it provides furnished flats for simulation purposes (see Figure 1).

VirtualHome is developed on the Unity3D game engine, which offers robust kinematic, physics, and navigation models. Moreover, users can take advantage of the vast collection of 3D models accessible through Unity's Assets store, providing access to a diverse range of humanoid models.

Figure 1. Virtual-Home apartment scenes

Moreover, VirtualHome offers a straightforward process for adding new flats by utilizing the provided Unity project [49]. Each environment in VirtualHome represents an interior flat with multiple rooms and interactive objects. The configuration of each flat scene is stored in a *.json* file, which contains nodes representing each object and their relationships with other objects (specified as *"edge labels"*). For instance, the label *"between"* can be used to describe the relationship between rooms connected by a door. By modifying these description files, users can easily add, modify, or remove objects, enabling the creation of diverse scenes for generating videos or training agents.

Another notable feature of VirtualHome is its capability to create custom virtual databases within specific environments, with a supportive community that contributes new features. In a study conducted by Liao et al. [50], VirtualHome was utilized to generate a novel dataset. The researchers expanded the original VirtualHome database by incorporating additional actions for the agents and introducing programs. These programs consist of predefined sequences of instructions that can be assigned to agents, enabling them to perform activities within their simulated environment.

In VirtualHome, an avatar's activity is represented by a sequence of actions, such as *"<char0> [PutBack] <glass> (1) <table>"*, as described in [51]. This flexible framework facilitates the training of agents to engage in various everyday activities.

The authors successfully collected a new dataset [50] based on VirtualHome [35], encompassing 3,000 daily activities. Furthermore, they expanded the database by incorporating over 30,000 programs, offering a wide range of actions and possibilities. Additionally, the researchers graphed each environment, which consisted of an average of 300 objects and 4,000 spatial relationships.

Using VirtualHome, users can create scenarios where 3D avatars perform daily activities, with the ability to capture the simulated actions through a virtual camera. Moreover, the simulator enables the replication of flats, facilitating the creation of digital twins of apartments. However, it is important to note that VirtualHome does not support the acquisition of data through home automation sensors.

In order to enhance the ability to reproduce scenarios and collect data from ambient sensors, we have implemented several new features in our simulator:

1. Interactive Objects: While VirtualHome already offers a variety of objects for inclusion in apartments, many are passive and non-interactive. To address this limitation, we added the functionality to interact with some new objects. Agents can now open trash cans, drawers of column cabinets, and push on toilet faucets.

Objects with doors are implemented by splitting them into two parts—one static and one capable of rotation around an axis to simulate interaction. Fluid objects like toilet faucets are simulated by placing the origin point of the fluid at its supposed source.

2. Simulation Time Acceleration: To generate a large volume of data quickly, we implemented the ability to accelerate simulation times. This feature utilizes the Time.timeScale function of the Unity game engine. However, the acceleration cannot surpass the rendering time of the Unity game engine, resulting in a maximum fourfold increase in simulation speed.
3. Real-Life Apartment Replication and Room Creation: To replicate a real-life apartment, we propose a methodology that involves creating a 2D map of the flat using tools like Sweet Home 3D [52]. This map is then reproduced in VirtualHome, respecting the hierarchical object structure imposed by the simulator. Finally, the interactive objects are placed in a manner similar to their real-world counterparts. We demonstrated the effectiveness of this method by replicating a real-life intelligent apartment based on our room dimension measurements. Additionally, we have introduced the ability to create new rooms, such as outdoor and entrance areas.
4. IoT Sensors: While VirtualHome previously focused on recording activities using videos, we have implemented IoT sensors to enhance the simulation. The following sensors have been incorporated: (1) opening/closing sensors, (2) pressure sensors, (3) lights, (4) power consumption, and (5) zone occupancy sensors. Except for the zone occupancy sensors, all other sensors are simulated using the environment graph of the scene. This graph lists all objects in the scene with their corresponding states (e.g., closed/open, on/off). The zone occupancy sensor takes the form of a sensitive floor, implemented using a raycast. It originates from the center of the avatar and is directed downwards. The floor of the flat is divided into rooms, and the intersection with the floor identifies the room in which the avatar is located.
5. Simulation Interface: We have developed an interface that allows users to launch simulations by specifying the apartment, ambient sensors, scenarios, date, and time. The interface facilitates the scripting of each labeled activity for reproduction in the simulation. It provides three main functions: (1) creation of an experiment configuration file, where the simulation flat and desired sensor data can be chosen; (2) creation of a scenario configuration file, offering choices such as experiment date, simulation acceleration, and various activities with their durations; (3) association of an experiment configuration file with a scenario configuration file and the subsequent launch of the simulation. This functionality enables the storage of synthetic sensor logs in a database file and provides a comprehensive record of the conducted experiment, including the experiment configuration file and the scenario configuration file.

3.2. Assessing the Simulator through Dataset Creation

Our objective is to demonstrate the Virtual Smart Home simulator's ability to generate realistic data that can be utilized for training Human Activity Recognition (HAR) algorithms. To achieve this, we undertook the following steps:

Firstly, we recorded Activities of Daily Living (ADLs) performed by volunteers in our physical smart apartment. This allowed us to collect real-world data that served as the ground truth for our assessment.

Next, we replicated our smart apartment within the Virtual Smart Home simulator, creating a digital twin of the environment. This virtual representation ensured an accurate simulation of the apartment's layout and characteristics.

Subsequently, we programmed the avatar within the simulator to replicate the ADLs performed by the volunteers. This process involved instructing the virtual agent

to mimic the actions and interactions observed during the recording phase. As a result, we generated synthetic sensor logs that mirrored the behaviors of the real-world ADLs.

Finally, we conducted an evaluation to compare the replicated data generated by the simulator with the original real-world data. This assessment aimed to measure the similarity and fidelity of the simulated data, providing insights into the effectiveness of the Virtual Smart Home simulator.

3.3. Assessing the Simulator through Dataset Creation

Our objective is to demonstrate the Virtual Smart Home simulator's ability to generate realistic data that can be utilized for training HAR algorithms. To achieve this, we undertook the following steps:

Firstly, we recorded ADLs performed by volunteers in our physical smart apartment. This allowed us to collect real-world data that served as the ground truth for our assessment.

Next, we replicated our smart apartment within the Virtual Smart Home simulator, creating a digital twin of the environment. This virtual representation ensured an accurate simulation of the apartment's layout and characteristics.

Subsequently, we programmed the avatar within the simulator to replicate the ADLs performed by the volunteers. This process involved instructing the virtual agent to mimic the actions and interactions observed during the recording phase. As a result, we generated synthetic sensor logs that mirrored the behaviors of the real-world ADLs.

Finally, we conducted an evaluation to compare the replicated data generated by the simulator with the original real-world data. This assessment aimed to measure the similarity and fidelity of the simulated data, providing insights into the effectiveness of the Virtual Smart Home simulator.

3.3.1. The Smart Apartment

To capture ground truth sensor data, our dataset was created in our smart apartment called Experiment'Haal [53]. This dedicated studio serves as a testbed for experimental assistive devices and enables user testing in a simulated domestic environment. It allows us to validate technological solutions proposed by end-users within their daily life context before deployment on a larger scale.

Our smart apartment is equipped with various smart sensors and a ceiling-mounted fisheye camera. The camera records and observes the ADLs performed by the volunteers, providing visual data for analysis. The smart sensor array includes:

1. A sensitive floor that tracks the movement or presence of a person in different areas.
2. Motion sensors placed strategically to detect motion in specific locations.
3. Magnetic open/close sensors to monitor the status of doors or windows.
4. Smart lights that can be controlled remotely or manually adjusted by switches.
5. Smart plugs to monitor the status and power consumption of various devices such as the stove, oven, kettle, outlets, and television.
6. Pressure sensors installed on the bed and sofa to detect occupancy.

Figure 2 provides a detailed overview of the smart apartment layout and the positions of the sensors.

It is important to note that the sensors in the apartment come from different brands, resulting in heterogeneity and potential communication challenges. To address this, we implemented the xAAL solution [54]. This solution serves as a universal bus and acts as an IP-based smart home hub protocol, enabling us to collect data and events from devices in a unified manner.

To facilitate data collection and analysis, we divided the open space of the apartment into six distinct rooms, as shown in Figure 3. The SensFloor [55], functioning as a presence detector, was utilized to monitor each room. This arrangement allowed us

to capture not only the presence of a person in a specific room but also the transitions between rooms.

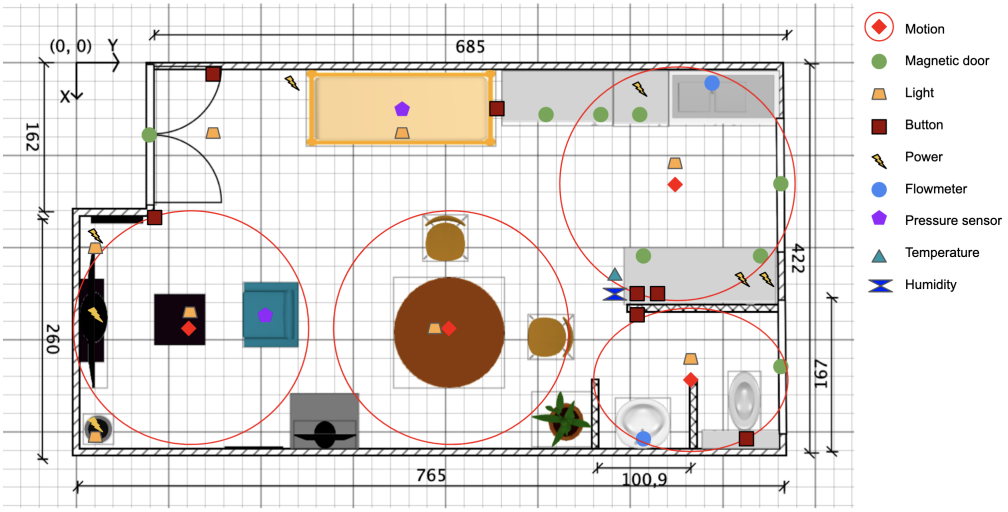


Figure 2. Layout showing the positions of the different sensors in the smart apartment

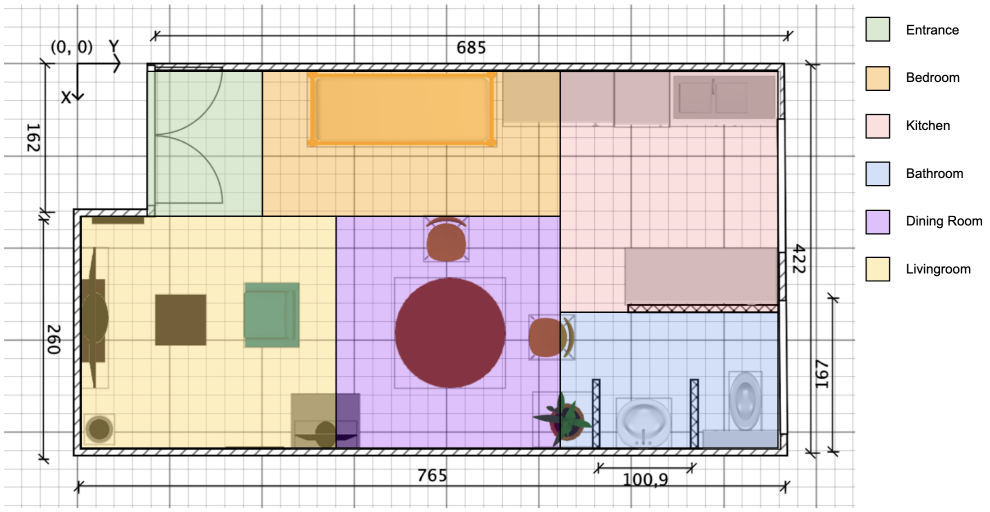


Figure 3. Room localization within the smart apartment

3.3.2. The Ground Truth Dataset

To generate the ground truth dataset, we asked volunteers to perform three different scenarios:

1. Morning scenario: This scenario involved getting out of bed and going through a typical morning routine before heading to work.
2. Lunchtime scenario: In this scenario, participants returned home from work, prepared and ate lunch, and then went back to work.
3. Evening scenario: This scenario focused on participants returning home for the evening until it was time to go to bed.

For these scenarios, we defined a list of 17 possible activity labels, including “Bathe”, “Cook breakfast”, “Cook dinner”, “Cook lunch”, “Dress”, “Eat breakfast”, “Eat dinner”, “Eat lunch”, “Enter home”, “Go to toilet”, “Leave home”, “Read”, “Sleep”, “Sleep in Bed”, “Wash dishes”, and “Watch TV”. These labels were inspired by existing literature datasets such as CASAS [20] and the ADL definitions provided by Katz et al. [56].

We recorded a total of 511 activity sequences with 8 different volunteers across the three scenarios: 158 in the morning, 154 at noon, and 199 in the evening. Tables 3, 4, and 5 show the number of activities performed by the volunteers in each scenario. Table 6 provides a global summary of the generated dataset.

Table 3. Summary of all recorded data from morning scenarios

Activity	Subject 1	Subject 2	Subject 3	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Total / Activity
Bathe	5	3	3	3	1	1	0	0	**16**
Cook	5	4	4	7	2	2	0	2	**26**
Dress	6	4	4	1	1	1	0	2	**19**
Eat	5	4	3	3	1	2	0	1	**19**
Enter Home	0	0	0	1	0	0	0	0	**1**
Go To Toilets	5	4	3	1	1	1	0	1	**16**
Leave Home	5	4	2	2	1	2	0	1	**17**
Read	0	0	2	2	0	0	0	0	**4**
Sleep	5	4	4	0	1	2	0	1	**17**
Sleep in Bed	0	0	0	2	0	0	0	0	**2**
Wash Dishes	5	4	3	0	1	2	0	1	**16**
Watch TV	0	0	0	3	1	0	0	1	**5**
Total / Subject	**41**	**31**	**28**	**25**	**10**	**13**	**0**	**10**	**158**

Table 4. Summary of all recorded data from mid-day scenarios

Activity	Subject 1	Subject 2	Subject 3	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Total / Activity
Bathe	5	0	1	3	0	0	0	0	**9**
Cook	7	6	1	3	1	4	2	4	**28**
Dress	7	2	0	1	0	0	0	1	**11**
Eat	5	2	1	3	0	2	1	2	**16**
Enter Home	5	2	1	2	1	2	1	2	**16**
Go To Toilets	8	4	1	4	1	0	0	2	**20**
Leave Home	5	1	1	2	1	2	1	2	**15**
Read	1	3	1	2	0	0	0	0	**7**
Sleep	2	0	0	1	0	0	0	0	**3**
Sleep in Bed	4	0	0	0	0	0	0	0	**4**
Wash Dishes	6	2	1	2	1	2	1	2	**17**
Watch TV	0	2	0	0	1	4	1	0	**8**
Total / Subject	**55**	**24**	**8**	**23**	**6**	**16**	**7**	**15**	**154**

Table 5. Summary of all recorded data from the evening scenarios

Activity	Subject 1	Subject 2	Subject 3	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Total / Activity
Bathe	6	2	3	5	0	1	0	2	**19**
Cook	6	5	2	5	0	3	0	3	**24**
Dress	8	2	1	4	1	2	1	0	**19**
Eat	4	2	2	4	0	1	0	2	**15**
Enter Home	5	2	2	3	1	2	1	2	**18**
Go To Toilets	10	3	1	4	0	2	0	2	**22**
Leave Home	0	0	0	0	0	0	0	0	**0**
Read	5	1	0	0	0	1	0	0	**7**
Sleep	4	2	1	3	1	2	1	2	**16**
Sleep in Bed	5	2	3	3	1	2	1	2	**19**
Wash Dishes	6	2	2	2	0	1	0	2	**15**
Watch TV	4	3	2	7	1	3	1	4	**25**
Total / Subject	**63**	**26**	**19**	**40**	**5**	**20**	**5**	**21**	**199**

3.3.3. The Synthetic Dataset

To replicate our real-life database in the Virtual Smart Home simulator, we first recreated our living lab within the simulator. The objective was to create a digital twin of the physical space, ensuring that each object and room was accurately represented with corresponding dimensions. A visual comparison between the real living lab and its virtual representation is shown in Figure 4.

Next, we utilized our implementation of IoT sensors in the Virtual Smart Home simulator to incorporate the connected sensors present in the living lab. We equipped the different objects in the virtual environment with these sensors. Specifically, we outfitted the floors representing the various rooms with person detection sensors to simulate our sensitive floor (SensFloor) as depicted in Figure 4.

Table 6. Summary of all recorded data from all scenarios

Activity	Subject 1	Subject 2	Subject 3	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Total / Activity
Bathe	16	5	7	11	1	2	0	2	**44**
Cook	18	15	7	15	3	9	2	9	**78**
Dress	21	8	5	6	2	3	1	3	**49**
Eat	14	8	6	10	1	5	1	5	**50**
Enter Home	10	4	3	6	2	4	2	4	**35**
Go To Toilets	23	11	5	9	2	3	0	5	**58**
Leave Home	10	5	3	4	2	4	1	3	**32**
Read	6	4	3	4	0	1	0	0	**18**
Sleep	11	6	5	4	2	4	1	3	**36**
Sleep in Bed	9	2	3	5	1	2	1	2	**25**
Wash Dishes	17	8	6	4	2	5	1	5	**48**
Watch TV	4	5	2	10	3	7	2	5	**38**
Total / Subject	**159**	**81**	**55**	**88**	**21**	**49**	**12**	**46**	**511**

Figure 4. (a) View of the living lab from a fisheye camera, (b) Representation in the Virtual Home Simulator

The synthetic dataset was created by replicating the recorded scenarios from the living lab within the simulator. Several post-processing steps were performed, including renaming the sensors and removing certain sensors (e.g., CO_2 sensors, WiFi, radio level sensors, noise sensor) from the real sequences that could not be implemented in VirtualHome or were not relevant for our project. We also binarized the activation values of devices such as the TV or the oven. While our real-life sensors provided power consumption values, we transformed them into *ON* or *OFF* states for simplicity in the virtual dataset.

4. Comparison Between Synthetic Data and Real Data

To assess the similarity between the synthetic data generated by the Virtual Smart Home simulator and the log sequences obtained from real-life recordings, we employed two comparison methods.

Firstly, we compared the frequency of sensor activations in the synthetic data with that of the real data. This initial analysis provided insights into the number of sensors triggered during activities and allowed for a comparison between the two datasets.

Secondly, we utilized cross-correlation [57] to measure the similarity between the log sequences from the real-life and synthetic data. Cross-correlation is a statistical measure that calculates the correlation between two sequences at different time lags.

To validate the synthetic log sequences, we implemented an activity recognition algorithm proposed by [5]. This algorithm consists of an Embedding layer, a Bi-Directional LSTM (Long Short-Term Memory) layer [59], and a Softmax layer for classification. The Embedding layer transforms integer values into vectors, while the Bi-Directional LSTM is a recurrent network layer that processes time steps in both the forward and backward directions. This type of network is commonly used in deep learning for handling tempo-

ral time series or sequential data. We used the same parameters and training approach as described in the original paper.

We trained the algorithm using four different tests:

1. Training and validating on synthetic data using three subjects with leave-one-out-subject validation.
2. Training and validating on real data using three subjects with leave-one-out-subject validation.
3. Training the algorithm on synthetic data from one subject and validating it on the real data of the same subject.
4. Training the algorithm on synthetic data from all subjects and validating it on the real data for each subject.

We will present our results in three sections to validate our digital twin approach and draw conclusions regarding activity recognition using synthetic data.

Firstly, in Section 4.2, we will compare the triggered sensor events between the synthetic and real logs. Then, in Section 4.3, we will present the results obtained using cross-correlation to determine the similarity between the sequences of real and synthetic logs. Finally, in Section 4.4, we will showcase the results obtained with the recognition algorithm on the data collected from simulation and reality.

4.1. Comparison of Triggered Sensors in Real and Synthetic Logs for Similar Scenarios

To gain an initial understanding of the synthetic data generated, we compared the frequency of sensor activations in the synthetic data with that of the real data.

Figure 5 illustrates the comparison of the number of triggered sensors in the real dataset (in blue) and the synthetic dataset (in red) across 15 scenarios. Most scenarios showed a similar count of triggered sensors in both datasets. However, some scenarios (1, 4, 5, 9, and 13) exhibited an excess of triggered sensors in the synthetic data. Upon examining Table 7, we observed that these scenarios predominantly involved presence sensors, which detect the presence of the avatar in a specific room. The difference in sensor activations can be attributed to the fact that the real-life sensor did not always detect the volunteer, and the path chosen by the avatar in the simulator did not always match the movement of the volunteer during the recording experiment.

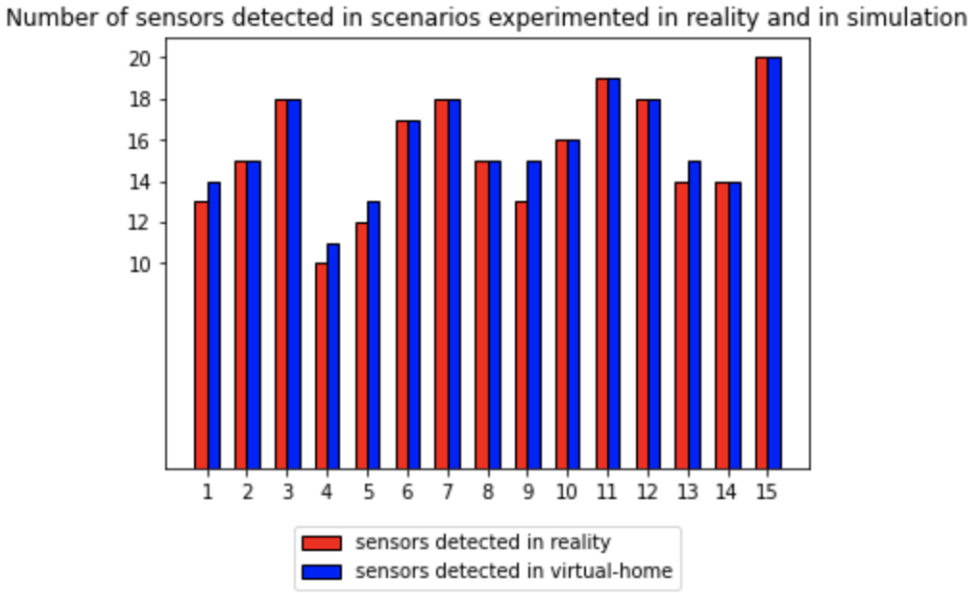


Figure 5. Comparison graph of sensors triggered in real and synthetic logs for similar scenarios

Table 7. Comparison table of sensors triggered in real and synthetic logs

Scenario Number	Real	Synthetic	Excess Sensors Detected	Index in Figure 5
s3_1_bis	13	14	floor_bedroom	1
s3_2_bis	15	15		2
s3_3_bis	18	18		3
s3_4_bis	10	11	floor_dining_room	4
s7_1	12	13	floor_bathroom	5
s7_2	17	17		6
s7_4	18	18		7
s7_5	15	15		8
s7_6	13	15	floor_bathroom, floor_bedroom	9
s9_1	16	16		10
s9_2	19	19		11
s9_3	18	18		12
s9_4	14	15	floor_bathroom	13
s9_5	14	14		14
s9_6	20	20		15

In conclusion, the comparison of triggered sensors between the real and synthetic logs for similar scenarios showed a generally close alignment. However, discrepancies were observed in scenarios, in particular for the presence sensors, which can be attributed to variations in detection and movement between the real-life recording and the simulation.

4.2. Comparison of Triggered Sensors in Real and Synthetic Logs for Similar Scenarios

In this section, we compared the frequency of sensor activations in the synthetic data with that of the real data to assess the similarity between the two datasets. The comparison focused on the number of triggered sensors in similar scenarios.

To perform this analysis, we examined the number of sensors activated in both the real and synthetic datasets across 15 scenarios. Figure 5 illustrates the comparison, with the blue bars representing the real dataset and the red bars representing the synthetic dataset. In most scenarios, the count of triggered sensors was similar between the two datasets. However, some scenarios (1, 4, 5, 9, and 13) exhibited an excess of triggered sensors in the synthetic data.

Upon further investigation (see Table 7), we discovered that these scenarios primarily involved presence sensors that detect the presence of the avatar in specific rooms. The discrepancy in sensor activations can be attributed to variations in detection and movement between the real-life recording and the simulation. In some cases, the real-life sensor did not detect the volunteer, while the simulated avatar followed a different path, resulting in additional sensor activations in the synthetic data.

Overall, the comparison of triggered sensors between the real and synthetic logs for similar scenarios demonstrated a generally close alignment. However, differences arose in scenarios involving presence sensors due to variations in detection and movement. This analysis provides valuable insights into the similarities and discrepancies between the real and synthetic datasets.

4.3. Comparison with Cross-Correlation

In this section, we utilized cross-correlation to compare the activation and order of sensors in both real and synthetic log sequences. The cross-correlation values were calculated for all sensors at each time point to evaluate the similarity between the two datasets.

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} \overline{f(m)}\, g(m+n) \tag{1}$$

The cross-correlation formulas 1 for discrete functions was used, and the cross-correlation values were calculated for all sensors and times in the sequences. To ensure a fair comparison with the longer real log sequences, we expanded the cross-correlation tables for synthetic sensors by duplicating the last line since the sensor values do not change.

To determine the similarity between the real and synthetic log sequences, we multiplied the value of each sensor in the real sequence by the corresponding synthetic sensor value. If the values matched (e.g., both *ON* or both *OFF*), a score of 1 was assigned; otherwise, a score of -1 was assigned. This calculation was performed for each sensor at each time point, resulting in the final cross-correlation table. The score was computed as the sum of all cross-correlation values for the sensors at each time.

The percentage likelihood between the two log sequences was calculated using the following formula:

$$\text{Percentage} = \frac{\text{Maximum Score}}{(\text{Number of Sensors in Reality} \times \text{Number of Events in Reality})} \times 100$$

Table 8 presents the results obtained for the 15 processed scenarios, including the similarity percentage and the average similarity across scenarios.

Table 8. Cross correlation similarity

Subject	S9						S7					S3			
Scenario Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Similarity (%)	75.73%	97.40%	75.42%	75.30%	69.61%	74.42%	81.24%	93.18%	85.22%	84.73%	88.28%	79.93%	77.53%	54.73%	82.78%
Average Similarity (%)	77.98%						86.53%					73.74%			

The obtained percentages were generally above 70%, except for one case that yielded 54.73%. Upon closer examination, we identified that the SensFloor sensor could be activated and deactivated multiple times in the same room, leading to variations in the log sequences. An example of such variations in the real log sequence is presented in Table 9, which corresponds to the scenario with the lowest similarity percentage.

In conclusion, the cross-correlation analysis revealed that the synthetic log sequences exhibited a high level of similarity to the real sequences, with similarity percentages exceeding 70% and reaching up to 86.53%. This indicates that the digital twin approach allowed us to generate synthetic data that closely resembled the real-world data. Although variations were observed in some scenarios due to the presence sensors, the overall comparison demonstrated a remarkable alignment between the two datasets. These findings suggest that the synthetic data generated through the digital twin approach can be effectively utilized for various applications, including activity recognition algorithms. In the following subsection 4.4, we will investigate whether this level of similarity is sufficient to achieve comparable performance using an activity recognition algorithm.

2021-07-12 18:01:29.112556	floor_livingroom	OFF	watch_tv
2021-07-12 18:03:33.039082	floor_livingroom	ON	watch_tv
2021-07-12 18:03:34.014050	floor_livingroom	OFF	watch_tv
2021-07-12 18:04:18.782342	floor_livingroom	ON	watch_tv
2021-07-12 18:04:19.785484	floor_livingroom	OFF	watch_tv
2021-07-12 18:04:21.030509	floor_livingroom	ON	watch_tv
2021-07-12 18:04:22.030509	floor_livingroom	OFF	watch_tv
2021-07-12 18:04:25.854238	floor_livingroom	ON	watch_tv
2021-07-12 18:04:29:084238	floor_livingroom	OFF	watch_tv
2021-07-12 18:05:02.588711	floor_livingroom	ON	watch_tv
2021-07-12 18:05:04.102735	floor_livingroom	OFF	watch_tv
2021-07-12 18:05:24.473798	floor_livingroom	ON	watch_tv
2021-07-12 18:05:26.010093	floor_livingroom	OFF	watch_tv
2021-07-12 18:06:09.667039	floor_livingroom	ON	watch_tv
2021-07-12 18:06:10.909138	floor_livingroom	OFF	watch_tv
2021-07-12 18:06:30.451480	pressure_armchair	OFF	watch_tv
2021-07-1218:06:34.380592	floor_livingroom	ON	watch_tv
2021-07-12 18:06:45.037284	floor_livingroom	OFF	watch_tv

Table 9. Activation/deactivation of the SensFloor in the real log sequence : s7_activity_6

4.4. Activity recognition

In our activity recognition tests, we did not consider the time of day. This limitation arose because we were unable to directly manage time within the VirtualHome environment. Instead, we relied on post-processing techniques to estimate action durations, which may not accurately represent real-world timing. Consequently, distinguishing between different meal times (morning, noon, or evening) during eating and cooking activities becomes challenging for an algorithm.

To address this challenge, we opted to group the specific "Eat ..." labels together under a general label, "Eat". This grouping included activities such as (1) "Eat breakfast", (2) "Eat lunch", and (3) "Eat dinner". Similarly, we combined the various cooking activities into a single label, "Cook", encompassing (1) "Cook breakfast", (2) "Cook lunch", and (3) "Cook dinner".

We conducted four different tests using the activity recognition algorithm proposed by [5], as described previously at the beginning of this section.

4.4.1. Experiments 1 and 2: Leave-One-Subject-Out Cross Validations

The objective of this step is to assess the performance of the activity recognition algorithm on both real and synthetic data. To achieve this, we employed the Leave-One-Subject-Out cross-validation method, which is a variation of Leave-One-Out cross-validation [60]. This method is commonly used for datasets with limited data or when the data are associated with distinct entities.

In this experiment, we conducted two cross-validations: one using only synthetic data and the other using only real data. With three subjects in our dataset, the cross-validation involved using one subject for testing and the remaining two subjects for training. We repeated this process by rotating the subject used for testing, enabling validation on each subject's data. Finally, we compared the results obtained from the synthetic and real Leave-One-Subject-Out cross-validations.

The results of the two cross-validations are presented in Tables 10 and 11. These results demonstrate that the algorithm can be trained with both real and synthetic data, yielding comparable outcomes. Notably, there is a high degree of similarity in the results for Subject "9". However, for the other two subjects, we observe more differences between the results of synthetic and real data. Specifically, the performance, as measured by F1-score and Balanced Accuracy, is better for Subject "7" and "3" with synthetic data.

Table 10. Experiment 1: Leave-One-Subject-Out cross-validations for real data

	Subject 9				Subject 7				Subject 3			
	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Bathe	40.00%	100.00%	57.14%	2	66.67%	100.00%	80.00%	2	100.00%	75.00%	85.71%	4
Cook	50.00%	30.00%	37.50%	10	25.00%	16.67%	20.00%	6	50.00%	100.00%	66.67%	4
Dress	100.00%	100.00%	100.00%	3	50.00%	33.33%	40.00%	3	66.67%	100.00%	80.00%	2
Eat	30.00%	50.00%	37.50%	6	12.50%	25.00%	16.67%	4	100.00%	33.33%	50.00%	3
Enter Home	100.00%	100.00%	100.00%	4	75.00%	100.00%	85.71%	3	60.00%	100.00%	75.00%	3
Go To Toilets	100.00%	20.00%	33.33%	5	100.00%	33.33%	50.00%	3	50.00%	50.00%	50.00%	2
Leave Home	100.00%	100.00%	100.00%	4	60.00%	100.00%	75.00%	3	0.00%	0.00%	0.00%	2
Sleep	100.00%	100.00%	100.00%	3	0.00%	0.00%	0.00%	4	0.00%	0.00%	0.00%	2
Wash Dishes	83.33%	100.00%	90.91%	5	100.00%	50.00%	66.67%	4	100.00%	100.00%	100.00%	4
Watch TV	100.00%	100.00%	100.00%	7	83.33%	100.00%	90.91%	5	100.00%	100.00%	100.00%	2
Accuracy	71.43%				51.35%				71.43%			
Balanced Accuracy	80.00%				55.83%				65.83%			
Macro Avg	80.33%	80.00%	75.64%	49	57.25%	55.83%	52.50%	37	62.67%	65.83%	60.74%	28
Weighted Avg	77.07%	71.43%	70.11%	49	54.19%	51.35%	49.19%	37	68.33%	71.43%	65.88%	28

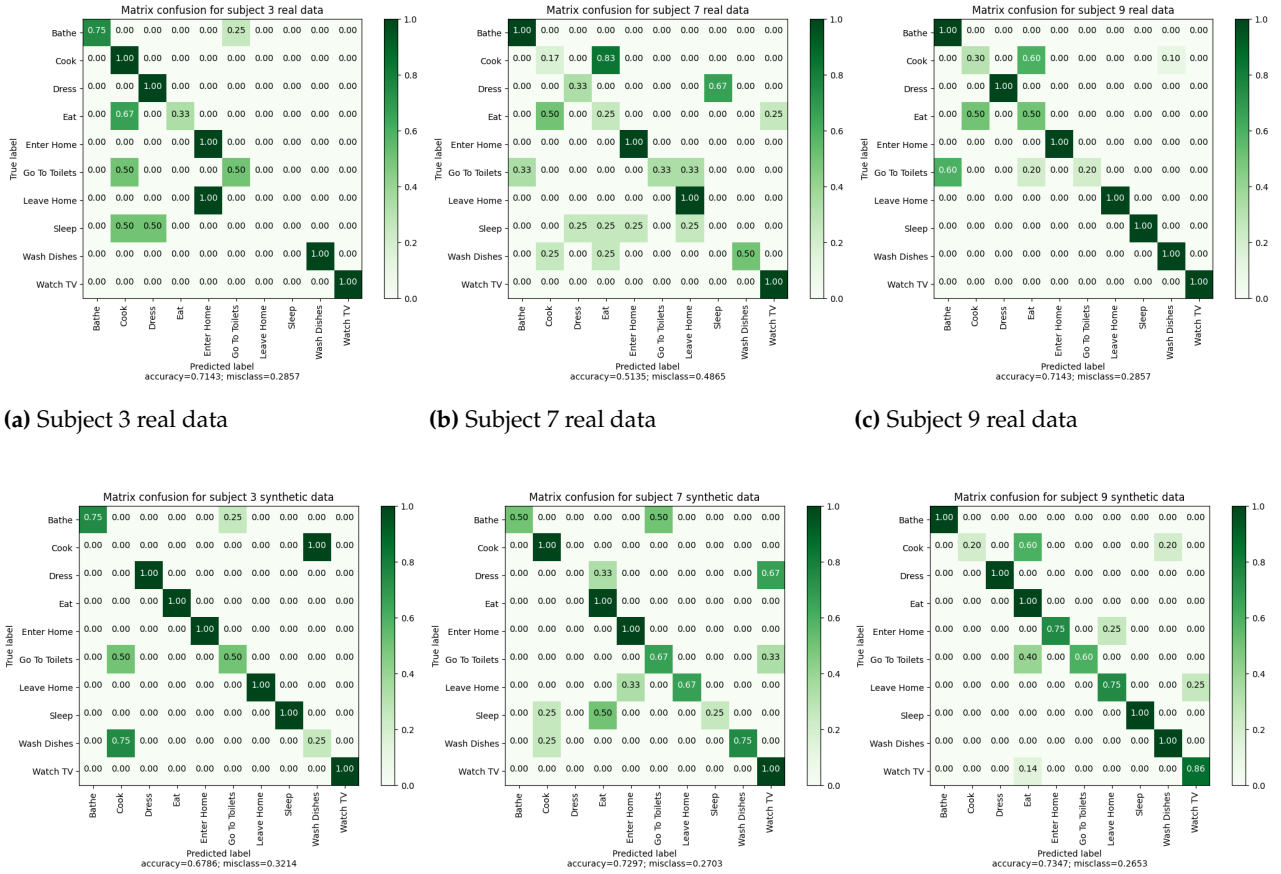
Table 11. Experiment 2: Leave-One-Subject-Out cross-validations for synthetic data

	Subject 9				Subject 7				Subject 3			
	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Bathe	100.00%	100.00%	100.00%	2	100.00%	50.00%	66.67%	2	100.00%	75.00%	85.71%	4
Cook	100.00%	20.00%	33.33%	10	75.00%	100.00%	85.71%	6	0.00%	0.00%	0.00%	4
Dress	100.00%	100.00%	100.00%	3	0.00%	0.00%	0.00%	3	100.00%	100.00%	100.00%	2
Eat	40.00%	100.00%	57.14%	6	57.14%	100.00%	72.73%	4	100.00%	100.00%	100.00%	3
Enter Home	100.00%	75.00%	85.71%	4	75.00%	100.00%	85.71%	3	100.00%	100.00%	100.00%	3
Go To Toilets	100.00%	60.00%	75.00%	5	66.67%	66.67%	66.67%	3	50.00%	50.00%	50.00%	2
Leave Home	75.00%	75.00%	75.00%	4	100.00%	66.67%	80.00%	3	100.00%	100.00%	100.00%	2
Sleep	100.00%	100.00%	100.00%	3	100.00%	25.00%	40.00%	4	100.00%	100.00%	100.00%	2
Wash Dishes	71.43%	100.00%	83.33%	5	100.00%	75.00%	85.71%	4	20.00%	25.00%	22.22%	4
Watch TV	85.71%	85.71%	85.71%	7	62.50%	100.00%	76.92%	5	100.00%	100.00%	100.00%	2
Accuracy	73.47%				72.97%				67.86%			
Balanced Accuracy	81.57%				68.33%				75.00%			
Macro Avg	87.21%	81.57%	79.52%	49	73.63%	68.33%	66.01%	37	77.00%	75.00%	75.79%	28
Weighted Avg	85.66%	73.47%	71.65%	49	73.41%	72.97%	68.19%	37	70.71%	67.86%	68.99%	28

Examining the confusion matrices in Figure 6, we observe more activity confusions when using real data. This finding suggests that real data pose greater complexity for the algorithm, likely due to sensor recording errors in the real apartment.

In general, the confusion matrices reveal certain patterns. For example, activities such as “Enter Home” and “Leave Home” are often confused, which is logical since they trigger similar types of sensors (entrance floor, principal door, etc.). Similarly, “Bathe” and “Go To Toilet” activities show confusion, likely because one may wash their hands after using the toilet, and in our apartment, the toilet is located in the same room as the bathroom. “Reading” and “Watching TV” activities can also be easily confused as they occur in the same room and trigger similar types of sensors. Additionally, “Wash Dishes” and “Cook” activities, being in the same room, are occasionally confused by the algorithm.

Comparing the confusion matrices for real and synthetic data, we observe that the aforementioned confusions occur in both cases. For instance, the activity “Read” is not correctly recognized in both synthetic and real data for Subject “3”, although the scores are slightly higher for real data. This difference can be attributed to the richness of real data, which exhibits more sensor activations in the sequence. Moreover, the avatar’s trajectory in simulation can introduce sensor activations that are not correlated with the current activity. In contrast, during the real recordings, subjects pay attention to room transitions, while the avatar does not, resulting in sensor activations from other rooms that disrupt activity sequences.



(a) Subject 3 real data **(b)** Subject 7 real data **(c)** Subject 9 real data

(d) Subject 3 synthetic data **(e)** Subject 7 synthetic data **(f)** Subject 9 synthetic data

Figure 6. Experiment 1 an 2: Leave-One-Subject-Out cross-validations confusion matrices

In conclusion, despite the logical confusions made by the algorithm, the recognition results obtained are quite similar for real and synthetic data. In the next subsection 4.4.2, we will investigate the extent to which synthetic data can be effectively used for activity recognition in our digital twin.

4.4.2. Experiment 3: One-to-One - Training on Synthetic Data and Testing on Real Data

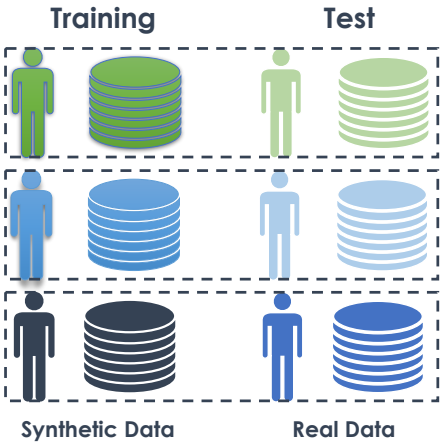


Figure 7. Experiment 3: One to One

In this experiment, we trained the model using synthetic data from a subject and tested the trained algorithm with the real data of the same subject (see details in Figure 7). The objective was to determine whether an algorithm trained on synthetic data could effectively recognize activities in real data.

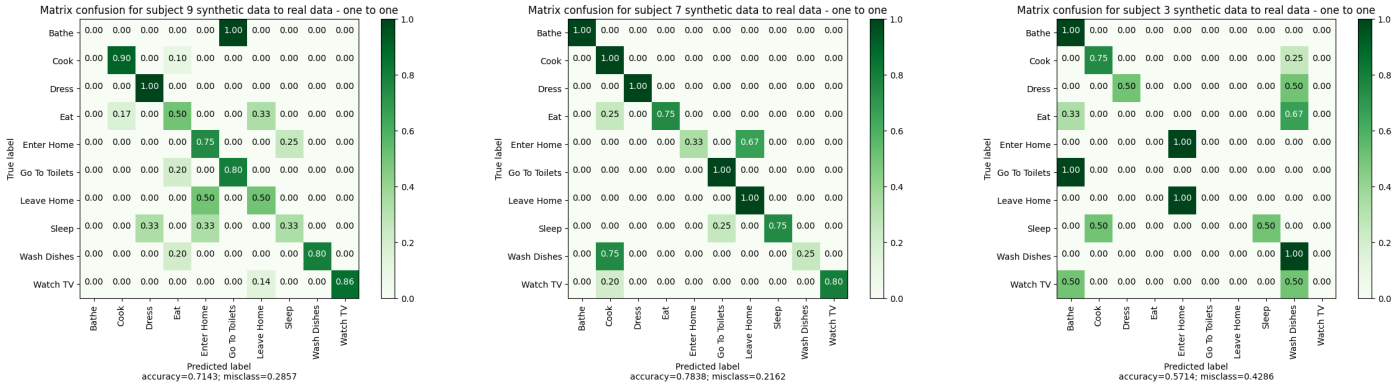
Analyzing Table 12 and Confusion Matrices 8, we can initially observe that the synthetic data generated for each subject enabled training and recognition of activity sequences for the corresponding real datasets (one subject at a time). Subjects "9" and "7" achieved good performance in terms of Accuracy, Balanced Accuracy, and F1-score. Notably, subject "7" exhibited the best performance among the subjects. For these two subjects, the synthetic data appeared realistic enough to achieve activity recognition with an accuracy of over 70% for both subjects.

In contrast, subject "3" displayed the lowest performance. It seems that the synthetic data generated for this subject were insufficient to enable accurate activity recognition. The poor performance for this subject suggests that there are differences between the synthetic and real data. A closer examination of the real data for subject "3" reveals sequences that are interfered with by sensors triggering unrelated to the activity. For example, the presence sensor on the floor is regularly triggered in the living room while subject "3" is in the kitchen. This disturbance occurs due to a sensor malfunction, detecting a second presence in the living room. Such malfunctions are not anticipated or simulated in the synthetic data.

Additionally, we observed that the activity "Bathe" was not recognized for subject "9", whereas it was recognized with 100% accuracy for subject "7". Subject "3" had four activity classes out of ten that were not recognized. These results indicate that synthetic data can be used to train an algorithm and recognize activities in real data. However, relying solely on activity data from a single subject may not always be sufficient. Furthermore, the performance can be degraded by sensor malfunctions in real conditions, which can disrupt the activity recognition algorithm. Therefore, incorporating more data and variability into the training dataset is necessary to address these challenges.

Table 12. Experiment 3: One to One - results of multiple metrics

	S9				S7				S3			
	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Bathe	0.00%	0.00%	0.00%	2	100.00%	100.00%	100.00%	2	50.00%	100.00%	66.67%	4
Cook	90.00%	90.00%	90.00%	10	54.55%	100.00%	70.59%	6	75.00%	75.00%	75.00%	4
Dress	75.00%	100.00%	85.71%	3	100.00%	100.00%	100.00%	3	100.00%	50.00%	66.67%	2
Eat	50.00%	50.00%	50.00%	6	100.00%	75.00%	85.71%	4	0.00%	0.00%	0.00%	3
Enter Home	50.00%	75.00%	60.00%	4	100.00%	33.33%	50.00%	3	60.00%	100.00%	75.00%	3
Go To Toilets	66.67%	80.00%	72.73%	5	75.00%	100.00%	85.71%	3	0.00%	0.00%	0.00%	2
Leave Home	40.00%	50.00%	44.44%	4	60.00%	100.00%	75.00%	3	0.00%	0.00%	0.00%	2
Sleep	50.00%	33.33%	40.00%	3	100.00%	75.00%	85.71%	4	100.00%	50.00%	66.67%	2
Wash Dishes	100.00%	80.00%	88.89%	5	100.00%	25.00%	40.00%	4	44.44%	100.00%	61.54%	4
Watch TV	100.00%	85.71%	92.31%	7	100.00%	80.00%	88.89%	5	0.00%	0.00%	0.00%	2
Accuracy	71.43%				78.38%				57.14%			
Balanced Accuracy	64.40%				78.83%				47.50%			
Macro Avg	62.17%	64.40%	62.41%	49	88.95%	78.83%	78.16%	37	42.94%	47.50%	41.15%	28
Weighted Avg	70.78%	71.43%	70.39%	49	87.36%	78.38%	76.91%	37	44.92%	57.14%	46.59%	28



(a) Subject 9 real data **(b)** Subject 7 real data **(c)** Subject 3 real data

Figure 8. Experiment 3: One to One - confusion matrices

4.4.3. Experiment 4: Many-to-One - Training on Synthetic Data and Testing on Real Data

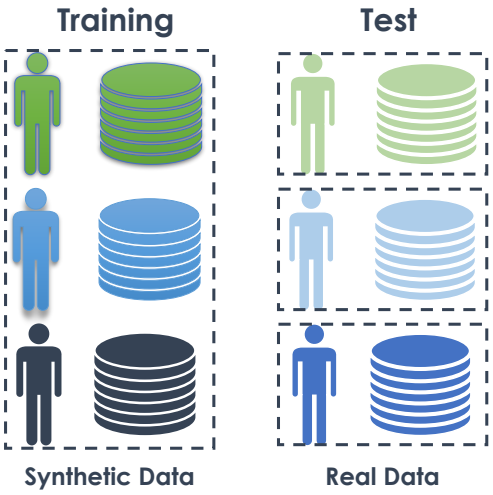


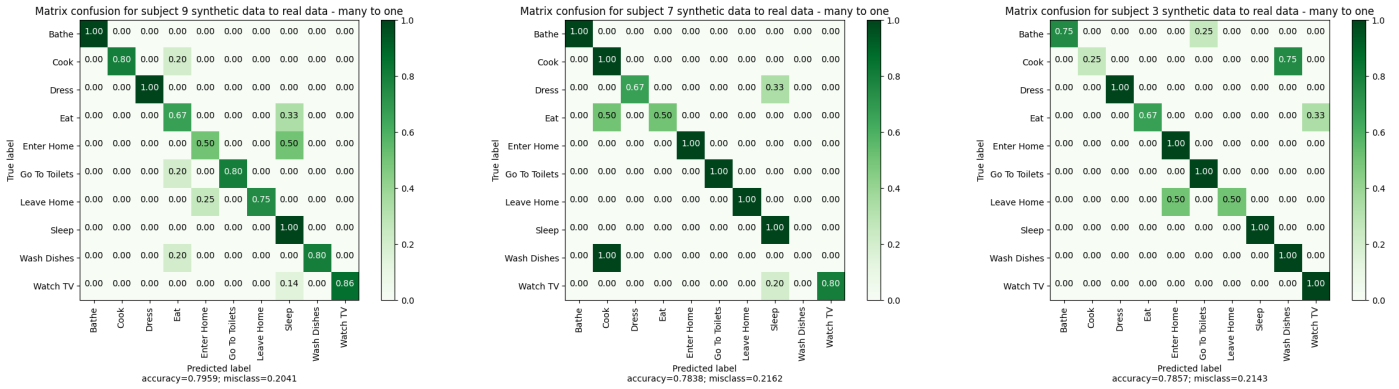
Figure 9. Experiment 4: Many to One

The objective of this experiment was to train the algorithm using all synthetic data and then test it on the real data of each subject independently (see Figure 9). The purpose was to evaluate whether increasing the amount of data provided by the synthetic data from different subjects would improve the algorithm's capabilities in recognizing daily activities. In the previous section, we observed that training an algorithm on synthetic data and testing it in real conditions can be successful. However, the quantity of training data was not very representative and still relatively close to the real data. The performance could be enhanced by presenting the algorithm with examples of different lifestyles within the same living environment, allowing for better generalization.

Table 13 demonstrates that the algorithm achieved higher classification scores (close to 80%) for all subjects compared to the previous experiment. Subject "7" maintained very similar performance to the previous experiment. However, subjects "9" and "3" showed notable improvement, particularly subject "3", which had previously exhibited the worst results. Subject "3" experienced an increase in accuracy and balanced accuracy from 57.14% and 47.50% to 78.57% and 81.67%, respectively.

Table 13. Experiment 3: Many to One - results of multiple metrics

	S9				S7				S3			
	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Bathe	100.00%	100.00%	100.00%	2	100.00%	100.00%	100.00%	2	100.00%	75.00%	85.71%	4
Cook	100.00%	80.00%	88.89%	10	50.00%	100.00%	66.67%	6	100.00%	25.00%	40.00%	4
Dress	100.00%	100.00%	100.00%	3	100.00%	66.67%	80.00%	3	100.00%	100.00%	100.00%	2
Eat	50.00%	66.67%	57.14%	6	100.00%	50.00%	66.67%	4	100.00%	66.67%	80.00%	3
Enter Home	66.67%	50.00%	57.14%	4	100.00%	100.00%	100.00%	3	75.00%	100.00%	85.71%	3
Go To Toilets	100.00%	80.00%	88.89%	5	100.00%	100.00%	100.00%	3	66.67%	100.00%	80.00%	2
Leave Home	100.00%	75.00%	85.71%	4	100.00%	100.00%	100.00%	3	100.00%	50.00%	66.67%	2
Sleep	37.50%	100.00%	54.55%	3	66.67%	100.00%	80.00%	4	100.00%	100.00%	100.00%	2
Wash Dishes	100.00%	80.00%	88.89%	5	0.00%	0.00%	0.00%	4	57.14%	100.00%	72.73%	4
Watch TV	100.00%	85.71%	92.31%	7	100.00%	80.00%	88.89%	5	66.67%	100.00%	80.00%	2
Accuracy	79.59%				78.38%				78.57%			
Balanced Accuracy	81.74%				79.67%				81.67%			
Macro Avg	85.42%	81.74%	81.35%	49	81.67%	79.67%	78.22%	37	86.55%	81.67%	79.08%	28
Weighted Avg	87.33%	79.59%	81.67%	49	77.48%	78.38%	74.89%	37	86.44%	78.57%	76.58%	28



(a) Subject 9 real data **(b)** Subject 7 real data **(c)** Subject 3 real data

Figure 10. Experiment 4: Many to One - confusion matrices

Furthermore, Table 13 and Figure 10 reveal that more activity classes were correctly identified. The introduction of additional synthetic data from other subjects within the same apartment led to improved classification performance. The contribution of data from different subjects introduced variability in the execution of activities, enabling the algorithm to better generalize and capture sensor behavior during activities. Having a diverse range of examples is crucial for training a deep learning algorithm.

In conclusion, by utilizing more synthetic data, the algorithm demonstrated increased performance in real conditions. The inclusion of behavioral variability from different subjects facilitated better generalization. This generalization resulted in significant improvements, particularly for subject "3".

4.5. Summary

The experiments conducted in this section yielded valuable insights. The results demonstrated that the simulator has the ability to generate synthetic data that closely resemble real-world data. The activity recognition algorithm performed similarly on both synthetic and real data, indicating that training the algorithm solely on synthetic data can effectively recognize activities in real-world scenarios. Moreover, when the entire set of generated synthetic data was utilized, the algorithm's performance improved for each subject. This improvement can be attributed to the increased variability and examples provided by the additional synthetic data, allowing the algorithm to better generalize and capture the behavior of sensors during different activities.

5. Conclusion

The limitations of existing datasets, such as their limited coverage of activities, lifestyles, and house configurations, pose challenges for activity recognition algorithms. These algorithms trained on specific datasets may not be applicable to different environments due to the tailored nature of their training data. Collecting new datasets is a costly and time-consuming process, requiring volunteers to live in the environment and label activities for an extended period. This presents practical difficulties and a risk of mislabeling.

To address the data scarcity challenge and bridge the gap between training data and real-world use cases, we proposed the use of a digital twin concept. By enhancing the VirtualHome simulation environment, we simulated an intelligent environment with home automation sensors, replicating a real smart apartment. Working with volunteers, we generated data in the actual smart apartment through various activity scenarios. Subsequently, we replicated the volunteers' activities in the digital twin of the apartment using an avatar. Through comprehensive analysis and metric evaluation, we observed the similarity between the time series generated by the simulator and those generated by the volunteers.

The synthetic sensor logs allowed us to train a HAR algorithm, which we then validated on the real data collected by the volunteers. Our experiments demonstrated that the algorithm achieved robust activity recognition performance on the real data, with an average F1 score of approximately 80%. It is important to note that these experiments were conducted on a relatively small dataset, and further validation with larger amounts of data is necessary. Additionally, future work should involve continuous data generation from residents living permanently in the target environment to further validate the algorithm's performance.

Nevertheless, our findings indicate that utilizing a digital twin to generate synthetic data resembling the target environment shows promise for training HAR algorithms. In our future work, we plan to expand the experiment by generating more synthetic data from additional volunteers' activities. We also aim to extend the evaluation to a larger number of houses to validate the generalizability of our approach. Furthermore, we will explore the possibility of training the algorithm from scratch in a house without reproducing the labeled activities of the final resident, solely utilizing the activity scripts provided by our volunteers.

By leveraging the concept of digital twins and generating realistic synthetic data, we can mitigate the challenges posed by limited datasets and enhance the applicability of HAR algorithms in real-world environments.

1. Dang, L.M.; Min, K.; Wang, H.; Piran, M.J.; Lee, C.H.; Moon, H. Sensor-based and vision-based human activity recognition: A comprehensive survey. *Pattern Recognition* **2020**, *108*, 107561.
2. Bouchabou, D.; Nguyen, S.M.; Lohr, C.; LeDuc, B.; Kanellos, I.; others. A Survey of Human Activity Recognition in Smart Homes Based on IoT Sensors Algorithms: Taxonomies, Challenges, and Opportunities with Deep Learning. *Sensors* **2021**, *21*, 6037.
3. Tan, T.H.; Gochoo, M.; Huang, S.C.; Liu, Y.H.; Liu, S.H.; Huang, Y.F. Multi-resident activity recognition in a smart home using RGB activity image and DCNN. *IEEE Sensors Journal* **2018**, *18*, 9718–9727.
4. Bouchabou, D.; Nguyen, S.M.; Lohr, C.; Kanellos, I.; Leduc, B. Fully Convolutional Network Bootstrapped by Word Encoding and Embedding for Activity Recognition in Smart Homes. IJCAI 2020 Workshop on Deep Learning for Human Activity Recognition; , 2021.
5. Liciotti, D.; Bernardini, M.; Romeo, L.; Frontoni, E. A Sequential Deep Learning Application for Recognising Human Activities in Smart Homes. *Neurocomputing* **2019**, *396*. doi:10.1016/j.neucom.2018.10.104.
6. Hussain, Z.; Sheng, Q.; Zhang, W.E. *Different Approaches for Human Activity Recognition: A Survey*; 2019.
7. Tapia, E.M.; Intille, S.S.; Larson, K. Activity recognition in the home using simple and ubiquitous sensors. International conference on pervasive computing. Springer, 2004, pp. 158–175.
8. van Kasteren, T.L.; Englebienne, G.; Kröse, B.J. Human activity recognition from wireless sensor network data: Benchmark and software. In *Activity recognition in pervasive intelligent environments*; Springer, 2011; pp. 165–186.
9. Alemdar, H.; Ertan, H.; Incel, O.D.; Ersoy, C. ARAS human activity datasets in multiple homes with multiple residents. 2013 7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops. IEEE, 2013, pp. 232–235.

10. Cumin, J.; Lefebvre, G.; Ramparany, F.; Crowley, J.L. A dataset of routine daily activities in an instrumented home. International Conference on Ubiquitous Computing and Ambient Intelligence. Springer, 2017, pp. 413–425.
11. Grieves, M. Digital twin: manufacturing excellence through virtual factory replication. *White paper* **2014**, *1*, 1–7.
12. Grieves, M.; Vickers, J. Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. In *Transdisciplinary perspectives on complex systems*; Springer, 2017; pp. 85–113.
13. Barricelli, B.R.; Casiraghi, E.; Fogli, D. A survey on digital twin: definitions, characteristics, applications, and design implications. *IEEE access* **2019**, *7*, 167653–167671.
14. Ngah Nasaruddin, A.; Ito, T.; Tee, B.T. Digital Twin Approach to Building Information Management. *The Proceedings of Manufacturing Systems Division Conference* **2018**, *2018*, 304. doi:10.1299/jsmemsd.2018.304.
15. Khajavi, S.; Hossein Motlagh, N.; Jaribion, A.; Werner, L.; Holmström, J. Digital Twin: Vision, Benefits, Boundaries, and Creation for Buildings. *IEEE Access* **2019**, *7*, 147406–147419. doi:10.1109/ACCESS.2019.2946515.
48. Puig, X.; Ra, K.; Boben, M.; Li, J.; Wang, T.; Fidler, S.; Torralba, A. Virtualhome: Simulating Household Activities via Programs. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8494–8502.
16. SEDKY, M.; HOWARD, C.; Alshammari, T.; Alshammari, N. Evaluating machine learning techniques for activity classification in smart home environments. *International Journal of Information Systems and Computer Sciences* **2018**, *12*, 48–54.
17. Gochoo, M.; Tan, T.H.; Liu, S.H.; Jean, F.R.; Alnajjar, F.S.; Huang, S.C. Unobtrusive activity recognition of elderly people living alone using anonymous binary sensors and DCNN. *IEEE journal of biomedical and health informatics* **2018**, *23*, 693–702.
18. Singh, D.; Merdivan, E.; Hanke, S.; Kropf, J.; Geist, M.; Holzinger, A. Convolutional and recurrent neural networks for activity recognition in smart environment. In *Towards integrative machine learning and knowledge extraction*; Springer, 2017; pp. 194–205.
19. Singh, D.; Merdivan, E.; Psychoula, I.; Kropf, J.; Hanke, S.; Geist, M.; Holzinger, A. Human activity recognition using recurrent neural networks. International Cross-Domain Conference for Machine Learning and Knowledge Extraction. Springer, 2017, pp. 267–274.
20. Cook, D.J.; Crandall, A.S.; Thomas, B.L.; Krishnan, N.C. CASAS: A Smart Home in a Box. *Computer* **2013**, *46*, 62–69. Conference Name: Computer, doi:10.1109/MC.2012.328.
21. De-La-Hoz-Franco, E.; Ariza-Colpas, P.; Quero, J.M.; Espinilla, M. Sensor-based datasets for human activity recognition–a systematic review of literature. *IEEE Access* **2018**, *6*, 59192–59210.
22. Kolve, E.; Mottaghi, R.; Han, W.; VanderBilt, E.; Weihs, L.; Herrasti, A.; Gordon, D.; Zhu, Y.; Gupta, A.; Farhadi, A. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv:1712.05474 [cs]* **2019**.
23. Xia, F.; Zamir, A.R.; He, Z.; Sax, A.; Malik, J.; Savarese, S. Gibson Env: Real-World Perception for Embodied Agents. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition; IEEE: Salt Lake City, UT, 2018; pp. 9068–9079. doi:10.1109/CVPR.2018.00945.
24. Roitberg, A.; Schneider, D.; Djamal, A.; Seibold, C.; Reiß, S.; Stiefelhagen, R. Let's Play for Action: Recognizing Activities of Daily Living by Learning from Life Simulation Video Games. *arXiv:2107.05617 [cs]* **2021**. arXiv: 2107.05617.
25. Savva, M.; Kadian, A.; Maksymets, O.; Zhao, Y.; Wijmans, E.; Jain, B.; Straub, J.; Liu, J.; Koltun, V.; Malik, J.; Parikh, D.; Batra, D. Habitat: A Platform for Embodied AI Research. *arXiv:1904.01201 [cs]* **2019**. arXiv: 1904.01201.
26. Alshammari, N.; Alshammari, T.; Sedky, M.; Champion, J.; Bauer, C. OpenSHS: Open smart home simulator. *Sensors* **2017**, *17*, 1003. doi:10.3390/s17051003.
27. Ho, B.; Vogts, D.; Wesson, J. A smart home simulation tool to support the recognition of activities of daily living. In *Proceedings of the South African Institute of Computer Scientists and Information Technologists 2019*; 2019; pp. 1–10.
28. Lee, J.W.; Cho, S.; Liu, S.; Cho, K.; Helal, S. Persim 3D: Context-Driven Simulation and Modeling of Human Activities in Smart Spaces. *IEEE Transactions on Automation Science and Engineering* **2015**, *12*, 1243–1256.
29. Synnott, J.; Chen, L.; Nugent, C.; Moore, G. IE Sim – A Flexible Tool for the Simulation of Data Generated within Intelligent Environments. Ambient Intelligence; Paternò, F.; de Ruyter, B.; Markopoulos, P.; Santoro, C.; van Loenen, E.; Luyten, K., Eds.; Springer: Berlin, Heidelberg, 2012; Lecture Notes in Computer Science, pp. 373–378.
30. Bouchard, K.; Ajroud, A.; Bouchard, B.; Bouzouane, A. SIMACT: a 3D Open Source Smart Home Simulator for Activity Recognition with Open Database and Visual Editor. *International Journal of Hybrid Information Technology (IJHIT)*, **2012**, *Volume 5*, 13–32.
31. Park, B.; Min, H.; Bang, G.; Ko, I. The User Activity Reasoning Model in a Virtual Living Space Simulator. *International Journal of Software Engineering and Its Applications* **2015**, *9*, 53–62.
32. Francillette, Y.; Boucher, E.; Bouzouane, A.; Gaboury, S. The Virtual Environment for Rapid Prototyping of the Intelligent Environment. *Sensors* **2017**, *17*, 2562.
33. Buchmayr, M.; Kurschl, W.; Küng, J. A simulator for generating and visualizing sensor data for ambient intelligence environments. *Procedia Computer Science* **2011**, *5*, 90–97.
34. Armac, I.; Retkowitz, D. Simulation of smart environments. IEEE International Conference on Pervasive Services. IEEE, 2007, pp. 257–266.
35. VirtualHome. http://www.virtual-home.org/, accessed on 2021-01-21.
36. Golestan, S.; Stroulia, E.; Nikolaidis, I. Smart Indoor Space Simulation Methodologies: A Review. *IEEE Sensors Journal* **2022**.
38. Bruneau, J.; Consel, C.; OMalley, M.; Taha, W.; Hannourah, W.M. Virtual testing for smart buildings. 2012 Eighth International Conference on Intelligent Environments. IEEE, 2012, pp. 282–289.

39. Savva, M.; Chang, A.X.; Dosovitskiy, A.; Funkhouser, T.; Koltun, V. MINOS: Multimodal indoor simulator for navigation in complex environments. *arXiv preprint arXiv:1712.03931* **2017**.
40. Roberts, M.; Ramapuram, J.; Ranjan, A.; Kumar, A.; Bautista, M.A.; Paczan, N.; Webb, R.; Susskind, J.M. Hypersim: A Photorealistic Synthetic Dataset for Holistic Indoor Scene Understanding. 2021.
41. Srivastava, S.; Li, C.; Lingelbach, M.; Martín-Martín, R.; Xia, F.; Vainio, K.; Lian, Z.; Gokmen, C.; Buch, S.; Liu, C.K.; Savarese, S.; Gweon, H.; Wu, J.; Fei-Fei, L. BEHAVIOR: Benchmark for Everyday Household Activities in Virtual, Interactive, and Ecological Environments. *CoRR* **2021**, *abs/2108.03332*, [2108.03332].
42. Shridhar, M.; Thomason, J.; Gordon, D.; Bisk, Y.; Han, W.; Mottaghi, R.; Zettlemoyer, L.; Fox, D. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 10740–10749.
43. Puig, X.; Shu, T.; Li, S.; Wang, Z.; Liao, Y.H.; Tenenbaum, J.B.; Fidler, S.; Torralba, A. Watch-And-Help: A Challenge for Social Perception and Human-{AI} Collaboration. International Conference on Learning Representations, 2021.
44. Cao, Z.; Gao, H.; Mangalam, K.; Cai, Q.; Vo, M.; Malik, J. Long-term human motion prediction with scene context. In *ECCV*; 2020.
45. Varol, G.; Romero, J.; Martin, X.; Mahmood, N.; Black, M.J.; Laptev, I.; Schmid, C. Learning from Synthetic Humans. CVPR, 2017.
46. Synnott, J.; Nugent, C.; Jeffers, P. Simulation of smart home activity datasets. *Sensors* **2015**, *15*, 14162–14179.
47. Kamara-Esteban, O.; Azkune, G.; Pijoan, A.; Borges, C.E.; Alonso-Vicario, A.; López-de Ipiña, D. MASSHA: an agent-based approach for human activity simulation in intelligent environments. *Pervasive and Mobile Computing* **2017**, *40*, 279–300.
. Helal, S.; Mann, W.; El-Zabadani, H.; King, J.; Kaddoura, Y.; Jansen, E. The gator tech smart house: A programmable pervasive space. *Computer* **2005**, *38*, 50–60.
49. Puig, X. VirtualHome source code. https://github.com/xavierpuigf/virtualhome_unity, accessed on 2021-01-25.
50. Liao, Y.; Puig, X.; Boben, M.; Torralba, A.; Fidler, S. Synthesizing Environment-Aware Activities via Activity Sketches. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 6284–6292. ISSN: 2575-7075, doi:10.1109/CVPR.2019.00645.
51. VirtualHome source code and API. https://github.com/xavierpuigf/virtualhome, accessed on 2021-01-25.
52. Sweet Home 3D - Draw floor plans and arrange furniture freely. https://www.sweethome3d.com/, accessed on 2022-09-10.
53. Experiment'Haal, le Living lab santé autonomie (LLSA). http://www.imt-atlantique.fr/fr/recherche-et-innovation/plateformes-de-recherche/experiment-haal, accessed on 2021-01-21.
54. Lohr, C.; Kerdreux, J. Improvements of the xAAL home automation system. *Future internet* **2020**, *12*, 104.
55. Future-Shape. SensFloor - The Floor Becomes a Touch Screen. https://future-shape.com/en/system/, accessed on 2021-12-06.
56. Katz, S. Assessing self-maintenance: activities of daily living, mobility, and instrumental activities of daily living. *Journal of the American Geriatrics Society* **1983**, *31*, 721–727.
57. Cross-correlation. https://en.wikipedia.org/w/index.php?title=Cross-correlation&oldid=1031522391, accessed on 2021-08-17. Page Version ID: 1031522391.
59. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural computation* **1997**, *9*, 1735–1780.
60. Sammut, C.; Webb, G.I., Eds., Leave-One-Out Cross-Validation. In *Encyclopedia of Machine Learning*; Sammut, C.; Webb, G.I., Eds.; Springer US: Boston, MA, 2010; pp. 600–601. doi:10.1007/978-0-387-30164-8_469.