Politecnico di Torino

Politecnico di Torino

Master's degree in Biomedical Engineering
A.y. 2023/2024
December 2024

Improvement of safety systems for human-robot collaboration through real-time detection of abrupt movements with inertial sensors and artificial intelligence

Supervisors:

Prof. Laura Gastaldi

Prof. Stefano Pastorelli

PhD Elisa Digo

Eng. Michele Polito

Candidate:

Greta Di Vincenzo

317504

ABSTRACT

Collaborative robotics plays a significant role in the industrial sector, especially following the advent of the 4th and 5th industrial revolutions. In this context, humans and robots share a workspace where they collaborate and exchange information, enhancing each other's strengths. Robots perform repetitive tasks with precision and speed, while humans provide essential decision-making capabilities, ensuring an effective production process. However, guaranteeing the safety of human-robot interaction is crucial, a concept known as "safety collaboration". To achieve this, robots must recognize human activities, such as detecting abrupt movements, and respond accordingly. The recognition needs to be rapid to make the safety system activating as quickly as possible to prevent collisions.

The objective of this study was to detect abrupt movements in real time using data from magneto-inertial measurement units (MIMUs) and an artificial intelligence network. A Long Short-Term Memory neural network was employed for this purpose, trained with a dataset of 61 subjects who performed a pick-and-place task involving impulsive movements. The data, acquired using MIMUs, consisted of accelerations and angular velocities of the forearm during the movements. Tests were conducted in three different spatial configurations relative to the experimental setup. First, the network was tested on the data from the 61 subjects, which were segmented into fixed overlapping sliding windows. The window length was set to 0.5 seconds, with various overlap percentages (50%, 75%, 90%, 95%, 99%) evaluated to estimate the network's performance and move closer to real-time conditions. Specifically, the network's ability to detect abrupt and standard movements, as well as the recognition time, were evaluated. The results demonstrated that a real-time recognition is achievable.

Subsequently, the same tests used to create the training dataset was repeated with the same protocol and with five new subjects. The goal was to achieve a real-time recognition of the movement. Sensor data were streamed in real-time directly into a Python script, where they were immediately stored, pre-processed, and then analysed by the network to identify the type of movement. Finally, the network's performance and the time required for data streaming, pre-processing, and recognition were evaluated. Results showed that the network could effectively distinguish between abrupt and standard

movements in conditions approaching real-time. For a single movement, data stream from sensors to the Python script took around 3 seconds, pre-processing took a few milliseconds (about 9 ms), and the network's recognition time was around a few hundred milliseconds (approximately 300 ms).

The findings of this study demonstrated the effectiveness of using inertial sensors together with artificial intelligence networks for a real-time identification of abrupt movements, aimed at enhancing safety systems for human-robot interactions in industrial settings.

Table of Contents

1. INTRODUCTION

1.1 Humans and robots in the industrial field

1.1.1 Industry 4.0 and Industry 5.0

The term “collaborative robotics”, or “*cobotics”*, refers to humans and intelligent machines working together dynamically to perform tasks in a safe and effective way. Cobotics has played a crucial role in the industrial sector, driving both the 4th and 5th industrial revolutions, known respectively as Industry 4.0 and Industry 5.0.

Industry 4.0 refers to the advanced integration of machines and processes within the industrial sector, enabling intelligent control and automation of industrial operations. This revolution is considered technology driven and it has the objective to achieve higher productivity and efficiency (Xu et al., 2021). Conversely, Industry 5.0 provides a different focus and point of view. It is considered a value-driven revolution, integrating social and environmental priorities into technological innovation (Xu et al., 2021). Industry 5.0 complements and goes beyond Industry 4.0, exploiting these new technologies to improve the worker's quality of life, sustainability, and social welfare.

Industry 5.0 relies on three core values (Xu et al., 2021):

1. *Human-centricity*: there is a shift from technology-driven processes to a human-centric approach, where a safe and inclusive work environment is prioritized. This approach emphasizes the physical and mental health of workers, as well as their fundamental rights.
2. *Sustainability*: the industry must respect planetary boundaries by reducing waste and minimizing environmental impact. To achieve this, it needs to develop circular processes that reuse, repurpose, and recycle natural resources.
3. *Resilience*: the future industry needs to be able to rapidly address (geo)political changes and natural emergencies.

The core goal is to create a shared space where humans and robots can exchange information and collaborate, enhancing each other's strengths. Machines can assist workers with tasks that require precision, strength, or speed, while humans bring creativity, decision-making, and problem-solving skills (Zafar et al., 2024). By combining

their abilities, we can develop adaptive production systems that can rapidly adjust to changes or faults (Zafar et al., 2024). Therefore, the aim is to work together rather than replaces human labour.

1.1.2 Human-robot collaboration: from caged robots to cobots

The first industrial robots were introduced during the Third Industrial Revolution, also known as Industry 3.0, where electronics and technology began playing a significant role in production processes. These robots, confined to cages, were pre-programmed and capable of performing specific tasks. As technology advanced, there has been a gradual transition from caged robots to collaborative robots, or cobots.

The main difference between robots and cobots lies in the concept of collaboration. While both can perform similar tasks, cobots work alongside human operators, whereas traditional industrial robots typically replace human workers (Borboni et al., 2023). For effective task performance, humans need to interact and work closely with cobots. As a result, sensors, software, and safety devices are incorporated to ensure safe and efficient collaboration, removing the traditional barriers between industrial robots and human workers. Moreover, this reduction in barriers, due to improved robot safety, has led to increased levels of collaboration, as illustrated in *Figure 1.1*.

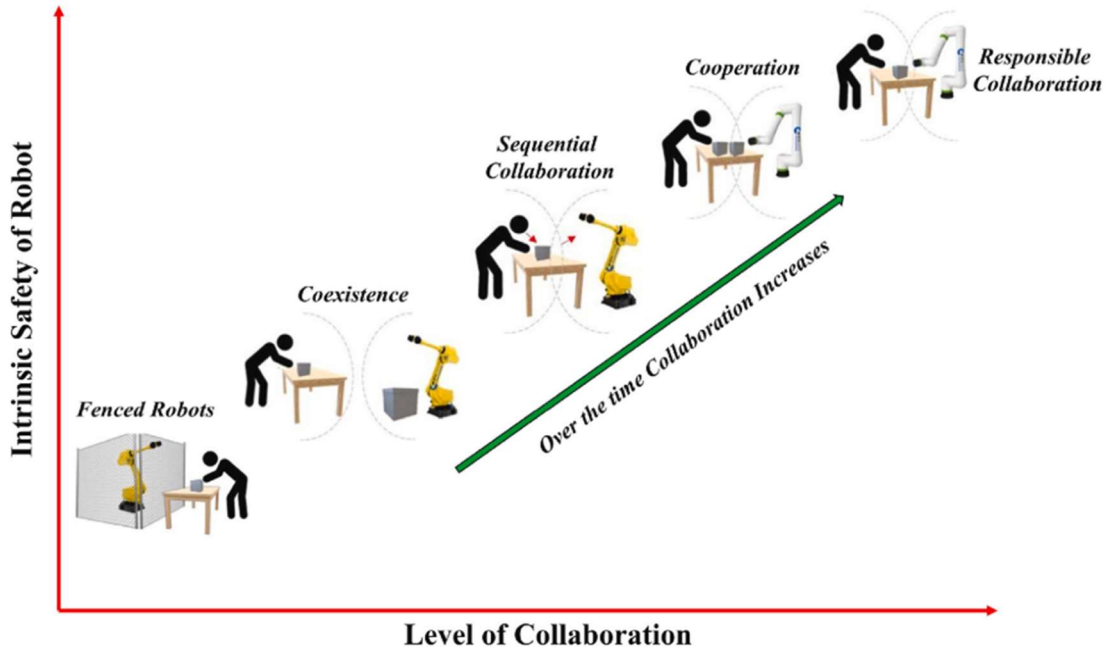


Figure 1.1. Increasing levels of human–robot collaboration as safety improves. (Zafar et al., 2024)

According to (Zafar et al., 2024), the main stages of this transition are as follows:

- Caged robots: the first industrial robots operated within physical barriers, such as secured cages or fences, to prevent direct interaction or contact between robots and human workers. The primary aim was to ensure safety in the workplace since robots had limited functionalities and lacked the advanced safety features required for safe collaboration.
- Collision Avoidance: as technology progressed, robots began incorporating sensors and cameras, allowing them to detect nearby humans. By sensing their presence, robots could adjust their actions to prevent potential collisions or accidents. This breakthrough significantly improved safety standards in collaborative work environments.
- Human-Robot Interaction (HRI): natural language processing and speech recognition technologies transformed the way robots interacted with humans. These innovations enabled robots to understand and respond to verbal instructions, reducing the communication gap between humans and robots. This marked the beginning of a new era in which robots became more interactive and responsive, allowing for smoother collaboration.
- Human-Robot Collaboration (HRC): at this stage, robots and humans work together on tasks, requiring robots to not only understand human intentions but also collaborate efficiently while ensuring safety. HRC marks a significant shift, moving toward a model where humans and robots function as complementary partners.
- Physical HRC (pHRC): this phase represents a deeper level of integration, where robots actively interact with humans through physical contact. This can involve activities like exchanging tools, handing over objects, or collaboratively manipulating items. Successful pHRC requires advanced control and sensing systems to ensure safe and efficient cooperation.
- Human-Robot Teaming (HRT): in this advanced stage, robots are no longer tools but become fully integrated as equal members of human teams. Achieving this requires sophisticated AI and machine learning that allow robots to learn from and adapt to human behaviour, preferences, and decision-making. Robots thus

become dynamic and adaptive team members that work together toward a shared goal, marking a fundamental shift in how humans and robots interact.

1.1.3 Advantages of collaborative robots in industry

The use of collaborative robots in industry brings socio-economic benefits. In this context, robots do not entirely replace human work but instead support and improve it. Robots perform automatic and repetitive tasks, ensuring accuracy, precision, speed, and strength. Additionally, they can handle heavy payloads and perform dangerous tasks, such as manipulating toxic or hot objects (Vysocky & Novak, 2016). However, they are not capable of adapting to changes or making decisions beyond the tasks they have been programmed for. For this reason, human presence is essential due to their decision-making and problem-solving skills. *Figure 1.2* summarizes the collaboration between human and robot, highlighting their respective strengths.

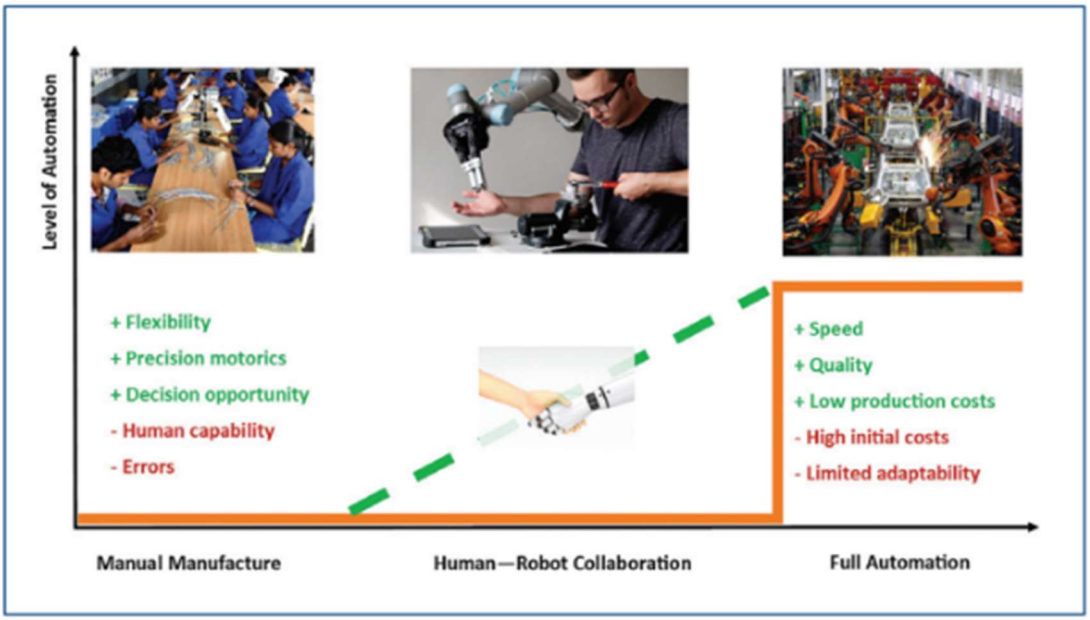


Figure 1.2. Graphical representation of Human–robot collaboration, highlighting their capabilities (Vysocky & Novak, 2016)

This collaborative workspace provides several crucial advantages in industry (Vysocky & Novak, 2016; Zafar et al., 2024):

- Manufacturing systems can be more complex as new technologies can perform a wider range of tasks.

- Robots' repeatable positioning accuracy and ability to operate continuously result in improved quality and a reduced need for post-processing and quality control, enabling lower-cost production.
- Robots can speed up some processes and adapt to specific conditions, which can lead to an increased production.
- Stressful, monotonous, and tedious labour, which can eventually lead to occupational illness, is reduced, easing the burden on workers.
- A reduction in occupational injuries is also achieved by improving workplace ergonomics and effectively managing the workload.
- The integration of collaborative robotics and safety-focused technology ensure a secure working environment, decreasing the risk of injury.

1.1.4 Safety standards and levels of collaboration

When discussing collaboration, various types of interactions between humans and cobots must be considered. Therefore, it is essential to define all levels of collaboration to address different scenarios from both safety and human factors perspectives.

The introduction of industrial robots in human-robot collaborative applications offers several advantages, as previously discussed, but it also presents new challenges regarding safety. For this reason, the International Organization for Standardization (ISO) published the specification ISO/TS 15066, which provides objective parameters for assessing safety in collaborative applications (Rosenstrauch & Kruger, 2017). This standard first addresses general hazard identification and risk assessment, serving as basic guidelines for identifying, evaluating, and reducing risks. The main steps are summarized in *Figure 1.3*.

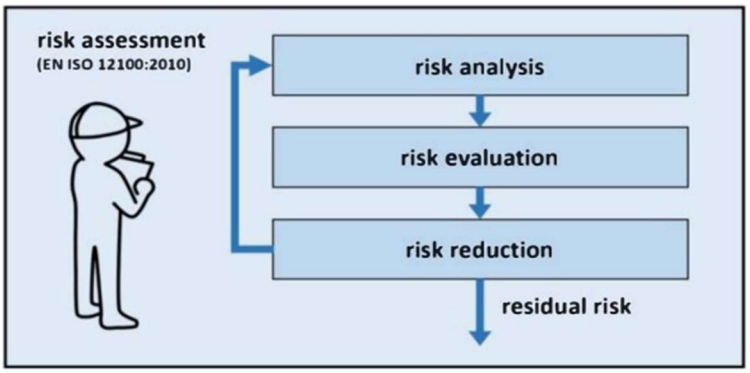


Figure 1.3. ISO/TS 12100:2010 basic procedure of risk assessment (Rosenstrauch & Kruger, 2017)

The first step is the risk analysis, which involves identifying all potential risks or hazards, including mechanical, electrical, thermal, and others. The second step is the risk evaluation, which is the combination of the probability of occurrence and the extent of potential damage. These two steps are followed by risk reduction, during which protective measures and safeguards are implemented. As shown in *Figure 3*, this process is iterative, continuing until the residual risk is minimized (Rosenstrauch & Kruger, 2017).

ISO/TS 15066 then presents the requirements for collaborative robot system applications, distinguishing between four different operating modes (Rosenstrauch & Kruger, 2017):

- **Safety-rated monitored stop** (*Figure 1.4a)*. The robot is allowed to move only when the operator is outside the collaborative workspace. As soon as he/she enters the area to interact, the robot halts, resuming operation only when the operator leaves the workspace.
- **Speed and separation monitoring** (*Figure 1.4b*). The robot's speed adjusts dynamically based on the distance between the operator and the robot. As the operator moves closer, the robot slows down, and if the distance falls below a predefined safety limit, the robot stops to prevent any risk of collision.
- **Hand guiding** (*Figure 1.4c*). Direct contact between the operator and the robot is permitted. In this mode, the operator can guide the robot's movements within the collaborative space using a hand-guiding device or a force-torque sensor located at the robot's tool centre point.
- **Power and force limiting** (*Figure 1.4d*). In a fully shared collaborative workspace, unintentional and unpredictable contact between humans and the robot is possible. Therefore, the robot's power and force are limited to ensure safety. Thresholds for pressure and force are set based on maximum permissible levels for different body parts, distinguishing between quasi-static and transient contact. This ensure that any contact remains within safe biomechanical limits.

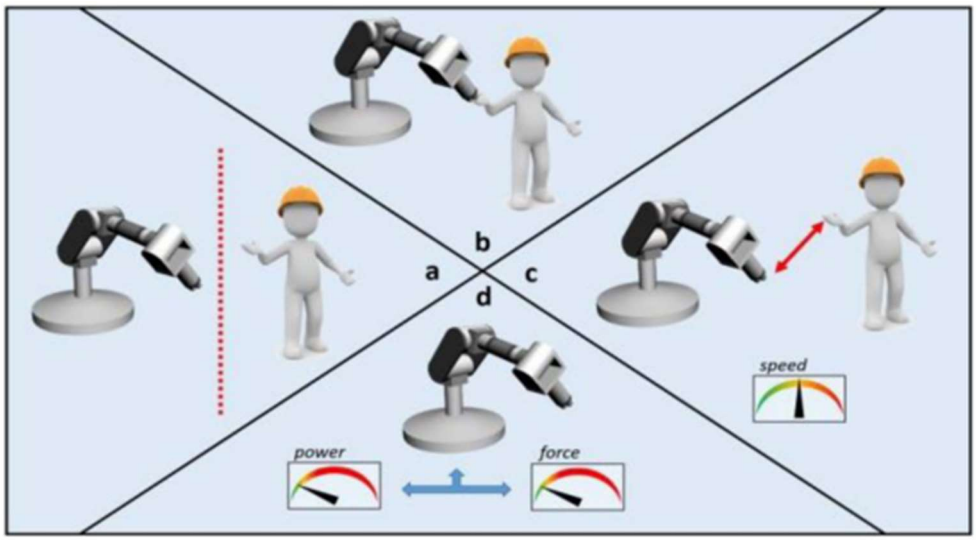


Figure 1.4. ISO/TS 15066:2016 collaborative operation modes (Vysocky & Novak, 2016)

These four modes describe collaboration from a technical perspective, offering specific configurations for the robot system to ensure safety after risk assessment. However, from the human worker's perspective, the assumption is that collaborative work is inherently safe, regardless of the implemented safety method. To address this, Aaltonen and colleagues proposed a new classification based on factors such as workspace sharing, the type of joint effort, and the physical contact involved (Aaltonen et al., 2018). The aim is to provide a comprehensive analysis of collaboration, ensuring compliance with safety standards while also creating a positive experience for the human worker. Four levels of collaboration are proposed:

- No coexistence: Physical separation, referring to traditional fenced robot cells.
- Coexistence: Humans and robots share the physical workspace (partially or completely), but they do not work towards a shared goal.
- Cooperation: Humans and robots work toward a shared goal in a partially or completely shared workspace.
- Collaboration: Humans and robots work simultaneously on the same object.

These levels are structured to represent progressively deeper forms of joint effort. For instance, while coexistence and cooperation might appear similar from a safety standpoint, they differ from the human worker's perspective. If the human's task depends on the robot's activity, the interaction reaches a higher level, and mutual awareness of the situation becomes crucial for effective collaboration (Aaltonen et al., 2018).

In the new industrial paradigm, humans are central to production processes, meaning that collaboration with robots must not only be effective but, most importantly, safe. The key concept is the "safety collaboration", highlighting the need for research to focus on ensuring a safe human-robot interaction.

1.2 The role of Artificial Intelligence in collaborative robotics

An essential component of Industry 5.0 is artificial intelligence (AI), making manufacturing processes smarter and more efficient. According to the work of Borboni and colleagues, many recent articles have highlighted the growing influence of AI in the development and functionalities of cobots (Borboni et al., 2023). Furthermore, the incorporation of AI into cobots has led to improved performance, suggesting that AI enhances their capability and efficiency in collaborative tasks.

Recent advancements in AI have significantly improved Human–Robot Collaboration through the development of a cognitive model. These models collect information from the environment and the human operator, process it, and convert it into data that enables the robot to adapt its behaviour (Zafar et al., 2024). This capability could reduce risks and promotes a safer human-robot collaboration.

One of the most widely used AI approaches is Machine Learning (ML), which refers to a machine's ability to analyse data, learn from it, and improve its performance over time. A subfield of ML is Deep Learning (DL), which is a neural network composed of multiple layers of interconnected neurons. The term "deep" refers to the abundance of layers, or "depth", that are hierarchically organised to mimic the human cognitive functions (Borboni et al., 2023).

In the work of Ordoñez and Roggen, differences between various deep learning architectures are outlined (Ordóñez & Roggen, 2016a). One effective model for classification tasks is the feedforward neural network, or multi-layer perceptron (MLP). This model consists of multiple neurons organized in layers and connected by weighted links. However, MLPs assume that all inputs and outputs are independent, meaning they don't capture relationships between sequential data points. To model time-dependent data, such as sensor signals, temporal information must be incorporated. Recurrent Neural Networks (RNNs) are specifically designed to address this limitation. Each unit in a RNN

has a recurrent connection, where the output of a neuron is fed back to itself with a weight and a unit time delay. This feedback loop gives the neuron a memory (hidden value) of past activations, allowing it to learn temporal patterns in sequential data. However, this memory mechanism can make learning difficult when applied to real-world sequences. To address this issue, Long Short-Term Memory (LSTMs) networks extend RNNs by using memory cells instead of simple recurrent units. These memory cells store and manage data more effectively, making it easier to learn patterns over long-time scales. At each time step, LSTMs update their memory using a gating mechanism. There are three different gates that control operations on the cell memory: the input gate controls when new information is written, the output gate controls when stored information is read, and the forget gate decides when to reset the memory. This mechanism allows LSTMs to better manage temporal patterns over long sequences.

Figure 1.5 provides an overview of the units that define the structure of these neural networks.

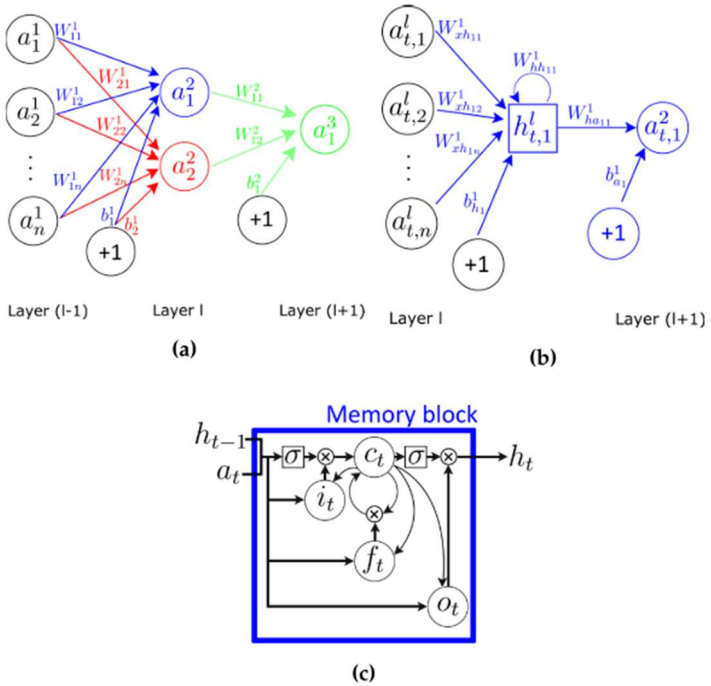


Figure 1.5. Different types of units in neural networks. (a) MLP with three dense layers; (b) RNN with two dense layers; (c) LSTM memory cell, where the internal memory can be updated, erased, or read. (Ordóñez & Roggen, 2016b)

1.3 Human activity recognition (HAR)

In a collaborative scenario, robots are designed to work alongside human workers. As a result, industrial robots are required to recognize human movement and position in order to dynamically adjust their pre-programmed task, both for safety reasons and to enable effective teamwork and seamless communication. Consequently, Human Activity Recognition (HAR) represents an important area of study in the field of human-robot interaction.

HAR is based on the hypothesis that specific body movements produce distinct patterns in sensor signals, which can be detected and classified using machine learning techniques. However, HAR presents several challenges in real-world settings. First, motor movements associated with specific activities can vary significantly (Ordóñez & Roggen, 2016a). Second, determining the appropriate experimental setup for accurate data collection can be difficult, as it is necessary to ensure that the collected data is representative of real-world scenarios (Imanzadeh et al., 2024). Moreover, the collected datasets are used to train a neural network, which plays a fundamental role in HAR. The choice of machine learning algorithms, along with an appropriate dataset, is crucial for achieving accurate recognition results.

1.3.1 HAR applications

In literature, various examples of collected datasets and neural networks can be found, depending on the application field, whether clinical or industrial, and the specific objectives. For instance, Buerkle and colleagues investigated the use of electroencephalogram (EEG) signals to detect upper-limb movement intention (Buerkle et al., 2021). The aim is to predict the operator's movements to prevent collisions with robots, ensuring safe collaboration. A Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) was trained to detect and classify arm movement intentions. The results suggested that this approach could be employed to dynamically adjust robot's speed and torque, thereby minimizing the risk of collisions.

Similarly, in the industrial field, in the work of Liu and colleagues three different datasets, human body posture, voice commands, and hand motion data, with the aim of developing a robot control interface, were collected (Liu et al., 2018). The first two datasets were used

to train a Convolutional Neural Network (CNN), while the hand motion data was used to train a Multilayer Perceptron (MLP). The workflow followed in this study is outlined in *Figure 1.6*. The results demonstrate the potential of deep learning algorithms for classification and recognition. However, for hand motion recognition, an LSTM would likely be more suitable, as it is expected to outperform the MLP model used.

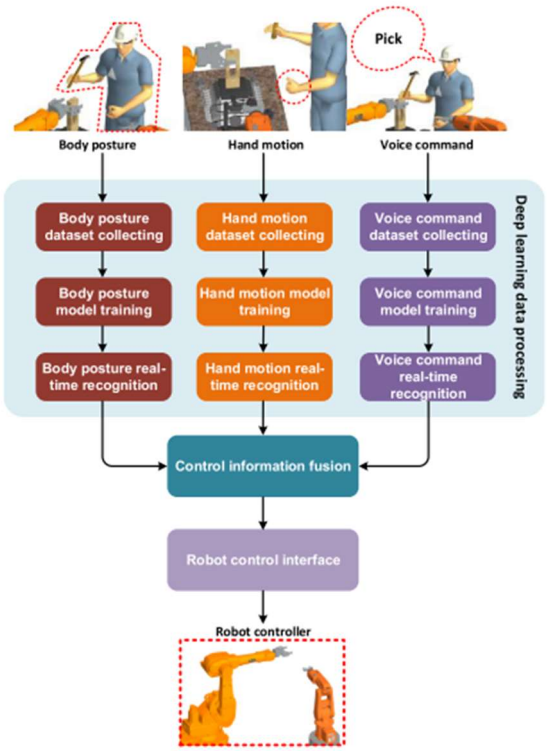


Figure 1.6. Workflow of the study, from data collection to the development of a deep learning-based robot control interface for human-robot collaboration. (Liu et al., 2018)

In addition to these types of signals, images can also be used to extract specific information. For example, Amaral and colleagues extracted hand landmarks to identify objects being grasped or manipulated (Amaral et al., 2023). A multi-class classifier was used to predict the object based on the hand key points. This study focuses on evaluating the classifier's generalization ability for real-world application. In this context, active data collection plays a crucial role.

Another way to collect data is through Inertial Measurement Units (IMUs), which are sensors that include an accelerometer, gyroscope, and, in the case of magneto-inertial measurement units (MIMUs), a magnetometer. These sensors can be worn on different parts of the body and measure linear acceleration (via the accelerometer) and angular velocity (via the gyroscope). IMUs offer several advantages: they are low-cost, minimally

invasive, easy to wear, and they can collect data outside of a lab setting (Digo, Polito, Pastorelli, et al., 2024; Xiang et al., 2024). These features make them suitable for biomechanical research in both industrial (Ordóñez & Roggen, 2016a) and clinical (Xiang et al., 2024) fields.

Ordoñez and Roggen proposed a deep neural network model called DeepConvLSTM for recognizing modes of locomotion, postures, and different right arm gestures using IMU sensors (Ordóñez & Roggen, 2016a). This model combines convolutional layers and recurrent layers. The convolutional layers act as feature extractors from the sensor data, while the recurrent layers take these features and learn how they evolve over time, capturing temporal patterns in the data. The study's results demonstrated that the LSTM-based model can distinguish between activities that are similar but differ in the sequence of sensor samples (e.g. Open/Close Door). Additionally, it works even when gestures extend beyond the observation window. These findings highlight that the LSTM-based model approach is better suited for handling sequences and time-dependent data, as it learns how features change over time, compared to convolutional models alone.

An LSTM-based model, combined with time-series data from IMUs, is also applied in the clinical field. For example, Xiang and colleagues implemented an LSTM-MLP model to predict ankle joint biomechanics (Xiang et al., 2024). The model can identify and learn gait characteristics and patterns from acceleration and angular velocity signals, enabling accurate prediction of ankle joint angles, torques, and contact forces.

These studies demonstrate that IMU sensors, combined with LSTM neural networks, provide a cost-effective and versatile tool for identifying human activity, representing a reliable solution for developing safety and control systems in collaborative robotics.

1.4 Abrupt movements and real-time recognition systems

Typically, the tasks performed by an operator are repetitive and characterized by controlled dynamics and kinematics. However, external disturbances or environmental factors can cause abrupt and unpredictable gestures. These sudden movements can lead to improper interactions with the robot, potentially creating unsafe conditions for the human operator (Digo, Polito, Pastorelli, et al., 2024; Polito et al., 2023a).

Accurate data collection that represents real-world conditions and the selection of the appropriate neural network are essential for effectively identifying human activity. Abrupt movements are characterized by high variability and uncertain patterns, making inertial sensors particularly suitable for detecting these variations. In fact, these sensors are easy to wear, do not restrict movement, and can capture accelerations and angular velocities at high frequency, allowing for the detection of significant motion changes. Moreover, they generate a time series of data. Given the importance of temporal dependencies, the LSTM network is the most appropriate choice, as it effectively captures patterns in long sequences.

To prevent collisions between humans and robots, it is crucial to identify these movements in real-time. However, one of the main limitations of real-time recognition systems is reaction time (Buerkle et al., 2021). Safety systems must be highly responsive, activating as quickly as possible. According to (Vysocky & Novak, 2016), there are four possible reactions based on the system's safety level:

1. Alert: When a potential hazard or risk of collision is detected, an audible alarm and visual warning are activated to signal imminent danger.
2. Stop: The robot automatically halts to prevent any collision.
3. Compliance Control: The robot adjusts its position in response to force or physical contact.
4. Trajectory Adjustment: The robot senses an obstacle and alters its trajectory to completely avoid a collision.

A primary goal of research should be to reduce the reaction time of these systems to enhance their efficiency and safety. The analysis of response times starts with the network's ability to recognize movement, followed by the activation of the safety system. Therefore, it is crucial to ensure that this classification occurs in the shortest possible time to facilitate an equally rapid response.

1.5 Aim of the thesis

Since abrupt movements are still little approached and studied, this master thesis aims to detect abrupt movements in real time using inertial sensors and an LSTM neural network. Using data collected on 61 participants through a specified experimental protocol, the

network's performance was tested on signals segmented into overlapping windows to closely simulate real-time conditions. Following this, a real-time recognition protocol was developed, enabling data from inertial sensors to be directly captured, pre-processed, and analysed by the network. The study evaluated the network's ability to distinguish between impulsive and standard movements, along with the time required for each step. Timing analysis is crucial to guarantee a rapid response in safety systems and to prevent potential collisions. The steps followed in this work are outlined in *Figure 1.7*. *Table 1.1* below provides a summary of the article analysed in this chapter.

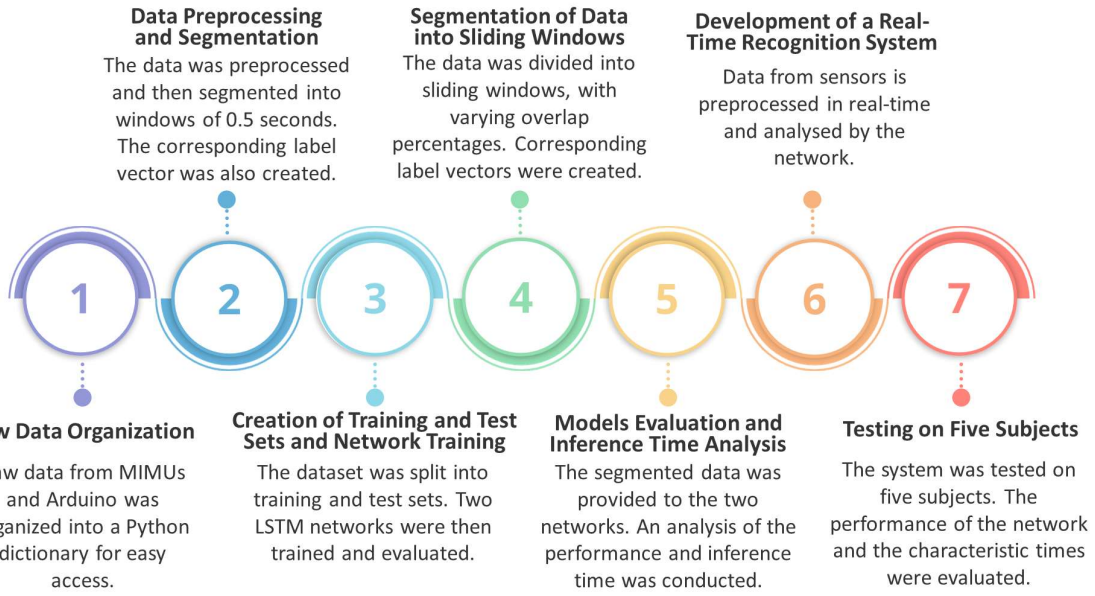


Figure 1.7. Workflow followed in the experimental work.

Table 1.1. Summary of article analysed in Chapter "1. Introduction"

Title	Authors	Year	Type of paper	Aim	Area of interest	Neural Network employed	Dataset	Prediction or Recognition	Results
Industry 4.0 And Industry 5.0 – Inception, Conception, Perception	Xu et al.	2021	Review	Comprehensive review on Industry 4.0 and Industry 5.0, with a focus on similarities and differences	Industrial	–	–	–	–
Exploring The Synergies Between Collaborative Robotics, Digital Twins, Augmentation, And Industry 5.0 For Smart Manufacturing: A State-Of-The-Art Review	Zafar et al.	2024	Review	The aim of this review is to analyze the main characteristics of collaborative robots, or 'cobots', while highlighting the benefits that the use of this technology, together with artificial intelligence, has brought to Industry 5.0	Industrial	–	–	–	–
Refining Levels of Collaboration	Aaltonen et al.	2018	Article	This article aims to define levels of collaboration	Industrial	–	–	–	–

to Support Design and Evaluation of Human-Robot Interaction in The Manufacturing Industry				between human workers and collaborative robots, facilitating the analysis of collaborative work from both human and safety perspectives					
Safe Human-Robot Collaboration – Introduction and Experimenting Using ISO/TS 15066	Rosenstrauch et al.	2017	Article	This article provides an introduction to safety standards and guidelines for risk assessment, with a detailed description of the requirements outlined for collaborative industrial robots in the technical specification ISO/TS 15066. Additionally, an experimental	Industrial	–	–	–	The experiment shows the residual hazard potential in case of incident despite compliance with ISO/TS 15066.

				use case demonstrates practical application of these guidelines.					
Human-Robot Collaboration in Industry	Vysocky et al.	2016	Review	Advantages of the use of collaborative robots in industry	Industrial	–	–	–	–
The Expanding Role of Artificial Intelligence in Collaborative Robotics for Industrial Applications: A Systematic Review of Recent Works	Borboni et al.	2023	Review	State-of-the-art research on the use of cobots in the industry, focusing on recent publications related to collaborative workspace-type robots and the application of artificial intelligence	Industrial	–	–	–	–
EEG Based Arm Movement Intention Recognition Towards Enhanced Safety in	Buerkle et al.	2021	Article	Recognition of the upper-limb movement intentions in order to increase system reaction time	Industrial	Long Short-Term Memory Recurrent Neural Network (LSTM-RNN)	EEG signals, divided into three phases: being idle, intention to move, actual movement	Recognition	The results demonstrate that EEG signals and the LSTM-RNN can be used to detect and classify the intention for arm

Symbiotic Human-Robot Collaboration				and improve safety in Human-Robot Collaboration					movement. This approach could be employed to dynamically adjust robot's speed and torque, thereby minimizing the risk of collisions.
Deep Learning-Based Multimodal Interface for Human-Robot Collaboration	Liu et al.	2018	Article	Development of a robot control interface using a deep learning algorithm for human-robot collaboration systems	Industrial	Convolutional Neural Network (CNN) and Multilayer Perceptron (MLP)	Body posture, voice command, and hand motion data	Recognition	The results demonstrate the efficiency of deep learning algorithms for classification and recognition, highlighting their potential benefits for application in human-robot collaboration.
Ensemble Of Deep Learning Techniques to Human Activity Recognition Using Smartphone Signals	Imanzadeh et al.	2024	Article	Their aim is to overcome the challenges associated with small and noisy datasets collected in real-world settings by developing a solution using an ensemble learning	Industrial and/or clinical	Ensemble of hybrid deep models	Data from the accelerometer, magnetometer, and gyroscope on the smartphone	Recognition	The proposed ensemble approach is able to classify and recognize the dataset collected via smartphone sensors. This novel approach enables improvement in accuracy and reliability of HAR in real-world applications.

				approach to achieve accurate human activity recognition (HAR).					
Integrating An LSTM Framework for Predicting Ankle Joint Biomechanics During Gait Using Inertial Sensors	Xiang et al.	2024	Article	This study aims to provide a model to predict ankle joint biomechanics, particularly angles, torques, and contact forces.	Clinical	LSTM-MLP model	Time-series data from IMU sensors	Prediction	The proposed LSTM-MLP model can identify and learn gait characteristics and patterns from acceleration and angular velocity signals, enabling accurate prediction of ankle joint biomechanics.
Deep Convolutional and LSTM	Ordóñez et al.	2016	Article	Evaluation and comparison of a deep learning	Industrial and/or clinical	Convolutional and long short-term memory	Data from IMU sensors	Recognition	The results demonstrate that this deep

Recurrent Neural Networks for Multimodal Wearable Activity Recognition				framework for activity recognition using data from wearable sensors.		recurrent layers (DeepConvLSTM)			architecture is capable of performing activity recognition using data from wearable sensors. Compared to a standard Convolutional Neural Network, it offers a good trade-off between performance and training/recognition time, and it is able to distinguish similar gestures.
Recognition of grasping patterns using deep learning for human-robot collaboration	Amaral et al.	2023	Article	Recognizing the object grasped by the operator based on the patterns of the hand and finger joints, enabling an efficient human-robot collaboration. This study focuses on evaluating the classifier's generalization ability for	Industrial	Convolutional Neural Network (CNN) and transformer	Hand landmarks detected from RGB images	Recognition	The conducted experiments emphasized the importance of active data collection to enable effective generalization of the classifier across various user behaviours and grasping patterns.

				application in real-world scenarios.					
Abrupt Movement Assessment of Human Arms Based on Recurrent Neural Networks for Interaction with Machines	Polito et al.	2023	Article	Distinction between normal and abrupt movements during a typical repetitive industrial task	Industrial	Long Short-Term Memory	Forearms accelerations measured by MIMUs	Recognition	The deep learning network adopted and the proposed pre-classification methods for MIMUs accelerations demonstrate potential for identifying abrupt movements
Deep Learning Techniques to Identify Abrupt Movements in Human-Robot Collaboration	Polito et al.	2023	Article	Identification of human abrupt movements using a recurrent neural network trained with wrist acceleration elaborated with two different methodologies	Industrial	Long Short-Term Memory	Accelerations of the wrist recorded through MIMUs	Recognition	The results demonstrated that the methodology adopted to address real-time situations achieved higher classification performance. Therefore, the deep learning network and the pre-classification method employed are suitable for identifying human abrupt movement.
Detection Of Upper Limb	Digo et al.	2024	Article	Training a recurrent	Industrial	Long Short-Term Memory	Forearms acceleration	Recognition	The results demonstrate that

Abrupt Gesture for Human-Machine Interaction Using Deep Learning Techniques				neural network to distinguish between standard and abrupt gestures, aiming for effective real-time gesture classification			signals recorded by MIMUs		the data pre-processing is fundamental for achieving effective network training. Specifically, reducing the window duration leads to improved classification. Furthermore, the results show that classification time can be reduced without negatively impacting the results, enabling real-time classification.

2. MATERIALS AND METHODS

2.1 Materials

2.1.1 Inertial sensors

For the quantitative analysis of participants' movement, the necessary data were collected using Opal™ V2R inertial sensors, produced by APDM WEARABLE TECHNOLOGIES INC. An example of wearable sensor is displayed in *Figure 2.1*.

*Figure 2.1. Opal V2R wearable sensor (*Precision Motion for Research*, n.d.)*

These are small, lightweight, wireless sensors that can be worn on the body and use micro-electromechanical systems to detect kinematic movement parameters. Specifically, these sensors are equipped with:

- Two tri-axial accelerometers, with ranges of ±16g and ±200g, which provide the instantaneous values of the three components of acceleration.
- A tri-axial gyroscope, with a range of ±2000 deg/s, which measures angular velocity.
- A tri-axial magnetometer, with a range of ±8 Gauss, which measures the components of the magnetic field along three directions. It is used to correct gyroscope drift and provides a stable reference relative to the magnetic north.

By combining data from these three sensors, quaternions providing, information about the object's orientation in space can be obtained.

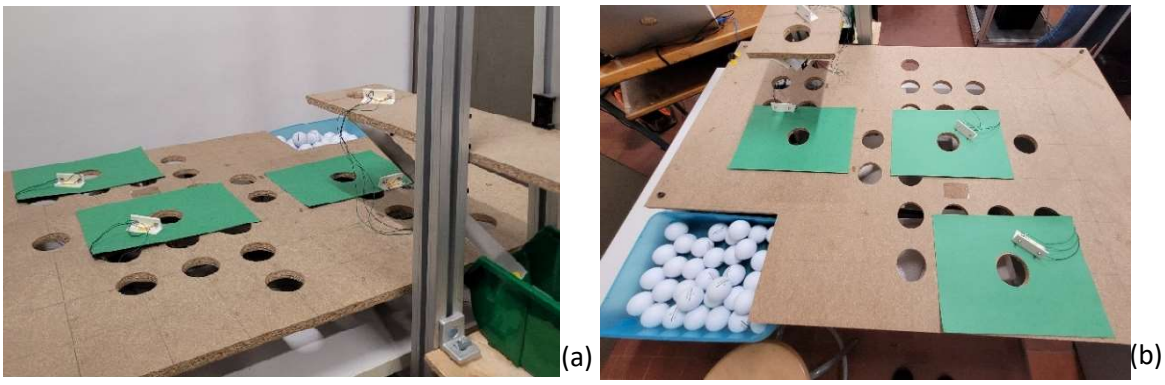
One of the advantages of inertial sensors is their ease of application to specific body segments. In our study, we focused on the movement of the upper torso, and five sensors were attached to various body areas:

- On the sternum
- On the upper right arm, just below the deltoid muscle
- On the upper left arm, just below the deltoid muscle
- On the distal part of the right forearm, near the wrist
- On the distal part of the left forearm, near the wrist

Data were collected using the proprietary software Motion Studio, with a sampling frequency set at 200 Hz.

2.1.2 Experimental set up

For data acquisition, a custom experimental setup was specifically designed for this study (Digo, Polito, Pastorelli, et al., 2024; Polito et al., 2023b), as shown in *Figure 2.2*.



(a) (b)

Figure 2.2. Experimental setup viewed from the front (a) and from above (b).

The setup consists of:

- A table and a stool
- Two chipboard panels: one (i) with 30 holes, each 6 cm in diameter, placed on the table and raised 10 cm on four legs, and another (ii) with a single hole of the same diameter, positioned higher using two lateral support rods.
- A container measuring 22 cm x 33 cm x 7 cm, holding 30 golf balls with a diameter of 43 mm.
- A 1-meter aluminium slide to guide the balls from the single hole in panel (ii) to a container measuring 12.8 cm x 20.4 cm x 9.6 cm.
- Three containers positioned between the table and the panel with 30 holes.
- Eight LED lights (four red, four green), mounted on 3D-printed supports.
- Three green cards indicating the correct holes based on the participant's anthropometric measurements.

- An Arduino Nano board with an ATmega328 microcontroller, connected to a computer via USB, and operated using Arduino software with instructions implemented in MATLAB. This controls the activation of the LEDs and a buzzer (auditory signal).

Each trial consists of 30 movements, of which 26 are normal and 4 are abrupt. The green LEDs signal the standard movements and are activated at a frequency of 20 beats per minute (every 3 seconds), while the red LEDs indicate abrupt movements, lighting up 0.5 seconds after the green LEDs. The MATLAB instructions are set so that the first five movements are always normal, and at least two of the four abrupt movements must be accompanied by a buzzer sound. Additionally, two of the abrupt movements occur during the first half of the experiment, while the remaining two take place within the final 15 movements.

2.2 Methods

2.2.1 Long Short-Term Memory Neural Network

The objective of this study is to recognize abrupt movements using an LSTM neural network. This neural network was developed using Keras, a high-level library written in Python facilitating the creation of deep learning models. Keras is integrated with TensorFlow, a framework that manages optimization and computational backend operations. Information on how to implement a neural network using Keras can be found in the official Keras documentation (*Keras 3 API Documentation*, n.d.). This resource provides detailed guides and examples for building and training neural networks with Keras.

The first step was to define a sequential model, characterized as a plain stack of layers where each layer has exactly one input tensor and one output tensor. The layers were defined as follows:

- InputLayer: This layer defines the shape of the input, specifying the number of time steps in the sequence and the number of features present.
- LSTM layer: This is the recurrent layer, where the number of hidden units is set, corresponding to the number of neurons.

- Dropout layer: Used to reduce overfitting by randomly setting a fraction of the units to zero during training at a user-defined rate.
- Dense layer: A densely connected layer that performs the final classification. For binary classification, this layer has a single neuron with a *'sigmoid'* activation function. The output is a probability value between 0 and 1, representing the likelihood of belonging to the positive class.

Next, it was necessary to compile the model, a critical step to configure it for training. During this phase, essential functions for model training are specified:

- Optimizer: This determines how the model updates its weights. The most common choice is Adam optimization, a stochastic gradient descent method that leverages adaptive estimation of first-order and second-order moments.
- Loss: This specifies the loss function that the model will use during training. The loss function measures the discrepancy between the model's predictions and true values, guiding the optimizer in updating the model's weights. For binary labels (0 and 1), the most suitable loss function is 'binary_*crossentropy*,' which handles binary classification tasks effectively.
- Metrics: A metric is a function used to evaluate the model's performance. Unlike loss functions, metrics do not influence weight updates during training but provide insights into model performance. An example of a commonly used metric is *accuracy*.

At this stage, the model is ready for training using the *model.fit* method. To proceed, it is necessary to have the input data, which is used to train the model; the target data, or labels, that the model aims to predict, which is used to calculate the loss during training; and the validation data, a separate dataset used to evaluate the model performance after each epoch. An epoch refers to one complete iteration over the entire input and target data provided. The LSTM network expects the data to be provided with a specific array structure in the form of *[samples, time steps, features]*. Each dimension represents:

- *Samples*: the number of sequences in the dataset.
- *Time steps*: the length of the time sequence.
- *Features*: the number of variables observed at each time step.

The *model.fit() method* returns a *history* object, which records the loss and metric values over each epoch for both the training and validation datasets.

The specifications used to implement the model in this study are detailed in *Figure 2.3*.

```
num_hidden_units = 100
mini_batch_size = 27
max_epochs = 20

model = Sequential([
  InputLayer(shape=(timesteps, n_features)),
  LSTM(num_hidden_units, return_sequences=False),
  Dropout(0.5),
  Dense(1, activation='sigmoid')
  ])

optim = Adam(clipnorm=2.0)
model.compile(optimizer=optim,
              loss='binary_crossentropy',
              metrics=['accuracy'])

history = model.fit(
  x_train, TrainLabel,   # input data and target data
  epochs=max_epochs,
  batch_size=mini_batch_size,
  validation_data=(x_val, ValidationLabel),
  shuffle=True,
  verbose=0)
```

Figure 2.3. Python code for creating and training the network.

Once the network is trained, predictions can be generated using the *model.predict()* function. This function takes new data as input, formatted to be compatible with the network, and outputs predictions. In binary classification, the output is a probability between 0 and 1, with higher values indicating a greater likelihood of belonging to the positive class. However, to obtain binary values (0 or 1), the probabilities are converted into class labels. Typically, a threshold of 0.5 is set to classify outputs into one of the two classes. If the output is above 0.5, it is classified as positive (label = 1); otherwise, it is classified as negative (label = 0). This threshold can be adjusted according to the specific task requirements. In the presented work, the two classes are highly unbalanced, which makes it more likely that standard windows will be incorrectly identified as abrupt, significantly increasing false positives. To address this, the threshold was set to 0.9, ensuring that only windows with high probability of being abrupt are classified as such.

2.2.2 Training the Network: Data Collection and Preprocessing Protocol

Data Collection Protocol

The network was trained using data from a database of 61 participants (Digo, Polito, Caselli, et al., 2024) who performed a specific task designed to simulate a typical industrial work environment. Specifically, the task was a pick-and-place activity, where participants were required to pick up a golf ball from a container and place it in a hole, indicated by a LED light. The possible movement directions, shown in *Figure 2.4*, included four orientations: anteroposterior (AP), oblique (OB), mediolateral (ML), and vertical (V).

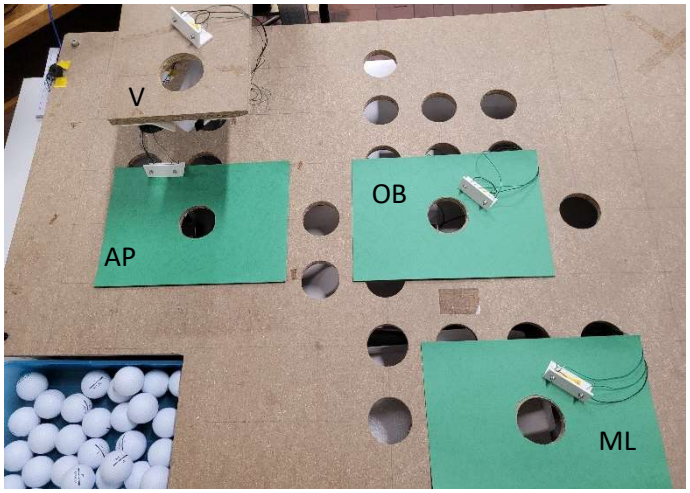


Figure 2.4. Experimental setup viewed from above, highlighting the four different directions: anteroposterior (AP), oblique (OB), mediolateral (ML), and vertical (V).

Each trial consists of 30 total movements, of which 26 are normal and 4 are abrupt. The movements are guided by the LEDs according to the following three scenarios:

1) Green LED activation (*Figure 2.5a*): The participant picks up the ball and places it in the designated hole, aiming for the smoothest possible movement (normal movement).
2) Red LED activation (*Figure 2.5b*): This occurs 0.5 seconds after a green LED is activated. In this case, the participant, initially moving toward the hole indicated by the green LED, must ignore the previous instruction and moves the ball quickly in a different direction to the hole indicated by the red LED, simulating an abrupt movement.

3) Buzzer activation (*Figure 2.5c*): After 0.5 seconds from green LED activation, a sound signal is emitted. As in case (2), the participant must disregard the green LED instruction and raises their arm vertically as quickly as possible, simulating another abrupt movement.

Figure 2.5. (a) Green LED activation: the participant picks up the ball and places it in the designated hole. (b) Red LED activation: the participant moves the ball quickly in a different direction to the hole indicated by the red LED. (c) Buzzer activation: the participant raises their arm vertically as quickly as possible.

Data collection followed a protocol divided into four phases:

1) Phase 1: Collection of anthropometric data:
    a. Age
    b. Gender
    c. Height (cm)
    d. Weight (kg)
    e. Dominant hand
    f. Arm length (cm)
    g. Forearm length (cm)
2) Phase 2: Placement of the five MIMU sensors using elastic straps, positioned as described in Section 2.1.1.
3) Phase 3: Task simulation to assess the correct distance of the participant from the table and to determine the appropriate holes for the easiest reach. Additionally, this phase allowed the participant to become familiar with the task and equipment.
4) Phase 4: Data acquisition, during which the participant performed the 30 movements guided by LED activation. Each trial lasted 90 seconds and was repeated in three different configurations:
    a. Participant seated facing the table, using the right arm for movements (Trial Frontal right – FR_r)

b. Participant seated facing the table, using the left arm for movements (Trial Frontal left – FR_l)
c. Participant seated sideways to the table, using the left arm for movements (Trial Lateral left – LA_l)

Data Preprocessing protocol

Once key features such as acceleration, angular velocity, and quaternions were extracted from the raw sensor data, they were further processed to serve as inputs to the LSTM network.

First, the gravity component was removed from the acceleration data using the rotation matrix derived from the quaternions. Next, the norms of both acceleration and angular velocity were calculated. Temporal data was then segmented into 3-second windows, each corresponding to a single movement and labelled as either normal (label = 0) or abrupt (label = 1). In *Figure 2.6*, the acceleration signal of subject 1, trial FR_r, segmented into 3-second windows is shown as an example.

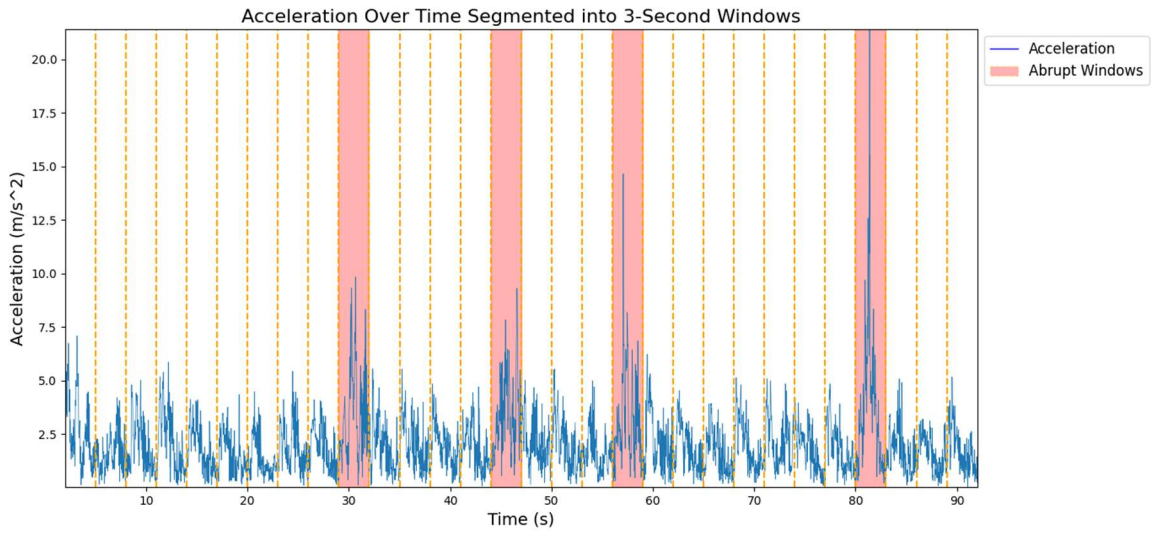


Figure 2.6. Acceleration signal of Subject 1 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the four abrupt movements highlighted (red rectangles).

Following this, each 3-second window was further divided into 0.5-second sub-windows, obtaining six sub-windows per movement, each associated with its corresponding label. The segmented acceleration signal of subject 1, Trial FR_r, segmented into 0.5-second windows is shown in *Figure 2.7*.

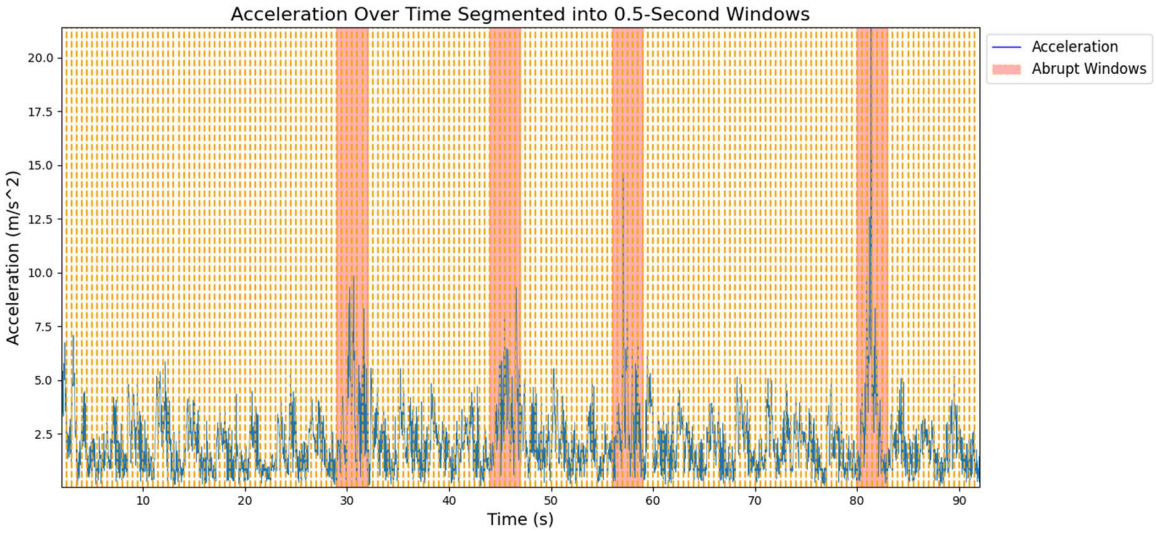


Figure 2.7. Acceleration signal of Subject 1 during trial FR_r, segmented into 0.5-second windows (orange dashed vertical lines), with the four abrupt movements highlighted (red rectangles).

Among these six sub-windows, only sub-windows 2, 3, and 4 (representing the intervals from 0.5 to 1s, 1 to 1.5s, and 1.5 to 2s, respectively) were classified as abrupt, as they capture the peak acceleration. An example of the selected sub-windows is highlighted in *Figure 2.8*. A new dataset was constructed from these identified abrupt sub-windows and an equal number of normal sub-windows.

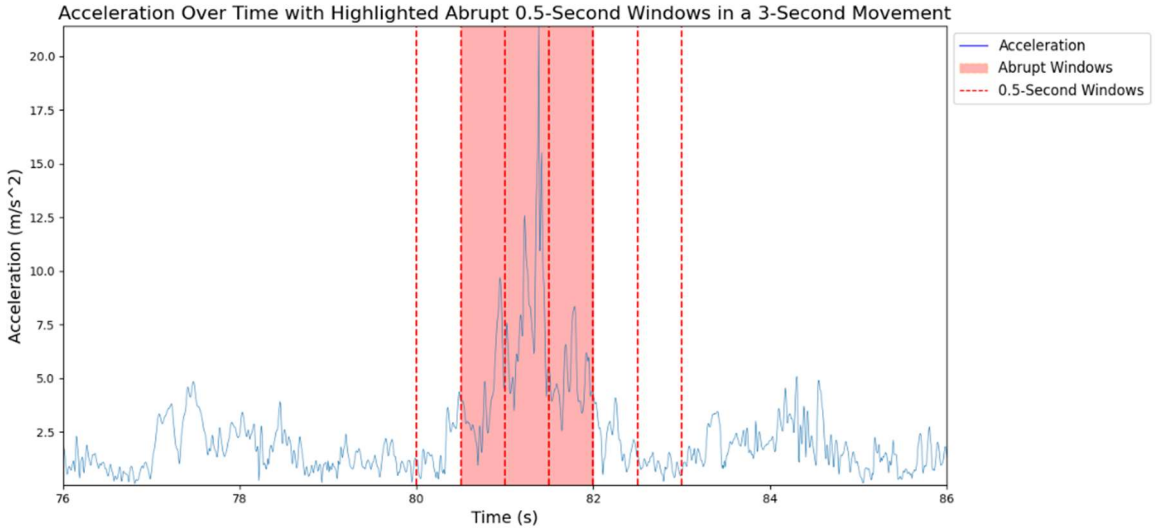


Figure 2.8. Acceleration signal of Subject 1 during trial FR_r, with three abrupt 0.5-second windows highlighted (red rectangles).

For training the network, the dataset was split into two parts: 80% was used for training, and the remaining 20% for testing, to evaluate the model's ability to generalize on new data. The training set was further divided using k-fold cross-validation, with k=5. This method involves dividing the set into five folds, using k−1 folds for training and one for validation, iterating through each fold. The model achieving the highest accuracy across these iterations was selected to classify data in the test set.

Finally, the segmented data was organized into arrays compatible with the LSTM network format, as described in Section 2.2.1. Two networks were trained: one using only acceleration data (Network 1) and another using both acceleration and angular velocity data (Network 2).

2.2.3 Sliding windows

To evaluate the network performance for approaching real-time recognition, the sliding windows approach was adopted. Each window has a fixed length and a step size that moves it forward incrementally. Since the step size is shorter that the window length, consecutive windows overlap.

In this study, the window length is fixed at 0.5 seconds, while the overlap percentage, and thus the step size, varies to assess its influence on network performance. The step size is calculated as the difference between the window length in samples and the overlap length, also in samples. Using this, the total number of windows can be determined with the following *Equation (1)*:

$$number\ of\ windows = \frac{x - n_{samples}}{step\ size} + 1 \quad (1)$$

Where:

- x: signal length (in samples)
- $n_{samples}$: number of samples in each window of fixed length
- $step\ size$: number of samples to shift to the next window

To segment the data into the calculated number of overlapping windows, a for loop was implemented to iterate over the signal, advancing by the determined step size. At each iteration, the code extracts and stores the data for each time window.

Once the signal has been divided, it is essential to assign a label to each window. For this purpose, windows that correspond to the interval used for training the network were classified as abrupt (see Section 2.2.2, *Figure 8*). Specifically, in a 3-second movement, the interval from 0.5 s to 2 s is considered abrupt. To calculate the number of abrupt windows within this interval, *Equation (1)* can be applied, using a signal length of 1.5 s.

The segmented data are provided as input to the network for recognition. The output is a prediction vector from which the recognized movements are derived. These predictions are then compared to the actual movements to construct a confusion matrix, a 2x2 table where the rows represent the actual classes, and the columns represent the predicted classes. A generic confusion matrix is illustrated in *Figure 2.9*.

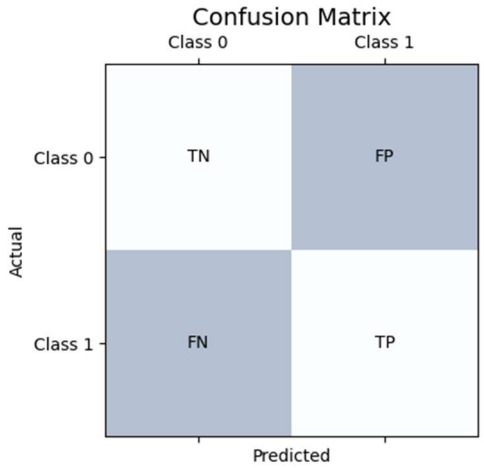


Figure 2.9. Generic example of a confusion matrix, with actual classes along the rows and predicted classes along the columns.

Referring to *Figure 2.9*, each value in the matrix has a specific meaning:

- True Negative (TN): the number of normal movements (class 0) correctly identified as normal (class 0).
- False Positive (FP): the number of normal movements (class 0) misclassified as abrupt (class 1).
- False Negative (FN): the number of abrupt movements (class 1) misclassified as normal (class 0).
- True Positive (TP): the number of abrupt movements (class 1) correctly identified as abrupt (class 1).

From the confusion matrix, various metrics can be calculated to evaluate the network's performance. The following metrics were specifically computed:

- Balanced Accuracy: arithmetic mean of sensitivity and specificity. Sensitivity measures the model's ability to correctly identify true positives, while specificity measures its ability to correctly identify true negatives. This metric is particularly useful for unbalanced datasets, as it averages the correct classification rates for both classes, giving them equal weight. The formula to calculate balanced accuracy is provided below (*Equation (2)*):

$$(2) \qquad Balanced\ Accuracy_{\%} = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) \times 100$$

- Precision (*positive*): evaluates the percentage of true *positive* predictions (TP) among all *positive* predictions (TP and FP). It is computed using *Equation (3)* below:

$$(3) \qquad Precision_{\%} = \frac{TP}{TP + FP} \times 100$$

- Recall (*positive*): measures the percentage of true *positive* predictions (TP) out of the total *positive* class (TP and FN). It is calculated using *Equation (4)* below:

$$(4) \qquad Recall_{\%} = \frac{TP}{TP + FN} \times 100$$

- Macro F1-score: arithmetic mean of the F1-scores calculated for each class individually. Each per-class F1 score is the harmonic mean of the precision and recall for that specific class. This metric is particularly useful for unbalanced classes, as it is independent of their distribution. The formulas for calculating the per-class F1-score (*Equation (5)*) and the Macro F1-score (*Equation (6)*) are shown below:

$$(5) \qquad F1score_{class} = 2 \times \frac{Precision_{class}\ x\ Recall_{class}}{Precision_{class} + Recall_{class}}$$

$$(6) \qquad Macro\ F1score_{\%} = \frac{1}{2}\left(F1score_{positive} + F1score_{negative}\right) \times 100$$

- Specificity: measures the percentage of true negative predictions (TN) out of the total negative class (TN and FP). It is calculated using the *Equation (7)* below:

$$(7) \qquad Specificity_{\%} = \frac{TN}{TN + FP} \times 100$$

Finally, a timing analysis was performed to evaluate the network's ability to quickly classify a movement and its applicability to a real-time context. The Python function *time.perf_counter()* was used to measure the time required for classification. Acting as a timer, it starts at the beginning of the classification process and stops at the end. Two inference times were calculated:

1) Average inference time: The average time taken by the network to classify the data for a single subject. After measuring the time for all 61 subjects, the average was calculated.
2) Total inference time: The total time required to classify all 61 subjects. The timer started at the beginning of the for loop, where subjects were analysed individually, and stopped once the entire loop was completed.

2.2.4 Real-time detection of abrupt movements

After evaluating the network performance with sliding windows, a real-time protocol for recognizing abrupt movements was developed. The goal is to analyse sensor data immediately, identifying the type of movement within a few milliseconds. This protocol involves a data collection from five new subjects using the same methodology outlined in Section 2.2.2 for training data. A Python script is employed for the analysis.

This code involves several steps:

- Step 1: Load the pre-trained LSTM model with acceleration data.
- Step 2: Configure the sensors using a Python script provided by the sensor developer, named *'autoconfigure_system.py'*.
- Step 3: Establish the communication between the sensors and the Python script using additional code, named *'stream_data.py'*, provided by the developer. This code returns the data acquired from the MIMU sensors.
- Step 4: Once the communication with the sensors is established, a connection is opened between Python and the MATLAB code controlling Arduino for LED activation.
- Step 5: Save acceleration and quaternion data for a single 3-second movement in a NumPy array, facilitating data preparation for the network. The time required to read and save the data is measured using Python's *time.perf_counter()* function.

- Step 6: Pre-process the data:
- Remove gravitational acceleration using the rotation matrix derived from the quaternions.
- Calculate the acceleration norm.
- Segment the data into overlapping windows with 99% overlap.
- Organize the data in the required format for the network: [samples, time steps, features].
- The time required to execute these steps is measured as in Step 4
- Step 7: Provide the pre-processed data to the network for recognition, generating an output vector that contains predictions for each window. The inference time, which is the time required for the network to process the input and return an output, is evaluated, using the same function as in Step 4 and 5.
- Step 8: Interpret the prediction vector. If at least one window is classified as abrupt, the entire movement is identified as such. If an abrupt movement is recognized, a red window appears; if the movement is standard, a green window is displayed.
- Step 9: Saving Results: The movement recognition outcomes and the times associated with Steps 4, 5, and 6, are saved in vectors for later performance analysis.

The Python script is designed for continuous streaming and saving of data while simultaneously recognizing movements. This is achieved using Python's *threading* library, which allows parallel operations. Two threads are created: one for streaming and pre-processing data, which encompasses Steps 4, 5 and 6, and another for recognition, covering Steps 7 and 8. Once the data is ready, it is passed from the first thread to the second one for recognition.

The performance of the system was then evaluated, focusing on both the network's recognition capabilities and the timing, which are crucial for real-time applications.

For the recognition assessment, the LED activation data, corresponding to the normal or abrupt movements performed by the subject, was compared to the predicted movements. A confusion matrix was created, and the same metrics as in Section 2.2.3 were calculated.

In the timing analysis, three main steps were evaluated: the time required to transmit acceleration data from the inertial sensors to the analysis system (i.e., streaming time, Step 4); the time to prepare the data (i.e., pre-processing time, Step 5); and the time required by the network to classify the movement (i.e., inference time, Step 7). For each of these steps, both the mean and standard deviation were calculated at two levels: intra-subject (the mean and standard deviation calculated across movements for a single subject) and inter-subject (calculated across all five subjects).

To further investigate the real-time system's performance, an analysis was conducted on the distribution of errors made by the network across the movements, divided into three intervals:

1. Movements from 1 to 10
2. Movements from 11 to 20
3. Movements from 21 to 30

A chi-square test for independence was performed to assess whether the observed error frequencies in the three intervals matched the expected frequencies. Pairwise comparisons were then conducted to determine whether statistically significant differences (p-value < 0.05) existed in the error distribution among the intervals.

2.2.5 Participants

Five new subjects were recruited to participate in the test. *Table 2.1* below provides a summary of the participants' data.

Table 2.1. Data of the five participants: gender, age, height, weight, BMI, dominant hand, forearm length, and arm length. The inter-subject mean ± standard deviation for age, height, weight, BMI, forearm length, and arm length were calculated.

Gender	Age	Height (cm)	Weight (kg)	BMI (kg/m^2)	Dominant hand	Forearm length (cm)	Arm length (cm)
4F, 1M	23.4±0.49	162.2±9.22	55.8±12.07	20.73±2.56	4dx, 1sx	31.6±2.24	26.2±2.48

3. RESULTS AND DISCUSSION

3.1 Sliding windows

3.1.1 Segmentation into overlapping windows

The acceleration and angular velocity data from the three trials conducted with the 61 subjects were divided into overlapping windows. Each trial lasted 90 seconds. A sampling rate of 200 Hz and a window length of 0.5 seconds were defined. The following parameters were calculated following the procedure outlined in Section 2.2.3: overlap percentage, step size, abrupt windows per movement, windows per trial, total windows per subject and total abrupt windows per subject. The resulting values are shown in *Table 3.1*.

Table 3.1. Step size, abrupt windows per movement, windows per trial, total windows per subject and total abrupt windows per subject for different percentages of overlap (50%, 75%, 90%, 95%, 99%) are presented.

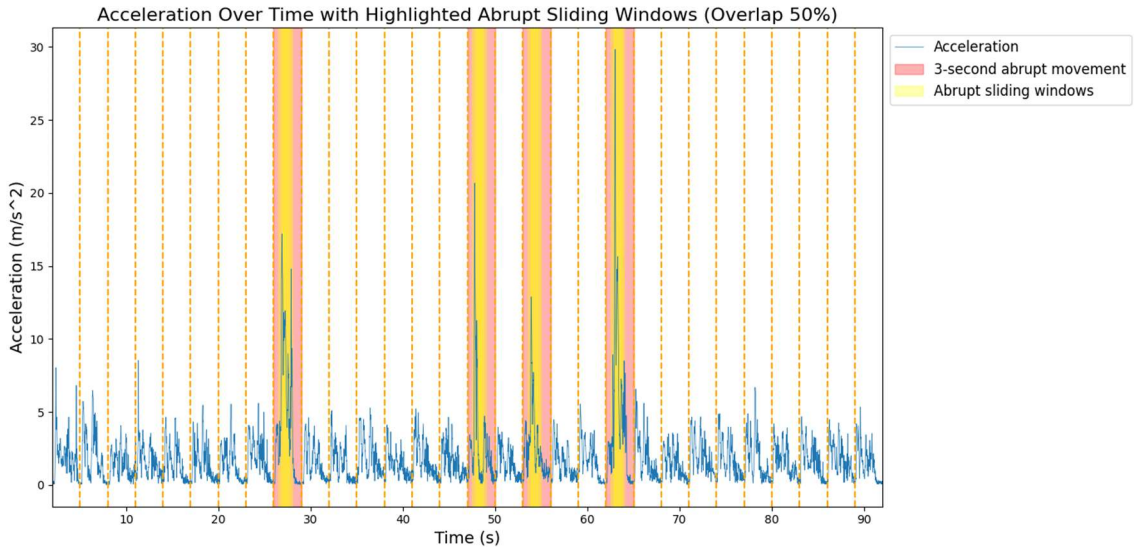
Overlap (%)	Step size (samples)	Abrupt windows per movement	Windows per trial	Total windows per subject	Total abrupt windows per subject
50	50	5	359	1077	60
75	25	9	717	2151	108
90	10	21	1791	5373	252
95	5	41	3581	10743	492
99	1	201	17901	53703	2412

The dataset is highly unbalanced, containing a significantly larger number of normal windows compared to abrupt ones. In fact, only about the 5% of a subject's windows is classified as abrupt. In cases like this, metrics such as accuracy and F1-score may not reflect the model's ability to detect the minority class. As suggested by Rivera and colleagues, performance metrics like balanced accuracy and macro F1-score are particularly useful as they are not affected by class distribution, providing a more balanced view of the performance of the model (Rivera et al., 2017).

In *Figures 3.1, 3.2, 3.3, 3.4, and 3.5*, the acceleration signal from Subject 26, trial FR_r, is shown, with the abrupt sliding windows highlighted for each percentage of overlap. A

single abrupt movement is enlarged to illustrate how the overlapping windows appear for an individual abrupt movement.

(a)



(b)

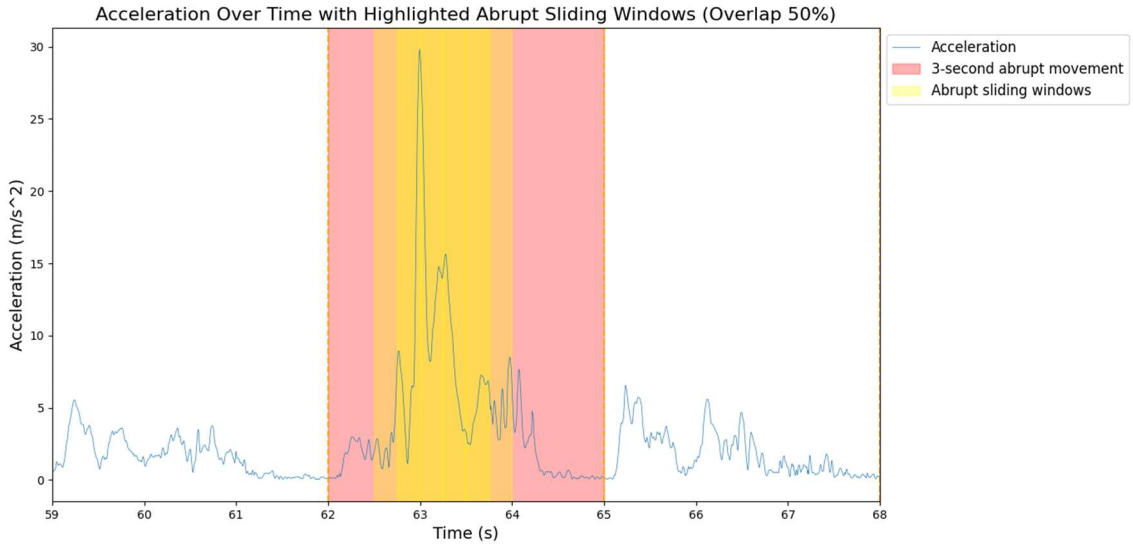
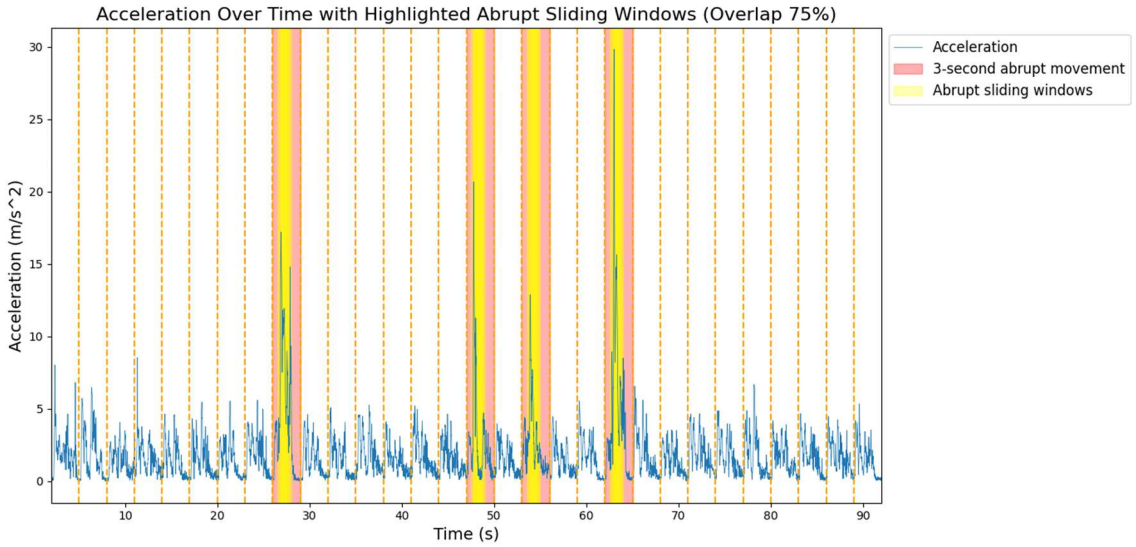


Figure 3.1. (a) Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the four abrupt movements (red rectangles) and the abrupt sliding windows with 50% overlap highlighted (yellow rectangles). (b) Zoom on the fourth abrupt movement.

(a)



(b)

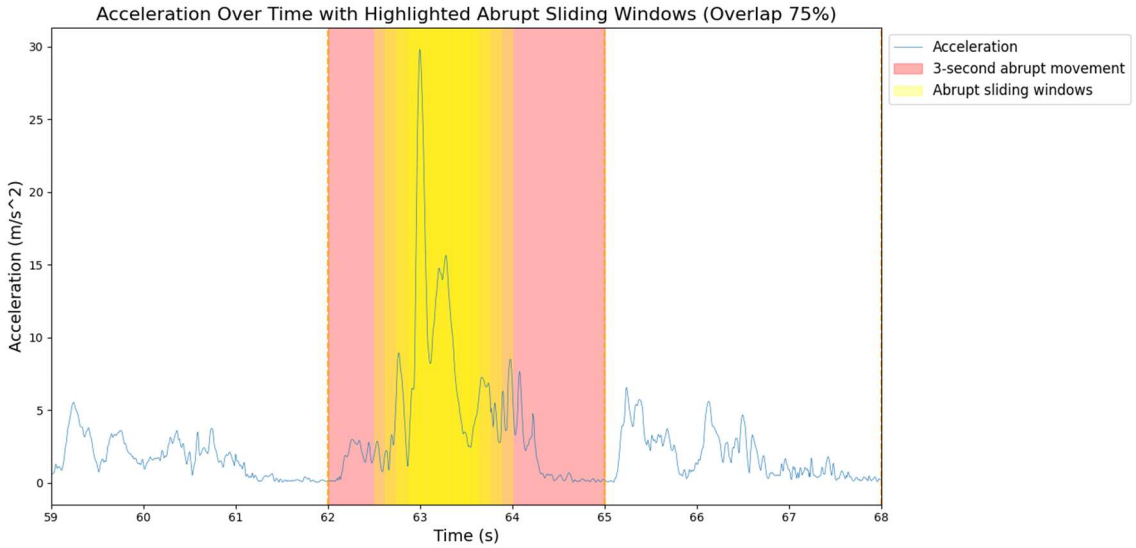
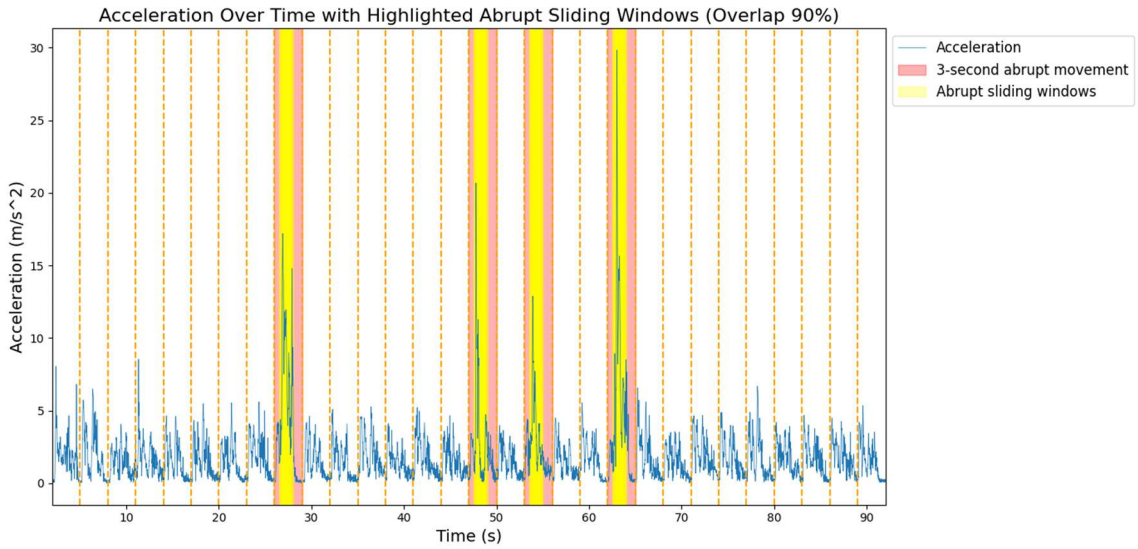


Figure 3.2. (a) Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the four abrupt movements (red rectangles) and the abrupt sliding windows with 75% overlap highlighted (yellow rectangles). (b) Zoom on the fourth abrupt movement.

(a)



(b)

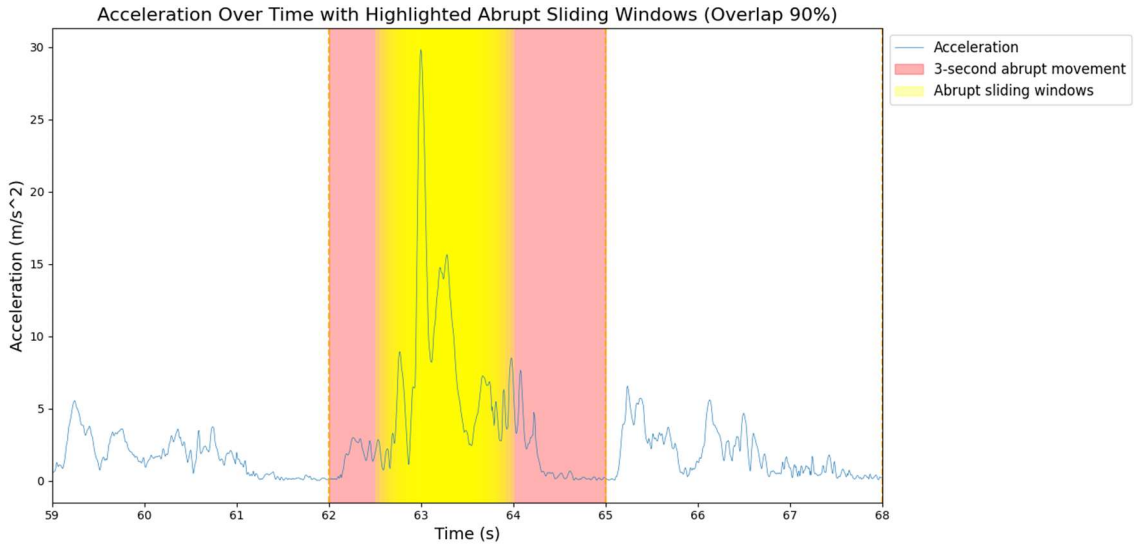
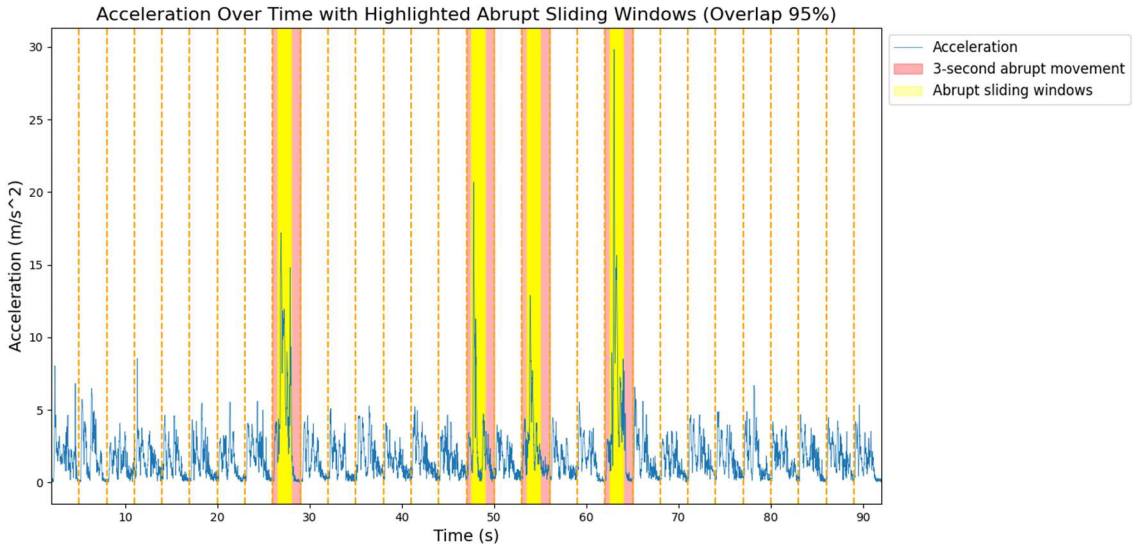


Figure 3.3. (a) Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the four abrupt movements (red rectangles) and the abrupt sliding windows with 90% overlap highlighted (yellow rectangles). (b) Zoom on the fourth abrupt movement.

(a)



(b)

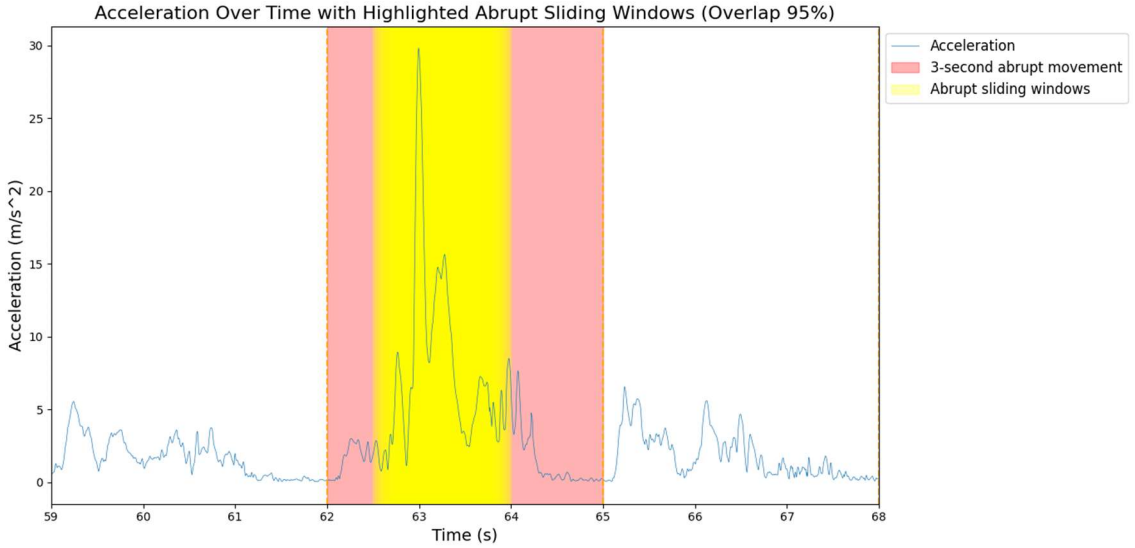
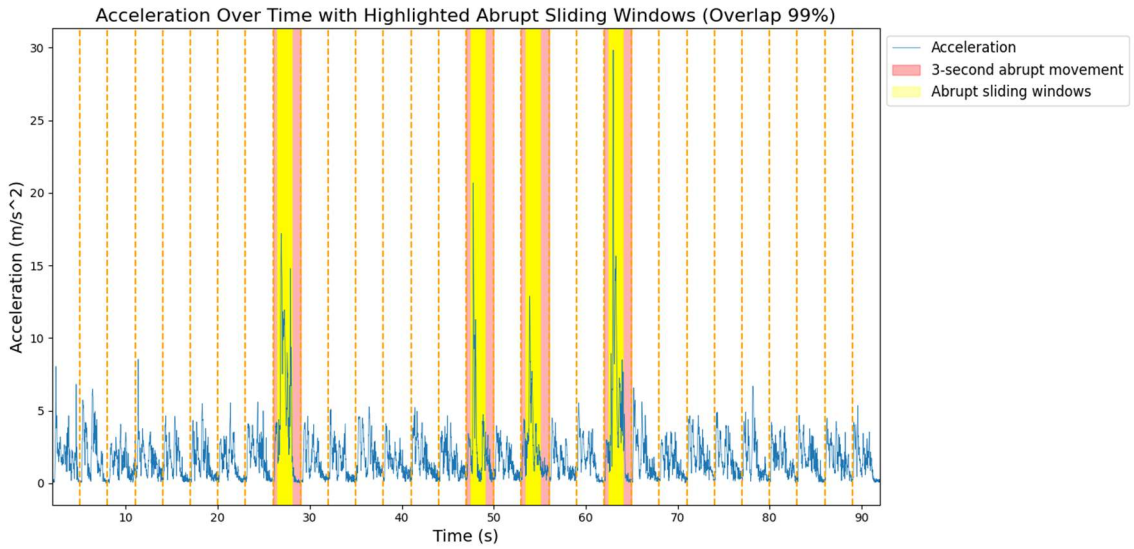


Figure 3.4. (a) Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the four abrupt movements (red rectangles) and the abrupt sliding windows with 95% overlap highlighted (yellow rectangles). (b) Zoom on the fourth abrupt movement.

(a)



(b)

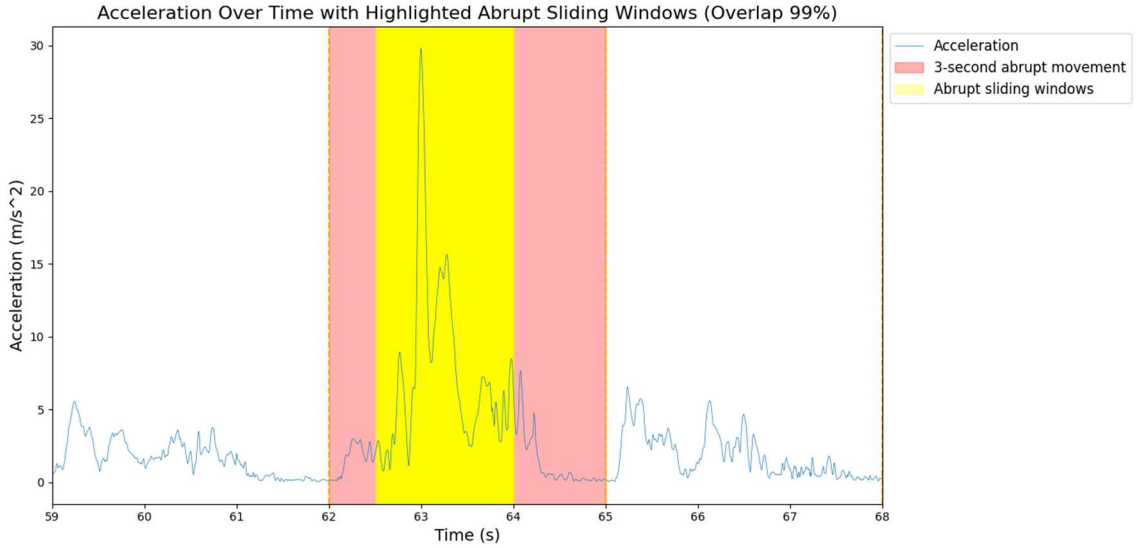


Figure 3.5. (a) Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the four abrupt movements (red rectangles) and the abrupt sliding windows with 99% overlap highlighted (yellow rectangles). (b) Zoom on the fourth abrupt movement.

3.1.2 Comparison between actual and predicted abrupt windows

Once prepared, the data were processed by the two networks: specifically, acceleration data were provided to Network 1, while both acceleration and angular velocity data were provided to Network 2. With the obtained prediction vector, it was possible to compare the actual and predicted abrupt windows. Referring to Subject 26, trial FR_r, *Figures 3.6, 3.7, 3.8, 3.9, and 3.10* visually compare the results for each percentage of overlap.

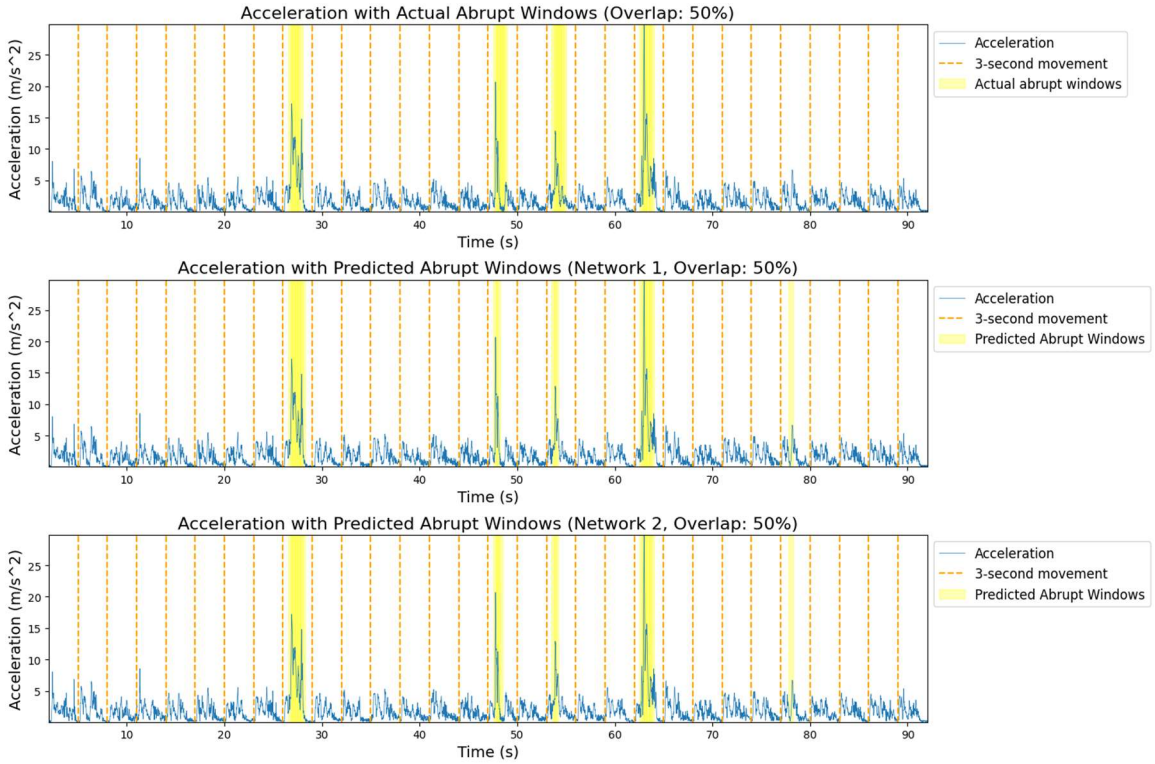


Figure 3.6. Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the actual and predicted abrupt sliding windows (50% overlap) highlighted (yellow rectangles) for both Network 1 and Network 2.

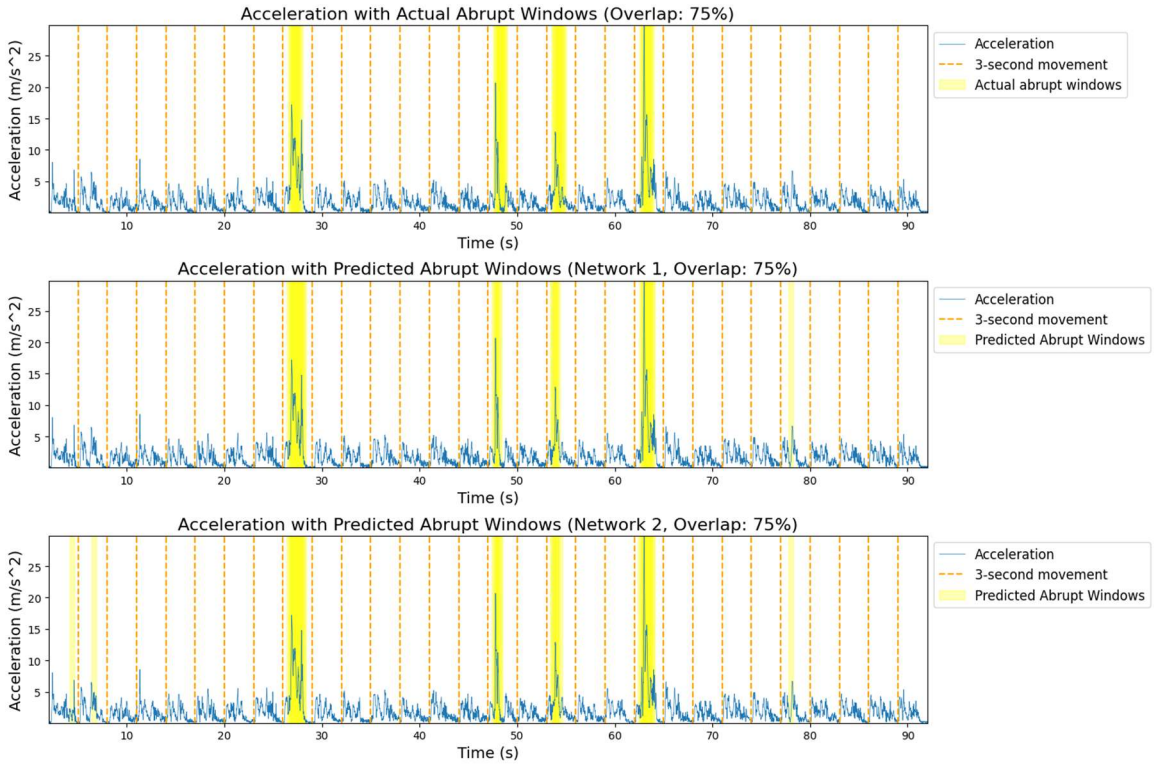


Figure 3.7. Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the actual and predicted abrupt sliding windows (75% overlap) highlighted (yellow rectangles) for both Network 1 and Network 2.

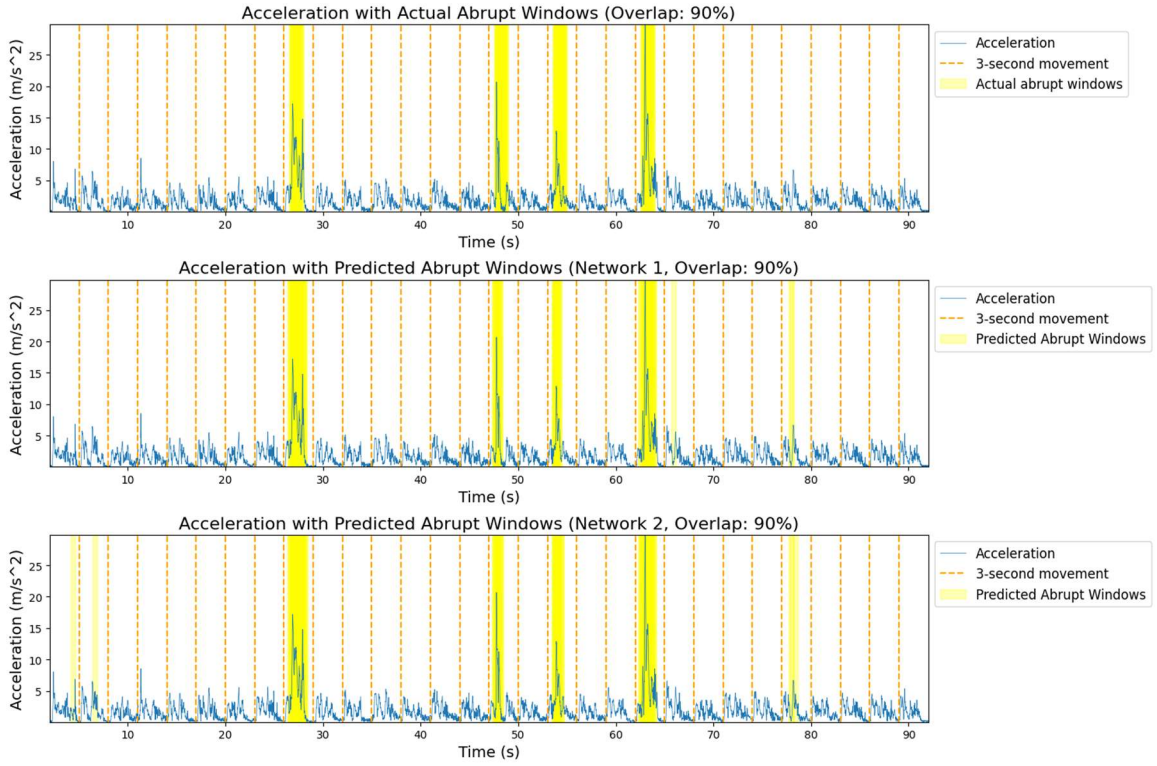


Figure 3.8. Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the actual and predicted abrupt sliding windows (90% overlap) highlighted (yellow rectangles) for both Network 1 and Network 2.

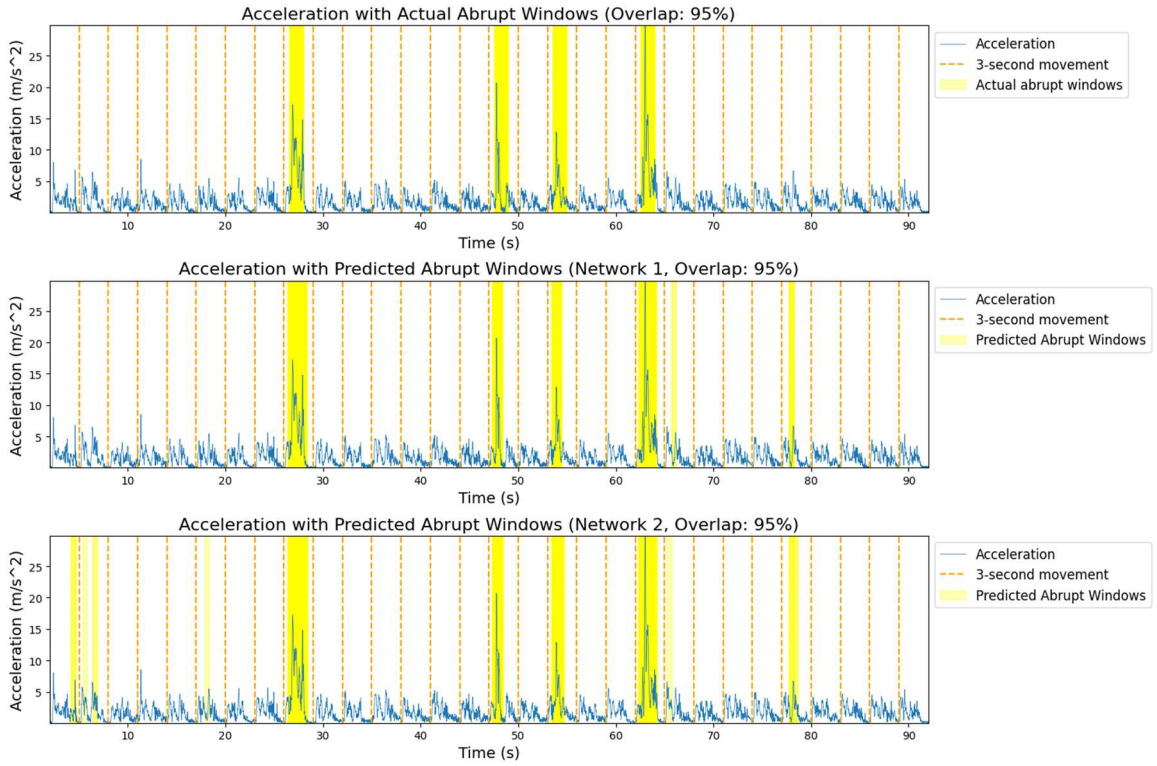


Figure 3.9. Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the actual and predicted abrupt sliding windows (95% overlap) highlighted (yellow rectangles) for both Network 1 and Network 2.

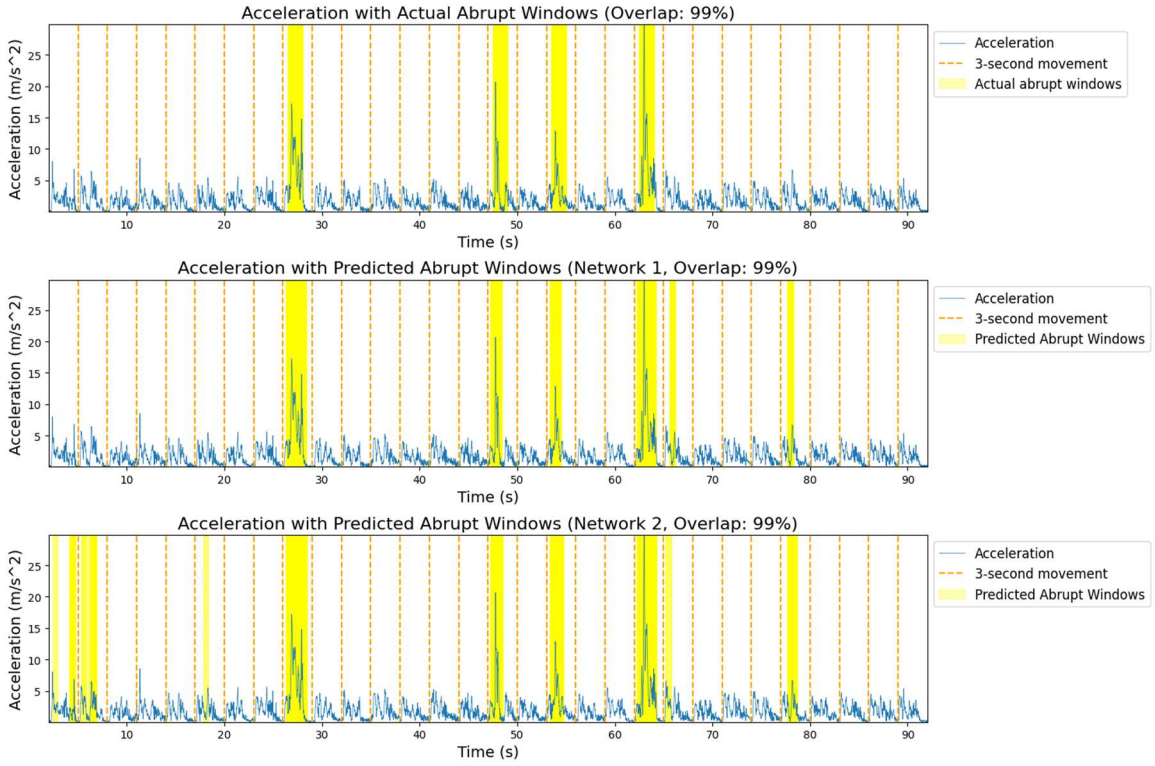
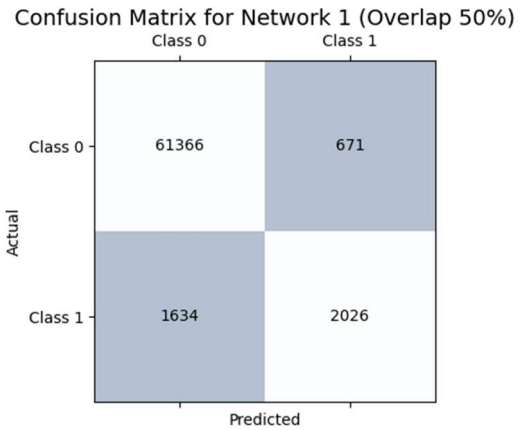
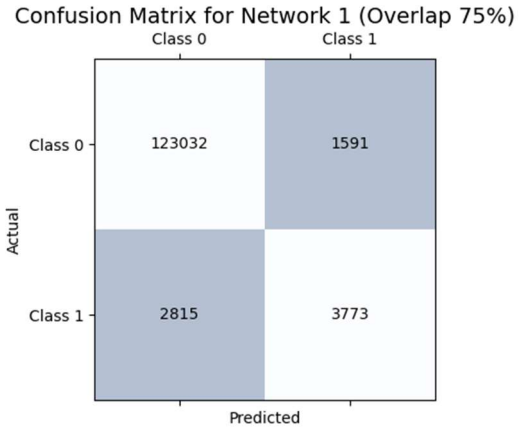


Figure 3.10. Acceleration signal of Subject 26 during trial FR_r, segmented into 3-second windows (orange dashed vertical lines), with the actual and predicted abrupt sliding windows (99% overlap) highlighted (yellow rectangles) for both Network 1 and Network 2.

3.1.3 Performance analysis

To evaluate the performance of the two networks, confusion matrices were first constructed by comparing the prediction vector with the vector of true labels. In *Figures 3.11 and 3.12*, the confusion matrices for each overlap percentage for Network 1 and Network 2, respectively, are presented.





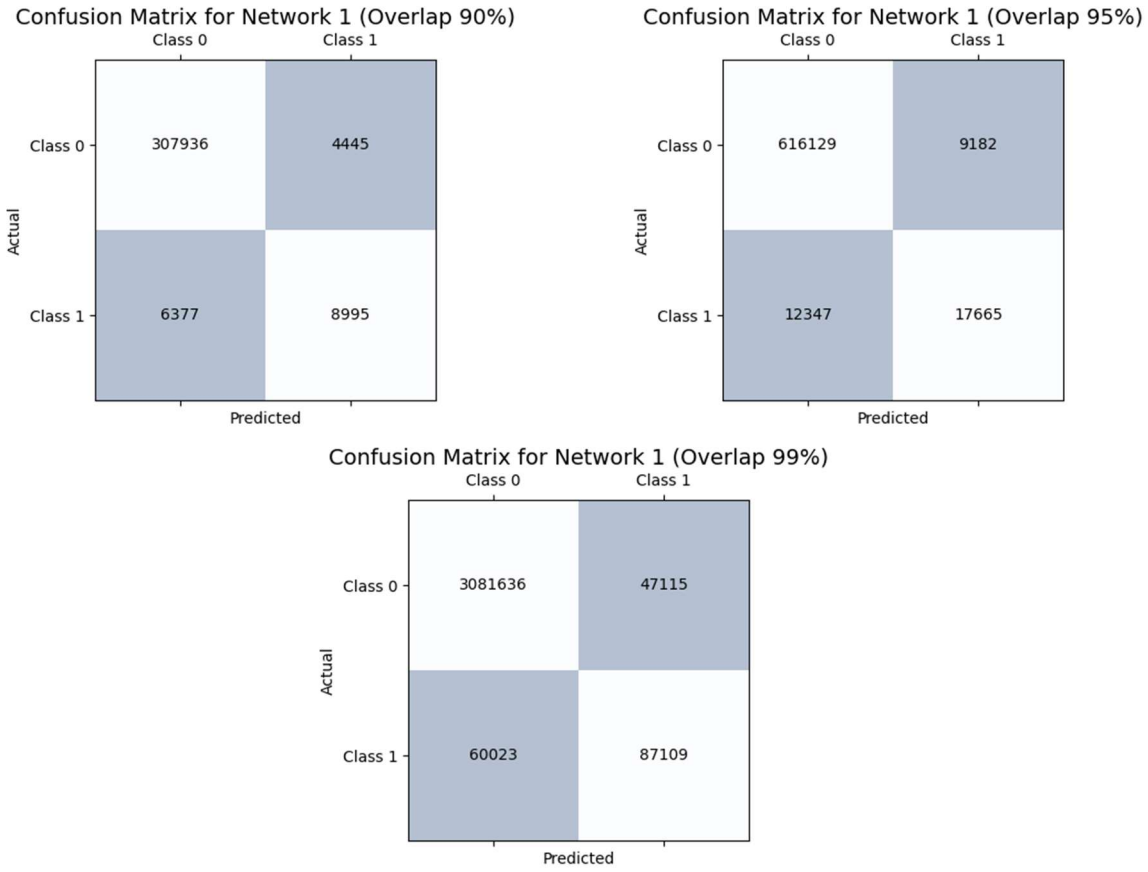
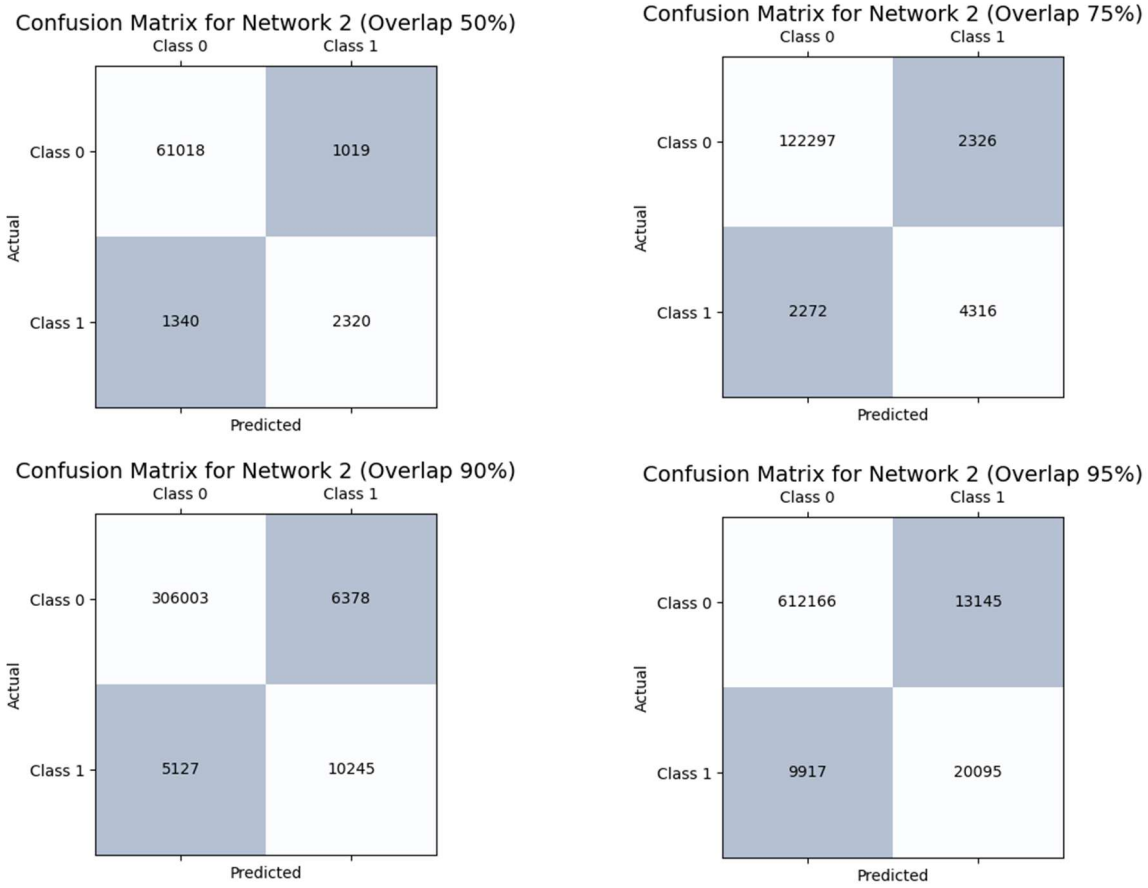


Figure 3.11. Confusion matrix for each overlap percentage (50%, 75%, 90%, 95%, and 99%) for Network 1.



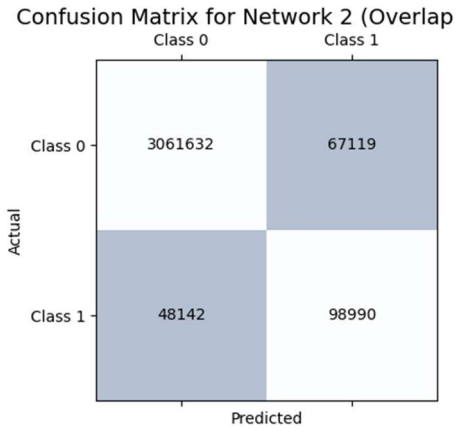


Figure 3.12. Confusion matrix for each overlap percentage (50%, 75%, 90%, 95%, and 99%) for Network 2.

From this matrix, the metrics outlined in Section 2.2.3 were calculated. The balanced accuracy trends across different overlap percentages for the two networks are shown in *Figure 3.13*.

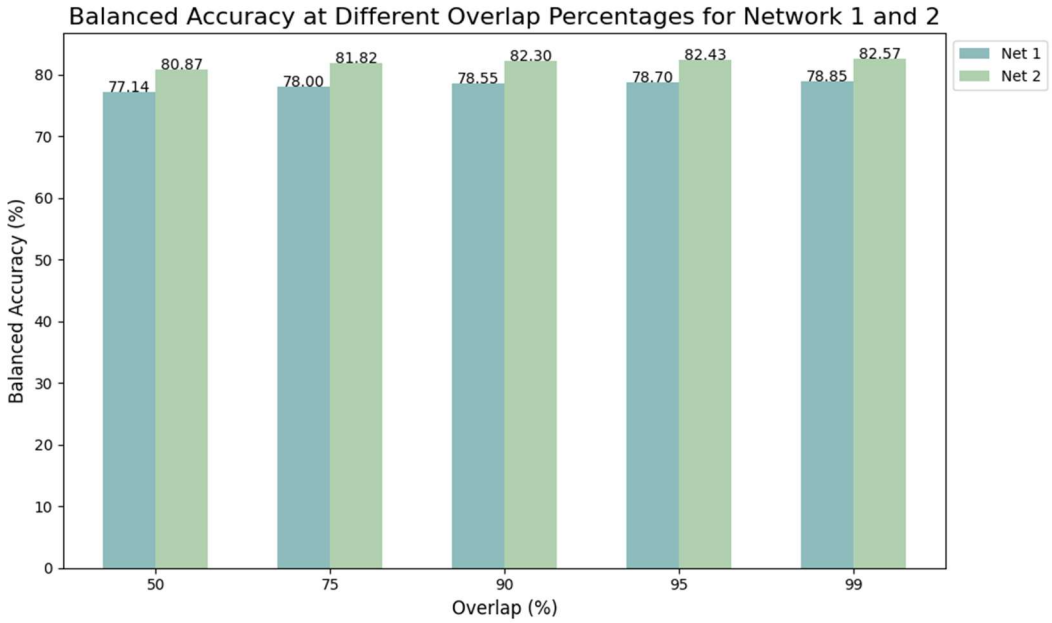


Figure 3.13. Bar chart of the percentage values of balanced accuracy across different overlap percentages, comparing Network 1 and Network 2.

The balanced accuracy values remain stable across different overlap percentages, with Network 1 around 78% and Network 2 around 82%. As shown in *Equation (2)*, the balanced accuracy depends on the network's ability to correctly detect both positive (abrupt) and negative (normal) windows. Network 2 is therefore slightly more accurate in identifying these windows, while varying the overlap percentage does not significantly affect the network's performance in correctly detecting the two classes.

In this study, the primary objective is the detection of abrupt windows. Therefore, the number of true positive should be high, while the false negatives, representing abrupt windows incorrectly identified as normal, should be kept to a minimum. To assess this, the precision, recall and F1-score for the positive class are analysed. These trends are shown in *Figures 3.14, 3.15, and 3.16*, respectively.

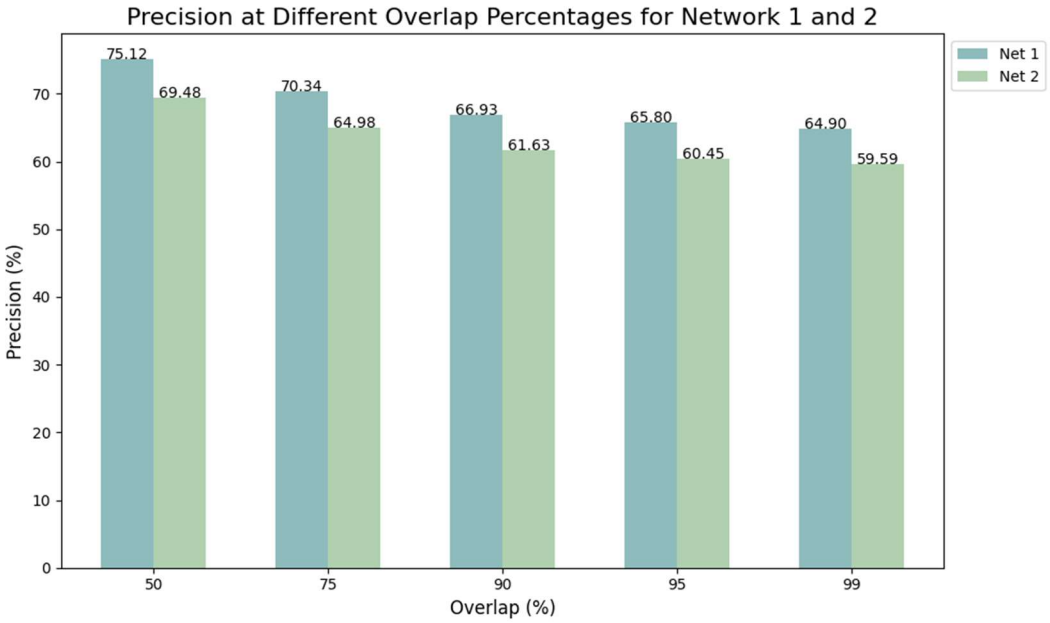


Figure 3.14. Bar chart of the percentage values of precision across different overlap percentages, comparing Network 1 and Network 2.

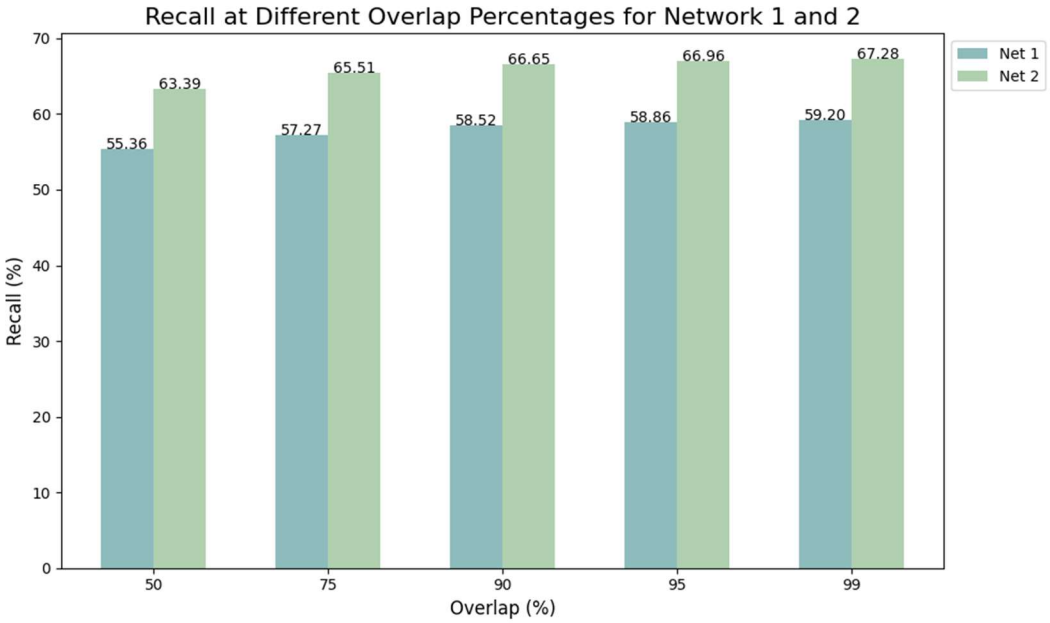


Figure 3.15. Bar chart of the percentage values of recall across different overlap percentages, comparing Network 1 and Network 2.

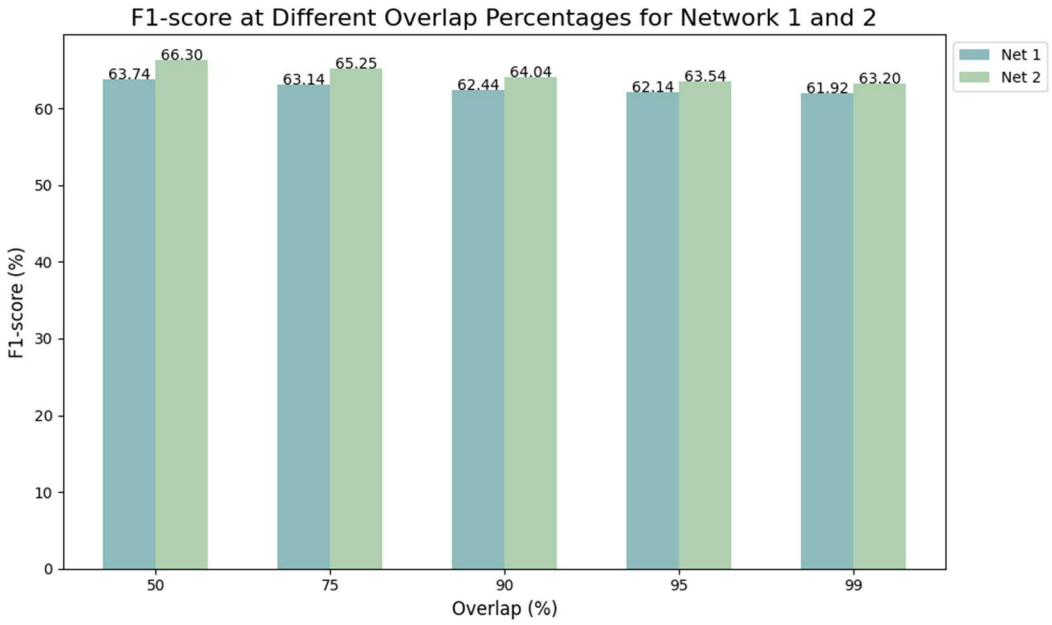


Figure 3.16. Bar chart of the percentage values of F1-score across different overlap percentages, comparing Network 1 and Network 2.

The performance results do not fully meet expectations. Precision values are consistently below 70%, with the only exceptions being the 50% and 75% overlap values for Network 1. This suggests a high number of false positives, meaning that normal windows are misclassified as abrupt. Recall values show a similar trend, with recall percentages for Network 1 specifically falling below 60%, indicating a high occurrence of false negatives. This trend is also reflected in the F1-score, which, as the harmonic mean of precision and recall, emphasises the overall performance. For Network 1, F1-scores are around 62%, while Network 2 shows slightly higher values, around 64%, highlighting no significant performance difference between the two networks in detecting abrupt movements. Additionally, as with balanced accuracy, there are no notable performance differences across the different overlap percentages.

The low precision and recall values may not necessarily reflect the network's ability to detect abrupt movements but rather indicate how many windows associated with such movements are classified as positive. Observing acceleration graphs comparing actual and predicted windows (see *Figures 3.6 to 3.10*) reveals that, within a single movement, the network may identify more or fewer windows as abrupt than expected. This discrepancy impacts the metrics, which may not fully represent the network's performance. It is

possible that the movement is correctly recognised, but the predicted number of windows associated with it may not align with the expected count.

For normal window detection, specificity values can be evaluated. *Figure 3.17* shows specificity values for both networks across different overlap percentages. This metric depends solely on the negative class, considering both correctly classified instances (True Negatives) and misclassifications (False Positives). The graph reveals that both networks perform exceptionally well in identifying the negative class, with values around 98% and no substantial differences across different overlap percentages.

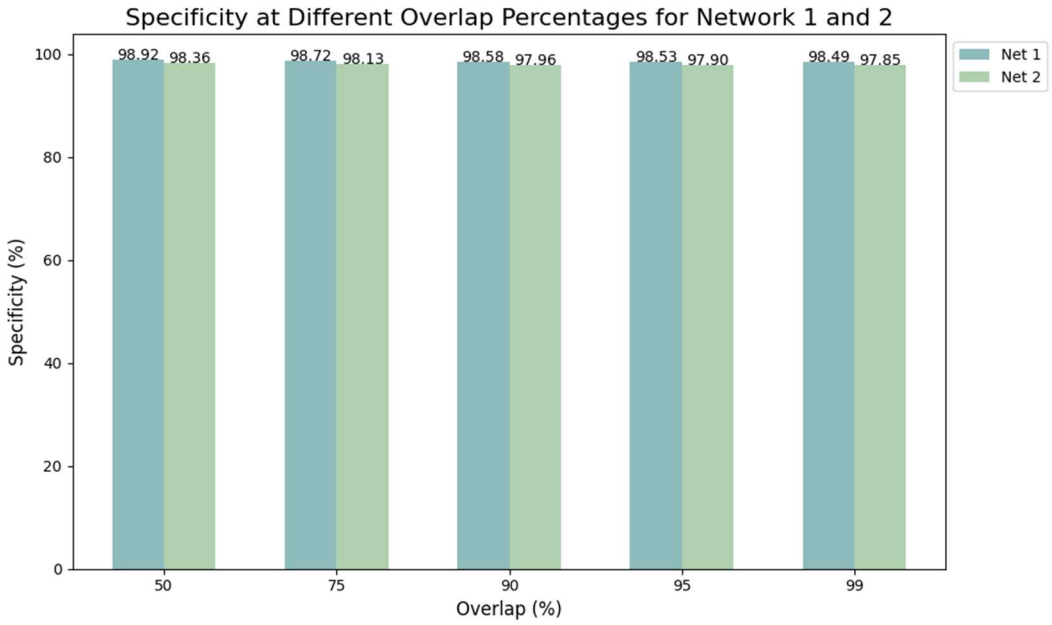


Figure 3.17. Bar chart of the percentage values of specificity across different overlap percentages, comparing Network 1 and Network 2.

This strong performance is further supported by the Macro F1-score, which balances both classes equally. Macro F1-score values are displayed in *Figure 3.18*. Notably, these values (ranging between 80% and 82% for both networks) are higher than the F1-score calculated for only the positive class, as they reflect the overall performance of the network. Moreover, the overlap percentages does not affect the performance of both networks.

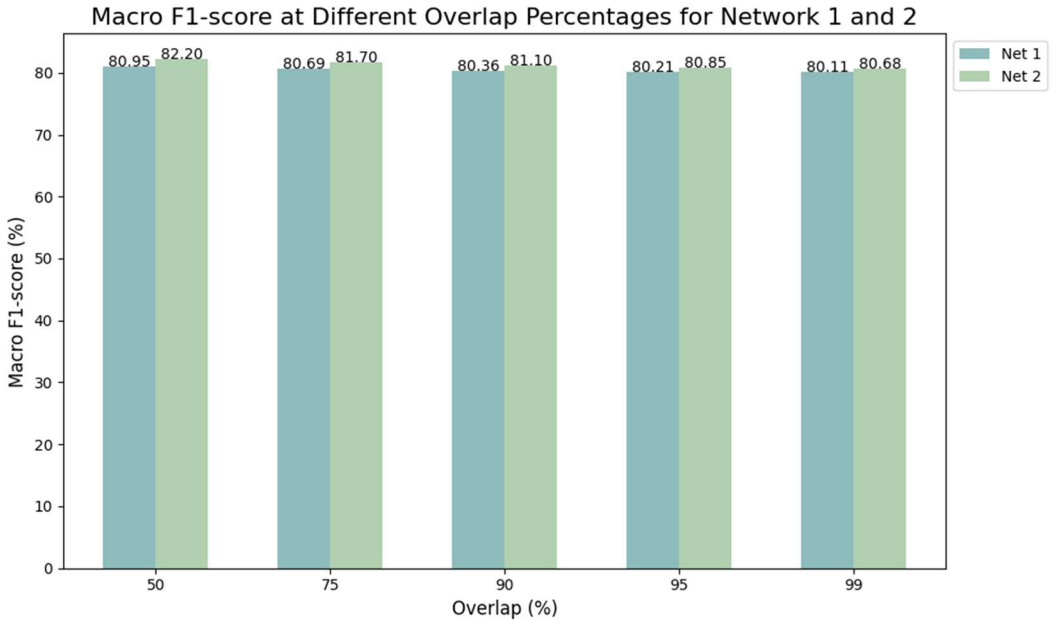


Figure 3.18. Bar chart of the percentage values of Macro F1-score across different overlap percentages, comparing Network 1 and Network 2.

The goal of this study is to assess whether the network can reliably identify abrupt movements. Therefore, in the performance analysis, the focus should be on recognizing movements rather than individual windows. Each trial includes 30 movements, 4 of which are abrupt. Each subject completed three trials, resulting in a total of 5490 movements across the 61 subjects, of which 732 are abrupt. In a 3-second movement, it was verified whether at least one abrupt window was detected. Based on a comparison between the movements performed by the subjects and the network's predictions, a confusion matrix was constructed. The confusion matrices for all overlap percentages for both networks are shown in *Figure 3.19 and 3.20*. Both networks present low false negative rates, always less than 100, demonstrating effective detection of abrupt movements. However, Network 2 has a significantly higher number of false positives. Although minimising false negatives is the primary objective, a high false positive rate can still be a problem in industrial

applications, as it would unnecessarily activate the safety system, slowing operations and reducing efficiency.

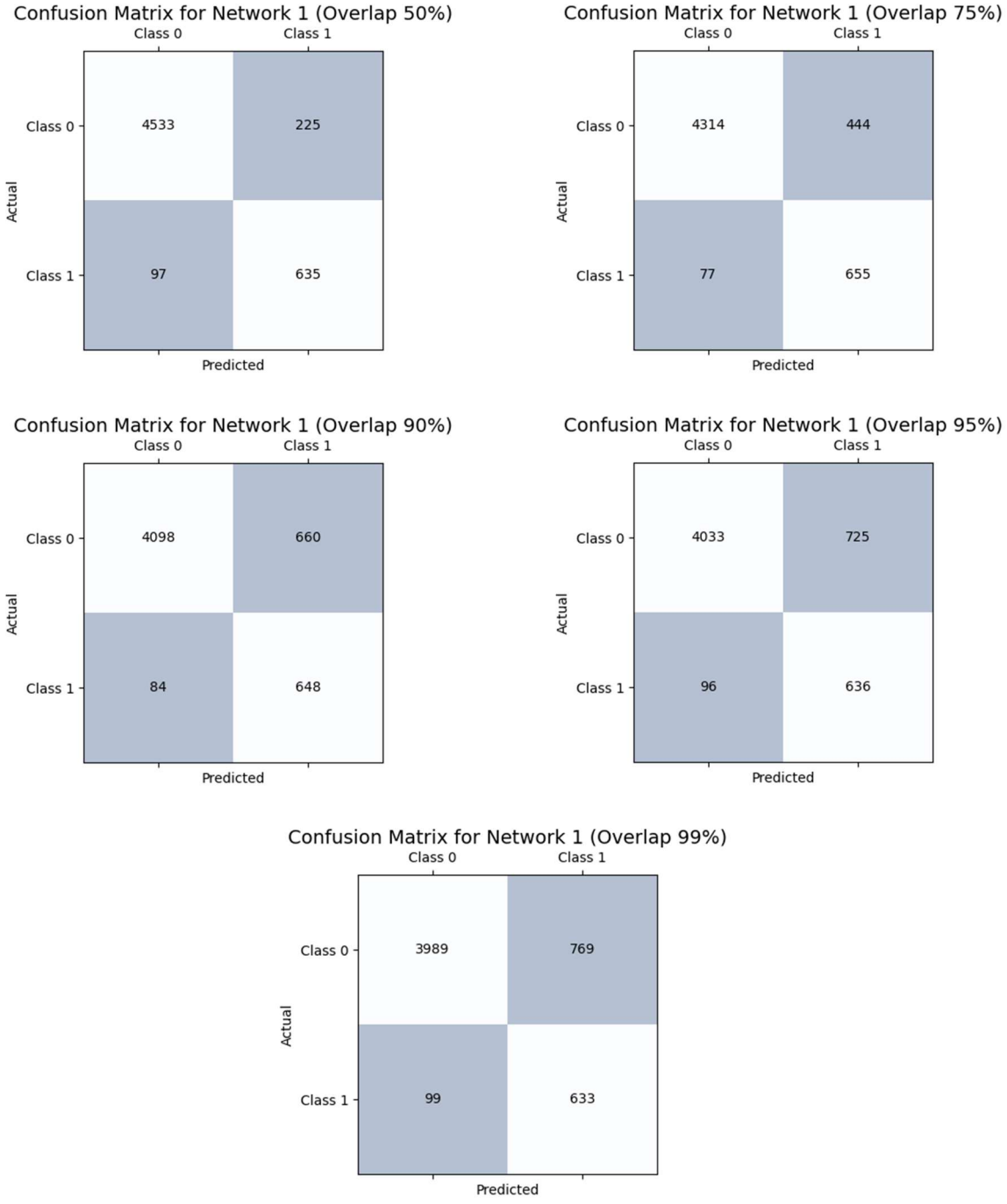


Figure 3.19. Confusion matrix for each overlap percentage (50%, 75%, 90%, 95%, and 99%) for Network 1.

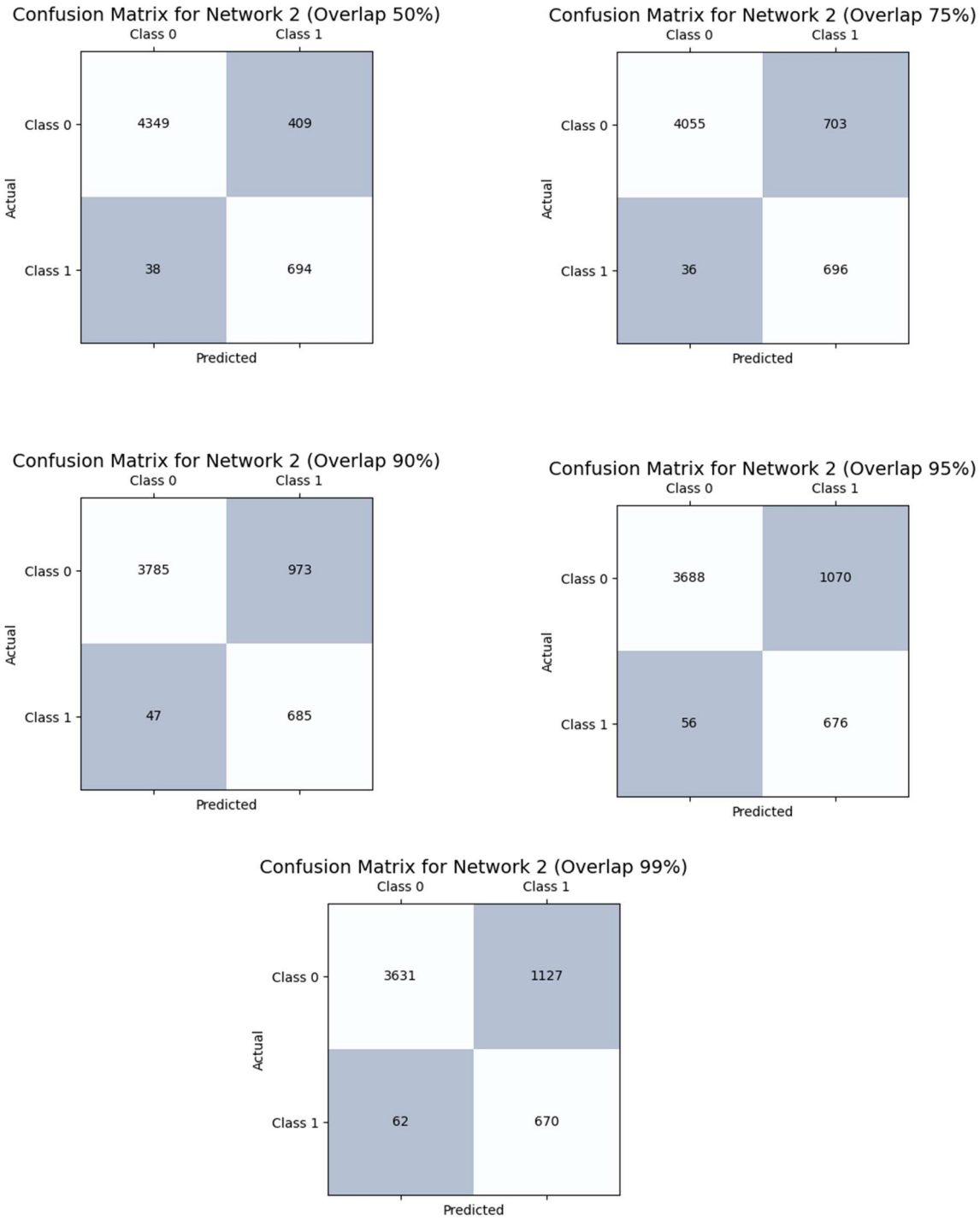


Figure 3.20. Confusion matrix for each overlap percentage (50%, 75%, 90%, 95%, and 99%) for Network 2.

From these data, the metrics described in Section 2.2.3 were calculated. *Figure 3.21* displays the balanced accuracy values as a function of the varying overlap percentages.

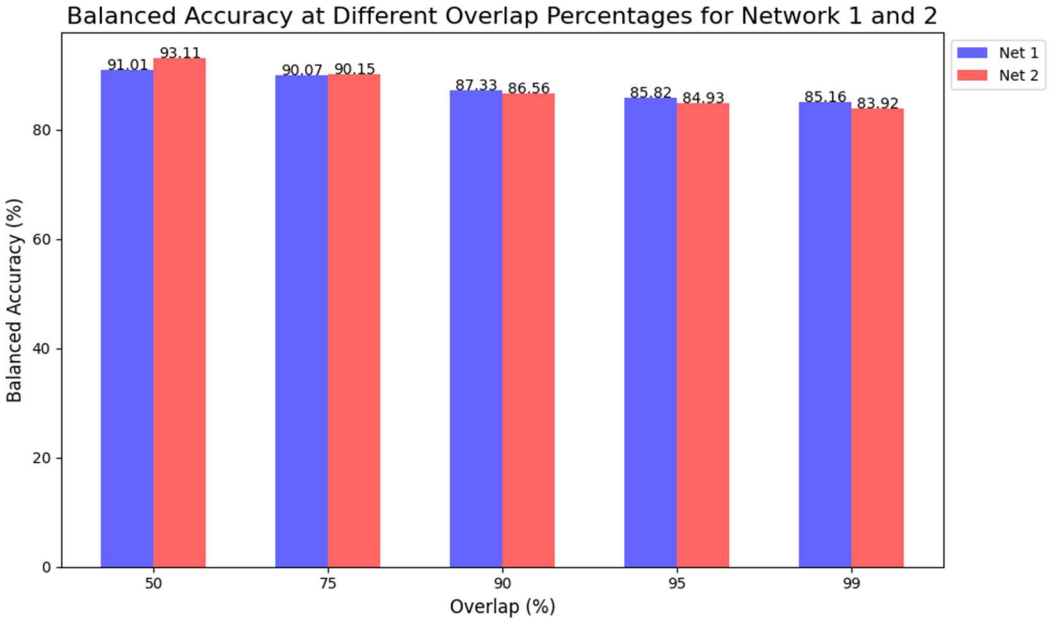


Figure 3.21. Bar chart of the percentage values of balanced accuracy across different overlap percentages, comparing Network 1 and Network 2.

The balanced accuracy values for 3-second movements confirm the strong overall performance of both networks. Specifically, values exceed 85% for Network 1 and 83% for Network 2. Except at overlap values of 50% and 75%, Network 1 demonstrates slightly higher accuracy than Network 2, in contrast to results observed when analysing individual windows. Overall, as before, no significant differences are observed across varying overlap percentages.

To further assess the networks' ability to identify abrupt movements, the trends of precision, recall, and F1-score for the positive class, shown in *Figure 3.22, 3.23, and 3.24* respectively, are analysed.

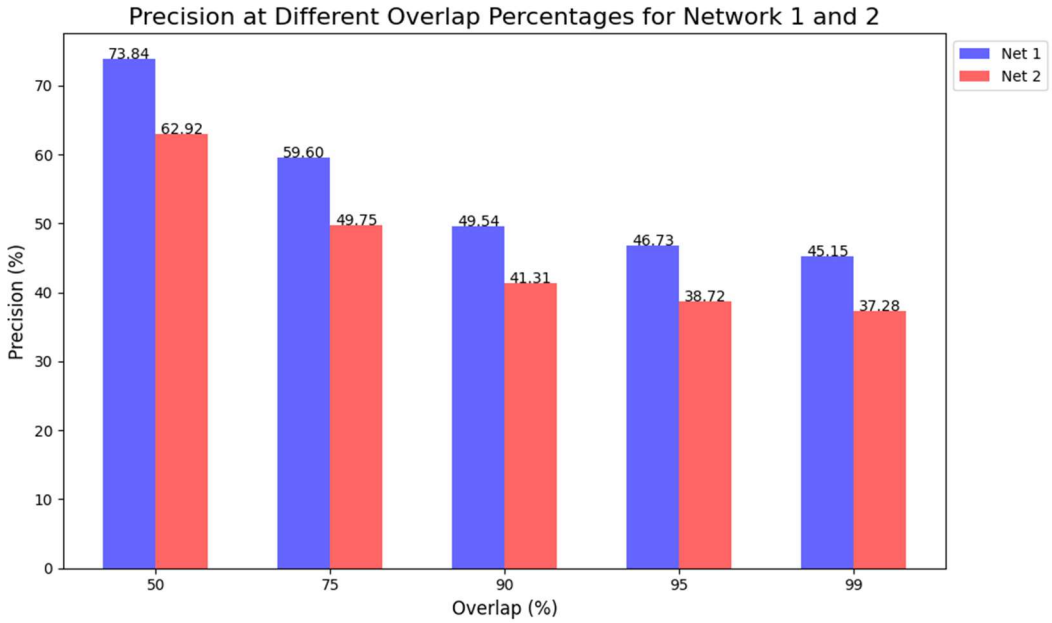


Figure 3.22. Bar chart of the percentage values of precision across different overlap percentages, comparing Network 1 and Network 2.

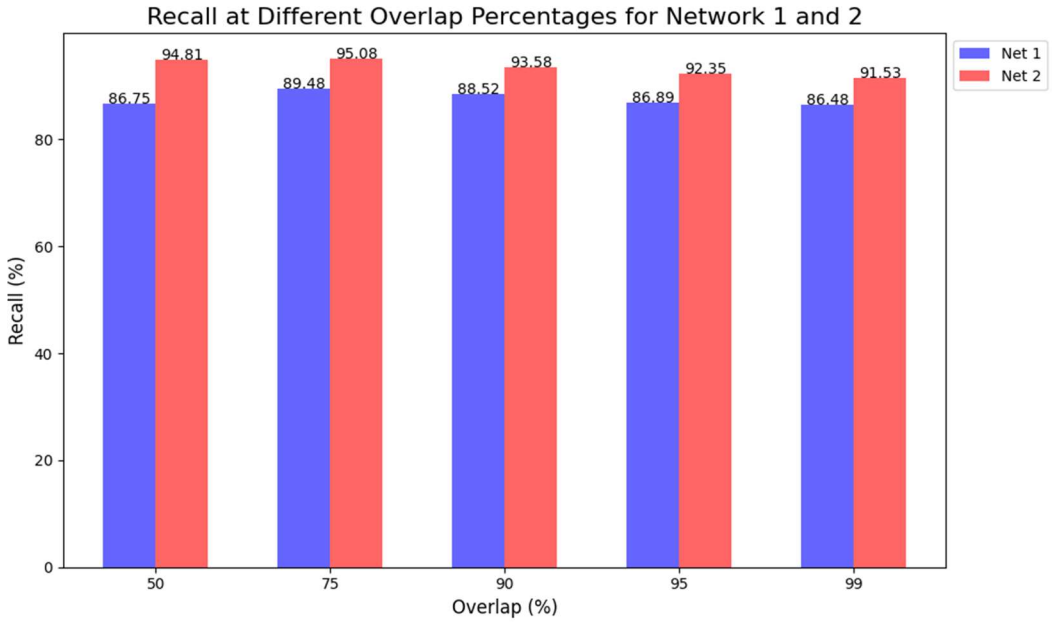


Figure 3.23. Bar chart of the percentage values of recall across different overlap percentages, comparing Network 1 and Network 2.

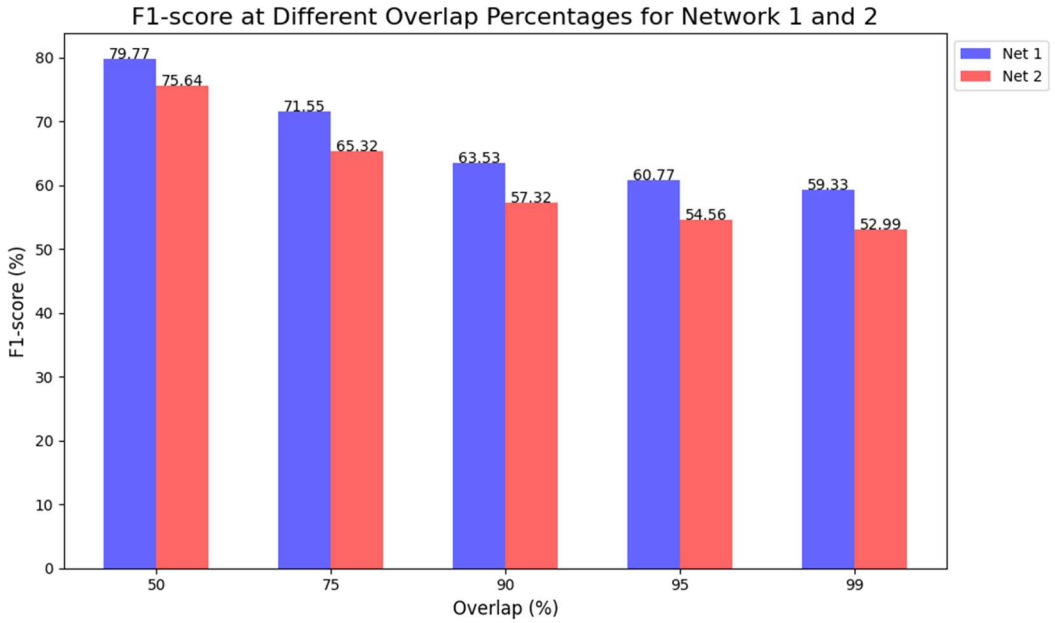


Figure 3.24. Bar chart of the percentage values of F1-score across different overlap percentages, comparing Network 1 and Network 2.

In this situation, precision and recall show completely different trends. Precision values are quite low for both networks. Specifically, for Network 1, values range from 45% to 60%, except at 50% overlap, where precision reaches 74%, making it the highest percentage. For Network 2, precision values vary between 37% and 63%. Comparing the two networks, it is clear that Network 2 has significantly lower precision percentages than Network 1. Low precision values indicate a high number of false positive, which, as observed also in the confusion matrices, are notably frequent in Network 2. Moreover, this time there is a difference with varying overlap: precision values decrease as overlap increase.

On the other hand, recall performance is very high for both networks. This means that false negatives remain consistently low, a highly favourable outcome for the objectives of our study. For both Network 1 and Network 2, recall percentages remain steady across overlap percentages, above 85% and 90% respectively.

Finally, F1-score values reflect the trends observed in precision and recall. Overall, Network 1 demonstrates stronger performance compared to Network 2 across all overlap percentages, with an average difference of 6% between the two networks.

For the negative class (standard movements), the specificity values shown in *Figure 3.25* confirm the networks' ability to accurately identify these movements. Network 1 demonstrates higher specificity values, all exceeding 83%, in comparison to Network 2, where specificity generally does not surpass 80% except for the 50% and 75% overlap cases.

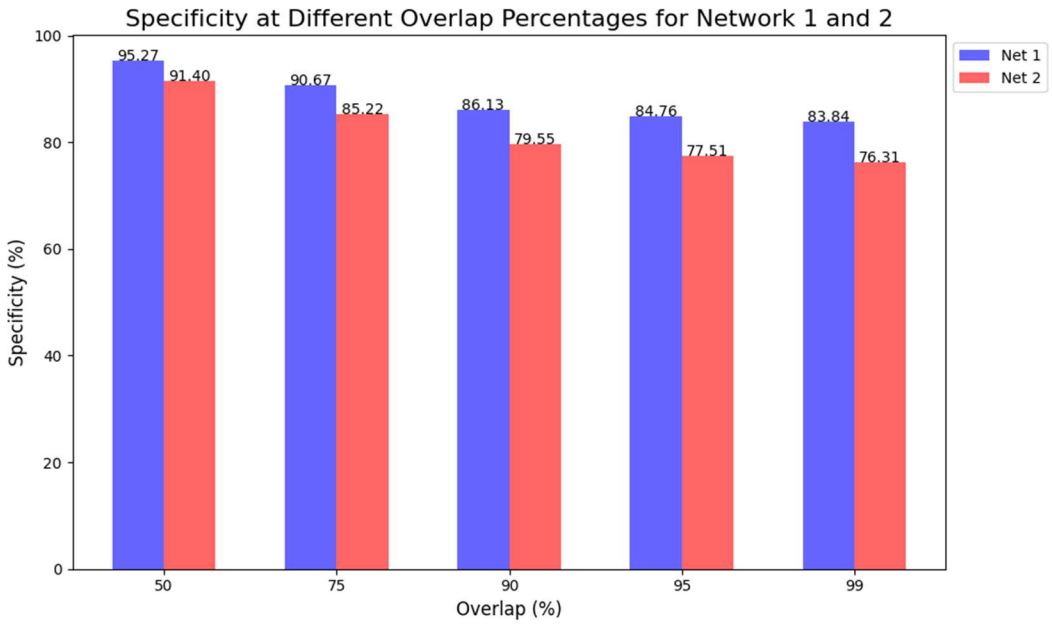


Figure 3.25. Bar chart of the percentage values of specificity across different overlap percentages, comparing Network 1 and Network 2.

Finally, the Macro F1-score, shown in *Figure 3.26*, was also analysed to provide an overview of the overall performance of both networks. Here again, Network 1 demonstrates better performance compared to Network 2 across all overlap percentages, with an average difference of 5% for each overlap case.

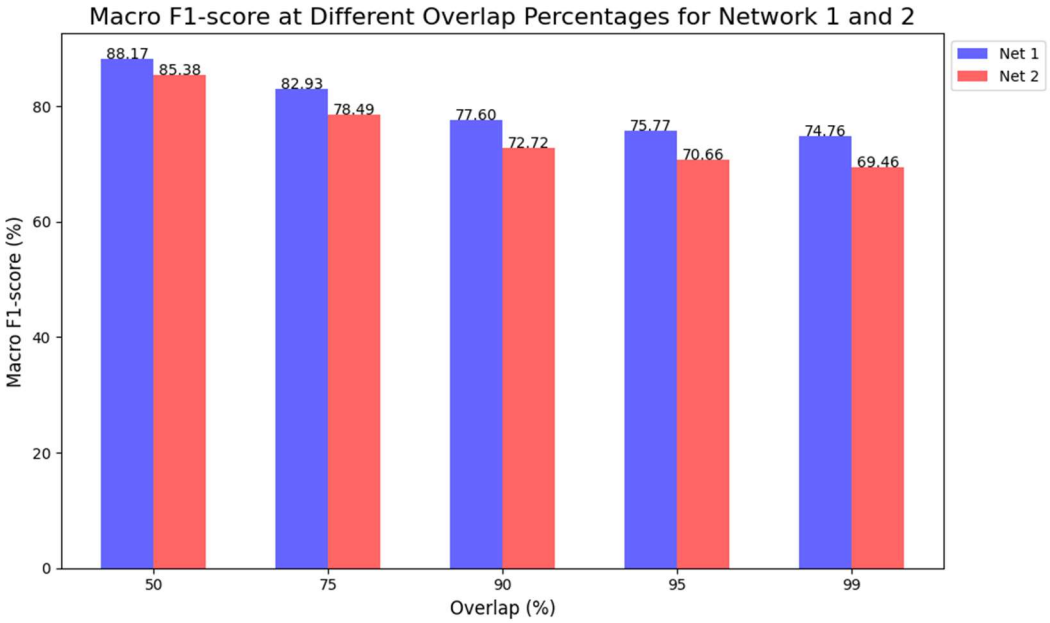


Figure 3.26. Bar chart of the percentage values of Macro F1-score across different overlap percentages, comparing Network 1 and Network 2.

In general, analysing the performance of both networks in terms of window recognition as well as 3-second movement recognition reveals several insights. Regarding window recognition, while both networks display relatively similar metrics, Network 2 shows slightly higher balanced accuracy, F1-score for positive class and Macro F1-score values than Network 1. This indicates that Network 2 is moderately more precise in recognizing individual windows, regardless of the overlap percentage.

However, as previously mentioned, our main interest lies in recognizing the overall movement rather than a single window. Looking at movement-level metrics, Network 1 generally exhibits better performance overall. Specifically, it achieves higher values in balanced accuracy, specificity, and macro F1-score compared to Network 2. Focusing on positive class recognition, Network 2 has a high number of false positives, which significantly lowers its precision. In terms of recall, Network 2 shows slightly higher values, but those of Network 1 are still highly acceptable, ensuring a low number of false negatives.

Based on these considerations, it can be stated that Network 1 demonstrates superior overall performance. Consequently, it has been selected for the next phase of the study, which involves a real-time movement recognition.

3.1.4 Time analysis

In real-time applications, timing analysis is crucial. For this reason, the time required by the two networks to classify the movements of all 61 subjects was evaluated, using the function described in Section 2.2.3. *Tables 3.2 and 3.3* present the average inference time per subject, representing the time needed by the network to classify the data for a single subject, as well as the total inference time, indicating the time taken to classify all 61 subjects, for Network 1 and Network 2 respectively.

Table 3.2. Average inference time across all subjects and total inference time (in seconds) for each overlap percentage (50%, 75%, 90%, 95%, and 99%) for Network 1.

Overlap (%)	**50**	**75**	**90**	**95**	**99**
Average Inference Time across all subjects (seconds)	1.41	2.39	4.52	9.23	63.54
Total Inference Time (seconds)	85.76	145.93	275.65	562.88	3875.87

Table 3.3. Average inference time across all subjects and total inference time (in seconds) for each overlap percentage (50%, 75%, 90%, 95%, and 99%) for Network 2.

Overlap (%)	**50**	**75**	**90**	**95**	**99**
Average Inference Time across all subjects (seconds)	1.34	2.14	4.65	9.43	65.34
Total Inference Time (seconds)	81.95	130.67	283.76	574.95	3985.81

The obtained values show a non-linear trend, which is also evident when looking at the graphs of the trends at varying overlap in *Figures 3.27 and 3.28*.

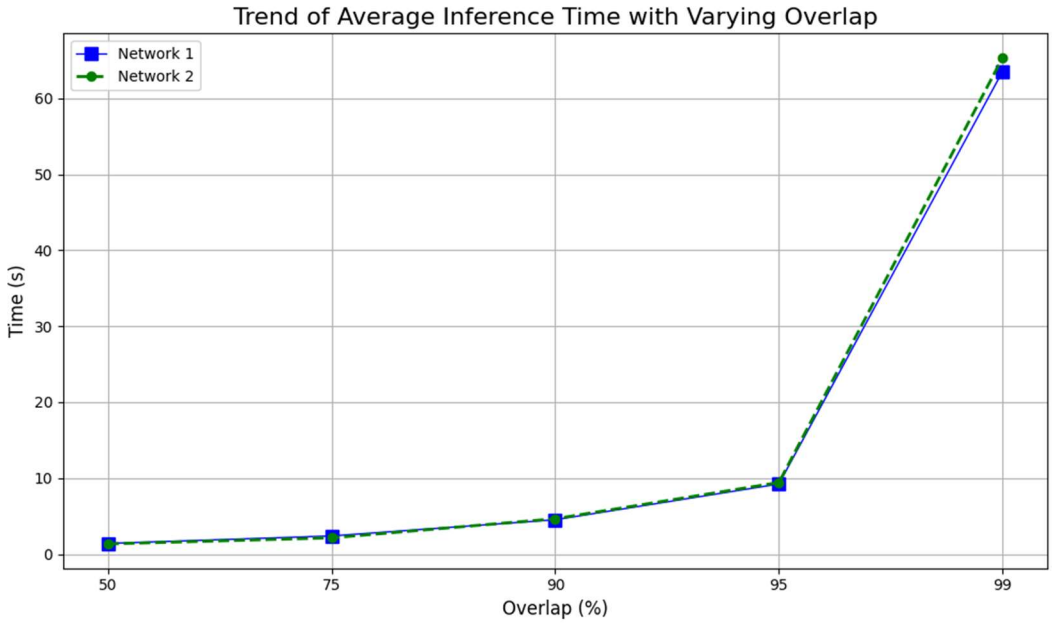


Figure 3.27. Line graph showing the variation of the average inference time across different overlap percentages (50%, 75%, 90%, 95%, and 99%) for both Network 1 and Network 2.

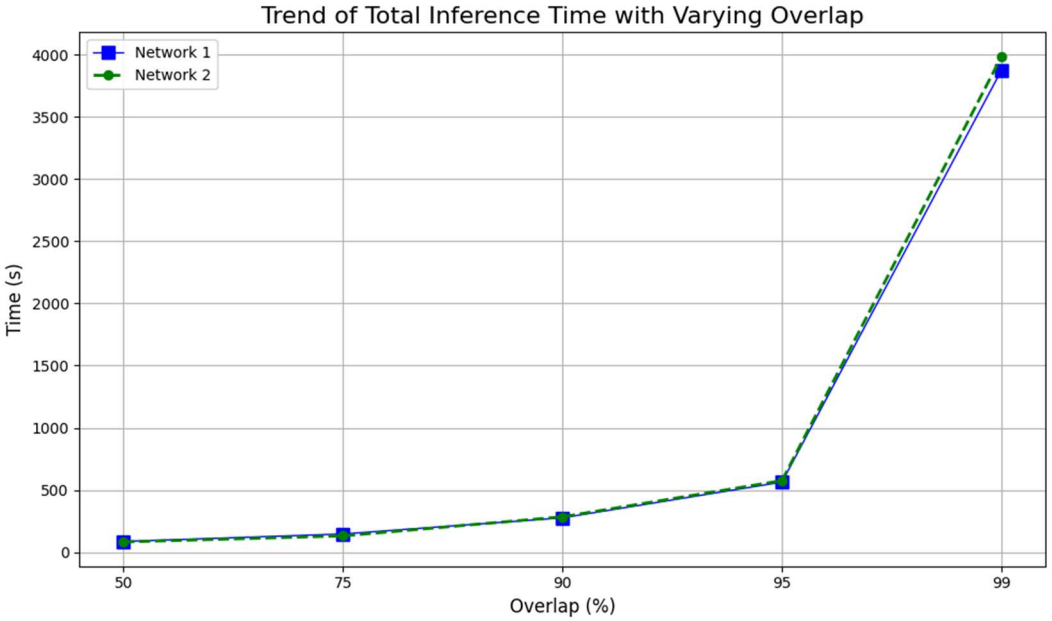


Figure 3.28. Line graph showing the variation of the total inference time across different overlap percentages (50%, 75%, 90%, 95%, and 99%) for both Network 1 and Network 2.

Up to 95% overlap, the average inference times for both networks remain low, below 10 seconds. However, when the overlap reaches 99%, inference times increase exponentially, exceeding one minute. The total inference time shows a similar pattern, with an

exponential rise between 95% and 99% overlap, reaching up to six times the previous value.

Both models are generally efficient, but as the overlap increases so does the number of windows to be analysed, leading to longer inference times. Specifically, with higher overlaps rates, the number of windows per subject nearly doubles compared to the previous overlap percentage (see *Table 1*). A particularly significant case is the 99% overlap, where the network must analyse 53703 windows per subject (see *Table 1*), more than 5 times the number of windows to be analysed for the 95% overlap. This explains the nonlinear progression of both average and total inference times.

Regarding the difference between the two networks, Network 2 is slightly faster than Network 1 for 50% and 75% overlap. However, for higher overlap values, the situation reverses. Moreover, up until 95% overlap, the inference times for both networks are comparable, with minimal differences. Once 99% overlap is reached, the difference becomes more evident, with Network 1 exhibiting both lower average and total inference times compared to Network 2.

For a more accurate analysis of the network's performance in terms of time, it is useful to examine the time it takes to analyse a single window. By knowing the average time and dividing it by the number of windows per subject, an estimate can be obtained. The obtained values are presented in *Table 3.4*.

Table 3.4. Average time required to analyse a single window (in milliseconds) for each overlap percentage (50%, 75%, 90%, 95%, and 99%) for both Network 1 and Network 2.

Overlap (%)	**50**	**75**	**90**	**95**	**99**
Network 1	1.30 ms	1.11 ms	0.84 ms	0.86 ms	1.18 ms
Network 2	1.24 ms	0.99 ms	0.87 ms	0.88 ms	1.22 ms

It is evident that both networks perform very well, with inference times for a single window on the order of a millisecond. Additionally, the differences between the two networks are confirmed: Network 2 is faster at 50% and 75% overlap, while Network 1 outperforms Network 2 at 90%, 95%, and 99% overlap.

The time analysis confirms the selection of Network 1 for the next phase of the study, as it is faster in conditions approaching the real-time.

3.2 Real-Time Recognition of Abrupt Movements

3.2.1 Performance analysis

This phase of the study involves the real-time recognition of abrupt movements as the subject performs various movements, including abrupt ones. Five new participants completed the test outlined in Section 2.2.2. The task required each subject to perform 30 movements of 3 seconds each across three configurations (Trial FR_r, Trial FR_l, and Trial LA_l). In total, the network analysed 450 movements, including 60 abrupt movements. It is important to note the continued class imbalance, with the positive class (abrupt movements) being the minority.

After comparing the network predictions with LED activations, which indicate the movements executed by the subject, a confusion matrix was constructed and displayed in *Figure 3.29*. It is evident that the network performs well with few errors. Out of 60 abrupt movements, only 9 are misclassified as normal, while the false positives amount to just 12.

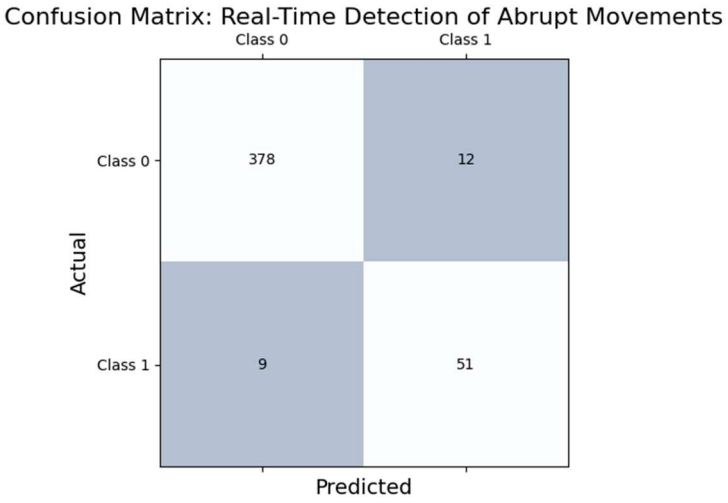


Figure 3.29. Confusion matrix showing the classification results of the real-time system for detecting abrupt movements.

For performance metrics, the same set of metrics as in previous analyses were calculated. *Table 3.5* presents the values for all six metrics.

Table 3.5. Performance metrics for the classification of abrupt movements, including balanced accuracy, precision (positive), recall (positive), F1-score (positive), macro F1-score, and specificity.

Balanced accuracy	Precision (positive)	Recall (positive)	F1-score (positive)	Macro F1-score	Specificity
90.96	80.95	85.0	82.93	90.11	96.92

The network demonstrates a high level of balanced accuracy at 90.96%, indicating its effectiveness in correctly classifying both abrupt and normal movements. In particular, the specificity value highlights the network's strong ability to correctly identify the negative class, reaching 96.92%. For the positive class, the network also shows strong performance, with precision and recall values of 80.95% and 85%, respectively. While the network exhibits a slightly higher rate of false positives than false negatives, this does not impact the primary objective of our study. The F1-score for the positive class further confirms the network's strong performance, reaching nearly 83%. Overall, the Macro F1-score slightly exceeds 90%, underscoring the network's robustness in recognizing both classes and ensuring a minimal error rate.

The network thus confirms its ability to effectively recognize abrupt movements even under real-time conditions, with all performance metrics considerably exceeding 80%.

3.2.2 Time analysis

In the context of human-robot collaboration safety, the activation timing of safety systems is crucial. For this reason, in real-time conditions, a timing analysis is essential. As outlined in Section 2.2.4, the mean and standard deviation for the three steps required to achieve classification were calculated (*Table 3.6*).

Table 3.6. Streaming time (s), preprocessing time (ms), and inference time (ms) for each subject and across all subjects. Values represent mean ± standard deviation.

Subjects	01	02	03	04	05	Inter-subject
Streaming time (s)	3.12 ± 0.05	3.12 ± 0.13	3.11 ± 0.11	3.11 ± 0.05	3.12 ± 0.18	**3.12 ± 0.11**
Preprocessing time (ms)	8.90 ± 1.87	9.12 ± 2.14	9.13 ± 2.45	8.83 ± 2.26	9.04 ± 2.49	**9.00 ± 2.26**
Inference time (ms)	263.8 ± 65.2	265.1 ± 95.6	251.3 ± 67.9	253.7 ± 70.6	264.7 ± 70.2	**259.7 ± 74.9**

Once the movement begins, the data is typically ready for the analysis after an average of 3.13 seconds, which includes both streaming and preprocessing times. Finally, the network takes approximately 260 milliseconds to analyse the movement.

In terms of efficiency, the network performs exceptionally well, achieving recognition few milliseconds after the movement's completion. The network's analysis operates in parallel with data streaming, storage, and preprocessing. As a result, the time required for recognition does not interfere with the data flow from the sensors. Consequently, the temporal performance of the network is highly acceptable, ensuring rapid movement recognition.

However, it is noteworthy that the time required to obtain the data (3.13 s) slightly exceeds the duration of a single movement, which is precisely 3 seconds. This delay could result in the network lagging in recognizing the movement or potentially missing critical samples necessary for accurate classification. In this case, the network demonstrates extremely high performance, so this time lag does not significantly compromise its capabilities, although it may still lead to occasional errors.

To determine whether these errors are caused by the real-time system rather than the network itself, an analysis of error distribution across the sequence of movements was conducted. *Table 3.7* summarises the 15 trials (three per each of the five subjects), including the abrupt movements correctly recognized, not detected, and normal movements misclassified as abrupt. Observing the error distribution reveals that the system's efficiency is excellent at the beginning of the test, with few errors. However, toward the end of the test, there is a noticeable increase in the number of errors.

When dividing the movements into three intervals (1–10, 11–20, 21–30), it becomes evident that errors increase significantly in the last interval, as shown in *Figure 3.30*. The difference in error distribution across these intervals is statistically significant. Specifically, there is a statistically significant difference between the first and third intervals, as well as the second and third intervals, with a p-value < 0.001 for both.

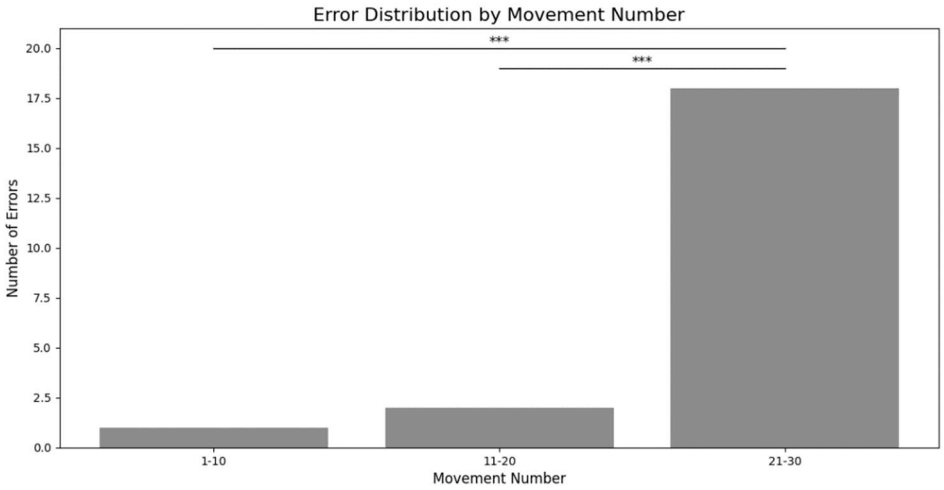


*Figure 3.30. Comparison of error distribution across three different movement intervals: 1 – 10, 11 – 20, and 21 – 30. Statistical significance was determined using a chi-square test. ***p<0.001*

The combined streaming and preprocessing time exceed the movement duration by few milliseconds. Over time, these extra milliseconds may lead to a misalignment between the system and the movements dictated by the LEDs, making the system more prone to errors as the test progresses.

Table 3.7. For each trial performed by all five subjects, abrupt movements correctly classified are highlighted in green, missed (not detected) abrupt movements are highlighted in red, and normal movements incorrectly classified as abrupt are highlighted in orange.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
01_FR_r																														
01_FR_l																														
01_LA_l																														
02_FR_r																														
02_FR_l																														
02_LA_l																														
03_FR_r																														
03_FR_l																														
03_LA_l																														
04_FR_r																														
04_FR_l																														
04_LA_l																														
05_FR_r																														
05_FR_l																														
05_LA_l																														

LEGEND:

- Correctly detected abrupt movements.
- Missed (not detected) abrupt movements.
- Normal movements incorrectly detected as abrupt

4. CONCLUSIONS AND FUTURE WORK

This experimental thesis demonstrates the effectiveness of an LSTM network trained with wrist acceleration data acquired via MIMUs in recognizing abrupt movements in conditions approaching the real-time. The aim of the study is to identify the abrupt movements performed by an operator in industry, guaranteeing both efficiency and safety for a scenario of human-robot collaboration. Indeed, ensuring an accurate real-time movement detection enables the development of effective safety systems, enhancing collaboration and making the workplace a safer environment for workers.

Initially, two networks were considered: one trained solely with acceleration data (Network 1) and the other trained with both acceleration and angular velocity data (Network 2). Their performance was compared using a sliding windows approach for signal segmentation. In terms of metrics and processing times, Network 1 showed slightly better performance, making it the preferred choice for the second phase of the study.

Subsequently, the LSTM network was integrated into a real-time system for abrupt movement recognition. A pick-and-place tack was performed by five participants across three different configurations. Results demonstrated the network's ability to recognize abrupt movements with high accuracy (balanced accuracy, macro F1-score, and specificity > 90%) within a few hundred milliseconds (259.7 ± 74.9 ms).

Despite some limitations, such as data streaming and preprocessing times that may slow the system and increase the likelihood of errors, these findings highlight the network's capability to recognise abrupt movements moving towards real-time conditions.

Future studies could focus on improving the real-time signal acquisition system to reduce streaming delays. Additionally, the current system processes signals only after the entire movement sequence is collected. A more advanced approach would involve a system capable of analysing incoming data in real-time, enabling the network to start processing partial movement data even before the movement is completed.

References

Aaltonen, I., Salmi, T., & Marstio, I. (2018). Refining levels of collaboration to support the design and evaluation of human-robot interaction in the manufacturing industry. *Procedia CIRP*, *72*, 93–98. https://doi.org/10.1016/J.PROCIR.2018.03.214

Amaral, P., Silva, F., & Santos, V. (2023). Recognition of Grasping Patterns Using Deep Learning for Human-Robot Collaboration. *Sensors (Basel, Switzerland)*, *23*(21). https://doi.org/10.3390/s23218989

Borboni, A., Reddy, K. V. V., Elamvazuthi, I., AL-Quraishi, M. S., Natarajan, E., & Azhar Ali, S. S. (2023). The Expanding Role of Artificial Intelligence in Collaborative Robots for Industrial Applications: A Systematic Review of Recent Works. *Machines*, *11*(1), 111. https://doi.org/10.3390/machines11010111

Buerkle, A., Eaton, W., Lohse, N., Bamber, T., & Ferreira, P. (2021). EEG based arm movement intention recognition towards enhanced safety in symbiotic Human-Robot Collaboration. *Robotics and Computer-Integrated Manufacturing*, *70*, 102137. https://doi.org/10.1016/J.RCIM.2021.102137

Digo, E., Polito, M., Caselli, E., Gastaldi, L., & Pastorelli. S. (2024). *Dataset of Abrupt and Standard Industrial Gestures (DASIG) [Data set].* Zenodo. https://doi.org/10.5281/zenodo.13927735

Digo, E., Polito, M., Pastorelli, S., & Gastaldi, L. (2024). Detection of upper limb abrupt gestures for human–machine interaction using deep learning techniques. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, *46*(4), 227. https://doi.org/10.1007/s40430-024-04746-9

Imanzadeh, S., Tanha, J., & Jalili, M. (2024). Ensemble of deep learning techniques to human activity recognition using smart phone signals. *Multimedia Tools and Applications*. https://doi.org/10.1007/s11042-024-18935-0

Keras 3 API documentation. (n.d.). Retrieved June 1, 2024, from https://keras.io/api/

Liu, H., Fang, T., Zhou, T., Wang, Y., & Wang, L. (2018). Deep Learning-based Multimodal Control Interface for Human-Robot Collaboration. *Procedia CIRP*, *72*, 3–8. https://doi.org/10.1016/J.PROCIR.2018.03.224

Ordóñez, F., & Roggen, D. (2016a). Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors*, *16*(1), 115. https://doi.org/10.3390/s16010115

Ordóñez, F., & Roggen, D. (2016b). Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors*, *16*(1), 115. https://doi.org/10.3390/s16010115

Polito, M., Digo, E., Pastorelli, S., & Gastaldi, L. (2023a). *Abrupt Movements Assessment of Human Arms Based on Recurrent Neural Networks for Interaction with Machines* (pp. 143–151). https://doi.org/10.1007/978-3-031-45705-0_15

Polito, M., Digo, E., Pastorelli, S., & Gastaldi, L. (2023b). *Deep Learning Technique to Identify Abrupt Movements in Human-Robot Collaboration* (pp. 73–80). https://doi.org/10.1007/978-3-031-32439-0_9

Precision Motion for Research. (n.d.). Retrieved October 27, 2024, from https://clario.com/solutions/precision-motion-for-research/

Rivera, P., Valarezo, E., Choi, M.-T., & Kim, T.-S. (2017). Recognition of Human Hand Activities Based on a Single Wrist IMU Using Recurrent Neural Networks. *International Journal of Pharma Medicine and Biological Sciences*, *6*(4), 114–118. https://doi.org/10.18178/ijpmbs.6.4.114-118

Rosenstrauch, M. J., & Kruger, J. (2017). Safe human-robot-collaboration-introduction and experiment using ISO/TS 15066. *2017 3rd International Conference on Control, Automation and Robotics (ICCAR)*, 740–744. https://doi.org/10.1109/ICCAR.2017.7942795

Vysocky, A., & Novak, P. (2016). HUMAN – ROBOT COLLABORATION IN INDUSTRY. *MM Science Journal*, *2016*(02), 903–906. https://doi.org/10.17973/MMSJ.2016_06_201611

Xiang, L., Gu, Y., Gao, Z., Yu, P., Shim, V., Wang, A., & Fernandez, J. (2024). Integrating an LSTM framework for predicting ankle joint biomechanics during gait using inertial sensors. *Computers in Biology and Medicine*, *170*, 108016. https://doi.org/10.1016/J.COMPBIOMED.2024.108016

Xu, X., Lu, Y., Vogel-Heuser, B., & Wang, L. (2021). Industry 4.0 and Industry 5.0—Inception, conception and perception. *Journal of Manufacturing Systems*, *61*, 530–535. https://doi.org/10.1016/J.JMSY.2021.10.006

Zafar, M. H., Langås, E. F., & Sanfilippo, F. (2024). Exploring the synergies between collaborative robotics, digital twins, augmentation, and industry 5.0 for smart manufacturing: A state-of-the-art review. *Robotics and Computer-Integrated Manufacturing*, *89*, 102769. https://doi.org/10.1016/J.RCIM.2024.102769