FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Human Partner Understanding: Recognition and Prediction of Human Action Sequences for Safe Human-Robot Collaboration

Sílvia Jorge Moreira da Rocha

Mestrado em Engenharia Informática e Computação

Supervisor: João Pedro Correia dos Reis

Second Supervisor: Vítor Hugo Machado Oliveira Pintor

July 30, 2022

Human Partner Understanding: Recognition and Prediction of Human Action Sequences for Safe Human-Robot Collaboration

Sílvia Jorge Moreira da Rocha

Mestrado em Engenharia Informática e Computação

July 30, 2022

Abstract

Within an Industry 4.0 context, Human-Robot Collaboration (HRC) is an emerging research field that seeks to develop methods to allow human operators to work alongside robots in a close manner. Although this idea sounds appealing, there are still a lot of technical constraints that need to be solved. In particular, existing safety solutions lack standards and solutions developed specifically for industrial applications. Additionally, industry applications require accurate estimation and classification of the human motion so that efficiency and safety of HRC can be guaranteed and human-robot coexistence can be fluent [38].

Existing systems require expensive equipment and high computational resources to fulfill this requirement, which is not economically viable for the companies. Another common problem with these solutions is that they are not developed robustly, ensuring that the system's accuracy remains the same regardless of the operator's size, pose and distance to the camera. Combined, these factors make developing a low-budget and lightweight but robust system for industrial applications beneficial.

This dissertation presents the development phases for an intelligent integrated framework for movement prediction and early recognition. This framework yields synergies from their mutual benefits by using feedback between the two models. The Recurrent Neural Networks used for prediction and early recognition were trained with 3D skeleton data from datasets and tested with real-time data to assure good performance in uncontrolled environments (with particular attention to bad lighting conditions frequently found in factory floors) and robustness regarding the operator's size and pose.

Additionally, the impact of using the output of the prediction model in early recognition and vice-versa is studied, demonstrating that the integrated framework presents more benefits when compared to the isolated methods that only focus on movement recognition or prediction.

Keywords: Robotics, Human-Robot Collaboration, Early Recognition, Movement Prediction, Computer Vision, Machine Learning, Deep Learning

Acknowledgements

I would like to start by thanking Professor João Reis and Vítor Pinto, my supervisors, for the guidance, ideas, and follow-up of the work done throughout this time.

The biggest thanks to Liliana Antão for giving me the privilege of working with her closely on this subject and always being available to help, motivate and support me at every difficult moment. Thank you for your sense of humor, patience, and understanding whenever needed. Without all of that, this dissertation would not have happened.

To all of the twenty-two people who were kind enough to help me build the dataset I needed, a big thanks for the patience and for agreeing to help me.

To my friends who patiently heard me complaining about the performance of my models for hours without understanding a word, I was saying. Thank you for all the times you helped me keep a social life, to have my so-called "mental breaks" and for being there. Without all of you, I would have gone crazy for sure.

An enormous thank you to my parents, sisters, and brothers (yes, you, Louis Pierre, and Tiaguinho) for all these years' support. You made impossible efforts to make this journey possible, supported me in my choices, and had the most incredible amount of patience anyone could have to keep up with me throughout everything and still be able to care about me. Thank you for being people I can trust and talk with about anything. Love you all.

My final and greatest thanks goes to the person who suffered more with this process besides me, Carolina Matias. Thank you for all the hours on the phone and video calls, all the times you tried to force me to work when I wasn't motivated, that you pretended to understand what I was complaining about, all the confidence, all the fun, and just for being there. Sorry for the times I wasn't the best company or friend, but I tried my best.

Sílvia Rocha

"The best way to predict the future
is to create it. "

Peter Drucker

Contents

List of Figures

List of Tables

Abbreviations

HRC	Human-Robot Collaboration
ML	Machine Learning
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
DL	Deep Learning
HRI	Human-Robot Interaction
CV	Computer Vision
ROC	Receiver Operating Characteristic
ConvNet	Convolutional Neural Network
IOU	Intersection Over Union
DNN	Deep Neural Network
MMED	Max-Margin Event Detectors
AMOC	Activity Monitoring Operating Characteristic
MLP	Multilayer Perceptron
MAE	Mean Absolute Error
SMOTE	Synthetic Minority Oversampling Technique

Chapter 1

Introduction

This chapter presents an overview of the context in which the project is inserted. It places it in the intersection of Computer Vision (CV) with Human-Robot Collaboration (HRC), its motivation, the definition of the problem, and the objectives. Additionally, it also presents the overall document structure.

1.1 Context

If machines could automatically interpret and predict the activities people perform in everyday life, many tasks would be revolutionized. Recognizing and predicting human tasks has been a significant subject in Machine Learning (ML), with applications in numerous fields. The ultimate goal of this research area is to create an automatic and intelligent system that can accurately perceive what type of activity is being performed by a person and use that information to predict his next movements and react in a natural way. Despite of human motion being inherently complex, we are able to decompose it into multiple atomic gestures arranged in a temporal order like "grab", "move" or "release".

In these problems of motion prediction and recognition, input data is typically collected as RGB images or 3D skeleton data using a vision system or device.

Looking particularly at the context of Industry 4.0, manufacturing systems are shifting to an intelligent level and HRC is emerging as an important research field that seeks to develop methods to allow human operators to work alongside robots in a close manner. These methods are essential to leverage the strengths of both elements and increase adaptability and productivity.

Although the idea of these two agents working together sounds appealing, there are still a lot of technical constraints that need to be solved. In particular, existing safety solutions might be reliable and trusted in laboratories but have low acceptance in manufacturing environments due to a lack of standards and solutions developed specifically for industrial applications. To overcome

this problem, these collaborative robots must adapt to the unpredictability of human behavior while following strict safety standards.

One of the first steps to do this is to discover the operator's location reliably, the task he is executing, and what his next move is going to be. To do that, we must have methods to classify and predict human motion and use that to determine what tasks the robot can do to help the human operator.

Initially, most of the solutions in this area was solved with traditional ML models. However, most recently, Deep Learning (DL) started to be applied and have increased significantly in accuracy by overcoming a lot of the constraints faced when using traditional ML methods. These methods are able to improve performance and scalability for more complex problems since they are a better approach to learn differentiating features.

1.2 Motivation

As previously mentioned, action recognition and prediction are currently a hot topic in Computer Vision. The recent introduction of Deep Learning methods and the increasing easiness in accessing the computational resources required to train these algorithms has allowed increasing their accuracies in a significant way and has motivated a lot of researchers to invest in it due to the enormous amount of applications it can have like Automatic Surveillance, Autonomous Driving or Human-Robot Interaction.

Inside this later field, both industrial setups and domestic environments can be optimized by giving a robot the perception of what the human is doing. The focus of the present dissertation is then the optimization and automation of manufacturing processes in industrial contexts. The concept of safe HRC for industrial setups is one of the hottest topics inside Robotics due to industry 4.0 that conceptualizes rapid change to technology.

Therefore, a system that can accurately perform early task recognition and short- and long-term movement prediction simultaneously and in real-time is an essential contribution to the field since it is crucial to ensure HRC safety and smoothness.

1.3 Problem Definition

The difficulty with this automatic activity estimation comes mainly from the dependency of human operator motion on many factors, resulting in a large diversity of ways to perform the same activity. Furthermore, environmental settings like lighting conditions, camera settings, and fixed or dynamic camera positions have a direct impact on the system's performance.

There are already several algorithms able to cover a limited set of movement types in some very confined environmental settings (fixed cameras and controlled lightning conditions), making their usage very use-case restricted. Also, most systems only focus on recognition, or prediction, existing very few solutions that allow having both outputs. Not only that, but most of the present systems focus mainly on simple movements like walking, waving, or sitting, presenting a gap

in the recognition and prediction of more complex and detailed human tasks. This gap is more noticeable in industrial settings, where the movements are very detailed and involve interaction with other elements like objects or robots.

Given this, a system that automatically detects and predicts complex human tasks, analyzing human motion from information acquired from chosen robust sensors in real-time, would greatly impact the computer vision area, when in an industrial HRC setting.

1.4 Research Questions

The present dissertation aims at testing the following hypothesis: "It is possible to develop a robust, real-time and integrated framework for industrial applications, using machine learning methods, for simultaneous human motion prediction and early recognition".

This hypothesis is decomposed in the following research questions:

- **Research Question 1** - Can the developed system be robust enough to generalize characteristics like the operator's height, pose and distance to the camera?
- **Research Question 2** - What is the level of impact in the performance of a model of prediction or classification whilst having retro-feedback from the other one?
- **Research Question 3** - Should the solution consist of a single model that outputs motion classification and prediction simultaneously or of an integrated pipeline of two different models that work symbiotically?
- **Research Question 4** - Can the system be optimized in such a way that it is able to process input data in real-time in a machine with low computational resources?

1.5 Objectives

The main goal of the present dissertation is to explore and develop an intelligent integrated framework for movement prediction and early recognition, yielding synergy from their mutual benefits, for human tasks in industrial collaborative robotics scenarios based on camera information, in real-time.

The described framework must be able to fulfill the following requirements:

- The system must be able to operate in real-time, that is, it must be able to complete its time-critical processes with acceptable timeliness.
- Accurate prediction of a human's movement at a given time.
- Accurate early recognition of a human's task at a given time.
- Be able to use movement prediction to improve the movement classification accuracy.

- Be able to use movement recognition labels to improve short- and long-term movement prediction accuracy.
- Be able to simultaneously use vectors of recognition label probabilities in movement prediction and the predicted sequence of movements for the next t milliseconds to refine the early recognition results.
- Usage of lightweight models that do not require computational resources that are not typically available in this type of industry.

1.6 Thesis Structure

The remainder of this dissertation is divided into four more chapters.

Chapter 2 presents the background of the datasets (Section 2.3) methodologies, and algorithms being considered for the project and existing technologies and approaches to the same or related problems in Computer Vision (Section 2.1) and Human Robot Collaboration (Section 2.2).

Chapter 3 includes an overview of the proposed solution, including its architecture, more technical details, the several components of the final solutions and a description of a dataset produced for live classification and prediction.

Chapter 4 presents several experiments conducted with each chosen dataset, either for each separate component or the integrated framework with both outputs. The chapter is divided into four main sections: Section 4.1 that describes the data preparation for each dataset, Section 4.2 that describes the experiments conducted for early recognition, Section 4.3 that describes the experiments conducted for movement prediction and Section 4.4 that describes the experiments conducted with the two types of networks combined. Each of these is divided into three subsections, one for each dataset.

Lastly, Chapter 5 presents the final remarks of this work, its limitations, and the problems found. It also presents suggestions for future work that would help mitigate said limitations.

Chapter 2

Early Recognition and Movement Prediction approaches and techniques

This chapter presents an overview of the state-of-the-art solutions and common challenges faced in the field. The chapter is divided into three sections. First, a summary of solutions in computer vision used for early recognition and movement prediction, both in a separate and integrated way. Then, analysis on existing solutions with application in Human-Robot Collaboration. Lastly, there is a section dedicated to existing datasets of skeleton data.

2.1 Computer Vision

Computer Vision began more seriously in universities during the decade of the 1960s. It was meant to reproduce the human vision system as a first step to develop intelligent behavior for robots. At the time, some predicted that a machine as intelligent as a human would be created within a generation. In the 1970s, the early steps for many of the algorithms used nowadays were established. This includes edge extraction, line labeling, optical flows, and motion estimation [29].

In 1980, Fukushima proposed the 'neocognitron', which is a hierarchical, multilayered artificial neural network proposed used for pattern recognition tasks, and served as the inspiration for convolutional neural networks[8].

In 2001, two researchers at MIT introduced the Viola-Jones framework, the first face detection system that works in real-time. Between 2010 and 2011, image recognition algorithms use was intensified, as seen in Goggles, a Google image recognition app for searches based on pictures taken by mobile devices, and the tag photos system at Facebook.

In 2012, the field of artificial intelligence had its breakthrough at the ImageNet Large Scale Visual Recognition Challenge with the introduction of a deep neural network called AlexNet that was a game-changer. This network was the first to adopt an architecture with consecutive convolutional layers. The final fully connected layer in the network contains a softmax activation

function that provides a vector that represents a probability distribution over 1000 classes. Deep neural networks like this were improved in the following years, making them the standard for image recognition tasks.

2.1.1 Early Recognition

Early recognition is a field of Computer Vision that aims to recognize an action as soon as possible, using partial action sequences. There have been developed many models for human action recognition in CV.

Wang et al. [40] focused on recognizing human actions in a real video where irrelevant factors dominated human activities. They used the ActionThread dataset to train their framework, labeled shot by shot. In this approach, the authors studied the benefits of removing non-action video segments, i.e., segments with no human action. They learned a non-action classifier that was used to down-weight irrelevant video segments. This non-action classifier can be used to identify high-precision non-action shots and subsequently improve the performance of action recognition systems. For recognition, the authors used Least-Squares Support Vector Machines (LSSVM). They proved that pruning non-action video segments improved the results on the Average Precision metric, on average, by 13,7% reaching up to 34,1% in some of the tested datasets. Despite that, this solution is focused on improving the accuracy of offline processing rather than the timeliness of the decision-making.

In Hoai et al. [10], it is proposed Max-Margin Early Event Detectors (MMED), a formulation for training event detectors that recognize partial events. The method is based on an extension of Structured Output SVM that accommodates sequential data. This was the first paper in computer vision that proposed a learning formulation for early event detection. The authors used the area under the ROC curve for accuracy comparison, Normalized Time to Detection (NTtoD) for benchmarking the timeliness of detection, and F1-score for evaluating localization quality. At a 10% false positive rate, MMED can detect the expression with an observational ratio of only 0.47. Other methods needed between 0.55 and 0.71 ratios for the same false-positive rate. The framework is tested using the Auslan dataset, an Australian sign language one, the Extended Cohn-Kanade dataset, a facial expression one, and the Weizmann dataset, a human action one.

Pavlovic et al. [24] focused on learning models of human dynamics by using switching linear dynamic system models. The results show that these models are a promising tool for figure motion analysis and could play a key role in gesture recognition and summarize approximate inference techniques. Additionally, they introduce a variational inference algorithm that shows the benefit of interpreting classical statistical models as (mixed-state) graphical models. This method was used for segmentation and offline recognition and presented limitations when considered for a task of early recognition.

Aliakbarian et al. [1] proposed a new action anticipation method that presented high prediction accuracy even with a minimal observational ratio of the video sequence. The authors developed a multi-stage LSTM architecture that leverages context-aware and action-aware features. Furthermore, they implemented a novel loss function that encourages the model to predict the correct

class as early as possible. This method outperformed the state-of-the-art action anticipation methods for early prediction with a relative increase in accuracy of 22.0% on JHMDB-21, 14.0% on UT-Interaction, and 49.9% on UCF-101.

Wang et al. [41] presents a system that is LSTM-free by being based on 2D ConvNet that does not require the accumulation of video frames for 3D ConvNet filtering. The Weighted Multi-Region ConvNet is designed to capture multiple spatial and short-term temporal cues simultaneously using primary and secondary regions, i.e., the foreground and background. The authors obtained a 76.2% accuracy for the u channel and a 77.4% for the v channel, a 75,9% accuracy when comparing with fusion methods using RGB data of UCF101 split-1, and a 78,8% accuracy when comparing with the Dynamic Network using only RGB for a fair comparison. This approach achieves state-of-the-art performance among other low-latency algorithms on the UCF101 and HMDB51 datasets. Additionally, it outperforms the 3D ConvNet based C3D algorithm that requires video frame accumulation.

Ma et al. [22] presents an alternative to the conventional training of LSTM models where the training loss only considers classification error. This paper proposes that the detection score of the correct activity category, or the detection score margin between the correct and incorrect classes, should be monotonically non-decreasing as the model observes more of the activity. The authors use ranking losses that penalize the model when it violates these monotonicities. This is used alongside the classification loss while training the LSTMs. The performance is evaluated on the ActivityNet and shows significant benefits in both activity detection and early detection tasks. For LSTM-s, the improvements are between 4.1 5.9% at all IOU thresholds. The relative improvement of LSTM-m and LSTM-s over LSTM increases with higher IOU thresholds.

Furnari et al. [7] proposes Rolling-Unrolling LSTM. This learning architecture is based on encoder-decoder sequence to sequence models for text processing and aims to anticipate actions from egocentric videos, i.e., first-person videos. The method has three components: two LSTMs to model the sub-tasks of summarizing the past and predicting the future, a Sequence Completion Pre-Training technique that encourages the LSTMs to focus on the different sub-tasks, and a Modality ATTention (MATT) mechanism to efficiently fuse multi-modal predictions performed by processing RGB frames, optical flow fields and object-based features. This approach reaches the second-best performance on ActivityNet - a third-person video dataset - compared to methods not based on unsupervised pre-training and can be generalized to the task of early action recognition.

Tran et al. [32] proposes a framework based on knowledge distillation, where the network for early recognition is viewed as a student model. This student is trained using knowledge distilled from a more knowledgeable teacher model that peeks into the future and incorporates other observations about the action in consideration. This approach has the advantage of allowing the usage of semi-supervised learning settings, utilizing both the labeled and unlabeled training data. Evaluating the NTU RGB-D dataset results, this approach obtains an AUC percentage of the ROC metric of 62,8%, which is better than the compared approaches of LSTMs and RNNs.

Similarly, Tran et al. [33] presents a knowledge distillation framework that uses an action recognition network to supervise the training of an action anticipation network, guiding the latter

to attend to the relevant information needed for correctly anticipating the future actions. They introduce a loss function that accounts for positional shifts of semantic concepts in a dynamic video. This is a form of self-supervised learning and thus takes advantage of unlabeled data. Replacing the loss function, the performance increased to 75.8%. Using symmetric bidirectional attention loss, they achieved a performance of 76.6%, obtaining the new state-of-the-art result on the JHMDB dataset, using both RGB and Optical Flow.

Wang et al. [35] proved the importance of identifying the action starting point. For this, they proposed a method based on a bidirectional RNN that computes the probability of a particular frame to be the starting point by comparing the dynamics of the actions before and after the frame. Bidirectional LSTM can keep two separate information flows: forward and backward. Using the Montalbano Gesture dataset, they showed that this proposal performs better than others in any situation where the starting point is unknown by presenting an AUC of 61,2%. It also outperforms the previously mentioned MMED proposal in [10] when comparing the AMOC curves. Additionally, it exceeds the work of Ma et al. in [22] presenting an AUC of 56,14% against their 49,64%.

Alvee et al. [2] compares the performance of various ML algorithms in the domain of human activity recognition. The authors present the most accurate algorithm to be SVM at 99,68% and the lowest scoring algorithm the Naive Bayes classifier with 77%, using the common dataset. LSTM is presented with an accuracy of 90,70%, and MLP is the second-highest scorer with 98,57%.

These methods focused on studying the benefits of different feature encodings and classifiers for early recognition without considering the benefits of movement prediction to improve the accuracy.

2.1.2 Short- and Long-Term Movement Prediction

As technology develops, the ability to predict how an environment is changing and what will be the behavior of the objects is becoming more crucial. Movement prediction can be divided into video, action, trajectory, and human motion. In this review, the focus is mainly on approaches in this latter field but also in the video prediction category. Video prediction aims to forecast future frames based on previous frames, relying on labeled data to do it. Human motion prediction seeks to predict future human poses given on past motion and focus on predicting changes in the dynamics of the observed agents.

Ye et al. [43] applies concepts from point cloud learning by using dynamic temporal learning to capture spatial and temporal information by splitting trajectory prediction into these two dimensions. The agents are viewed as an unordered point set in the spatial dimension, making it easy to apply point cloud learning techniques to model their locations. The authors add dynamic temporal learning to model agents' motion over time. Spatial learning consists of pointwise feature learning, point-voxel feature propagation, voxelwise feature learning, voxel-point feature propagation, and dual representation fusion. Dynamic Temporal Learning consists of Multi-interval Learning and Instance Pooling. Analyzing its results on the Argoverse test set, the approach outperforms the existing ones in minADE1, minFDE1, and MR1 without any complex postprocessing and is

the first method to achieve minADE1 less than 1.7m, minFDE1 less than 3.7m, and MR1 less than 0.59.

Liu et al. [18] presented a DNN-based framework that predicts and compresses video sequences in the latent vector space. The compression of individual frames is obtained by a deep autoencoder trained with a generative adversarial network (GAN), and the temporal correlation within the video frame sequence is exploited by using a convolutional LSTM network to predict the latent vector representation of the future frame. This GAN-based autoencoder architecture works together with trainable quantization and entropy coding. The convolutional LSTM network only stores the differences between the predicted and actual representation in low dimensional latent space, resulting in entropy reduction of the residuals. Finally, they investigate the effectiveness of the algorithm while performing perceptual tasks in latent space, showcased in anomaly detection. This experiment was unsupervised and presented results that outperformed the state-of-the-art. Using the ROC curve metric, they obtained 90,9%, 93,6%, 88,2%, 94,5%, and 85.4% on the UCSD Ped1, UCSD Ped2, Subway Entrance, Subway Exit, and CUHK Avenue datasets, respectively.

Jain et al. [12] proposed an approach that combines the power of high-level Spatio-temporal graphs and sequence learning success of RNNs.The method developed is scalable and aims at casting an arbitrary Spatio-temporal graph as a feed-forward RNN mixture that is jointly trainable. This is a generic method that can transform any Spatio-temporal graph by employing a particular set of well-defined steps. In the human motion application, the motion was represented over st-graphs and learned to model them with S-RNN.According to the st-graph, the spine interacts with all the body parts, and the arms and legs interact with each other. This S-RNN outperforms both LSTM-3LR and ERD on short-term motion forecasting on all activities. The authors consider short-term as up until 500ms. This approach can generate human-like motion even though it diverges from the ground truth label in the long term.

Tonchev et al. [30] proposes the prediction of human motion using a Gated Recurrent Unit (GRU) network, a variant of RNNs. The forecast is based on the human skeleton model and joints position change throughout time. This network was optimized by substituting the weighting of inputs and recurrent outputs with convolution utilizing the graph structure of the human skeleton. This proposal is outperformed in earlier stages, presenting better results between the 320ms and the 1000ms but by a small margin. This architecture presents an intuitive approach to calculate the graph by introducing a weight matrix that captures the joint displacement on a frame-to-frame basis and modifies the GRU learning parameters to address the spatial and Spatio-temporal structure of the human motion.

Phillips et al. [25] investigated the issues that arise in state-of-the-art autonomy stacks under localization error and designed a system that jointly performs perception, prediction, and localization. The architecture can reuse computation between the three tasks, making the correction of localization errors efficient. This approach used multi-task learning to jointly localize against an HD map while also performing object detection and motion forecasting and showed that localization errors could be successfully detected and corrected in less than 2 ms of GPU time.

Zhang et al. [44] says that 3D joints can be seen as sparse point clouds, transforming this problem into a problem of point cloud prediction. The authors propose to predict a sparse set of locations on the body surface that correspond to motion capture markers. Given such markers, they fit a parametric body model to recover the 3D body of the person. These sparse surface markers also carry detailed information about the human movement that is not present in the joints, increasing the naturalness of the predicted motions. To mitigate the accumulated errors over time in motion prediction that result in joints or markers that diverge from actual human bodies, they fit the SMPL-X body model to the projections at each step, projecting the solution back onto the space of valid bodies before propagating the new markers in time. This approach produces state-of-the-art results and realistic 3D body animations. This work presents an alternative way of representing 3D bodies that do not use joints but markers on the body surfaces. It offers a margin of improvement given that the recursive projection scheme slows down the inference process, and the motion realism is still not comparable with the ground truth.

2.1.3 Integrated Approaches

After this more comprehensive view regarding early recognition and movement prediction, it is essential to focus on the possible benefits of combining these two fields of computer vision to obtain better results. In Table 2.1 all the state-of-the-art methods explored are summed up regarding their capabilities of prediction, recognition and retro-feedback.

In Wang et al.[36] this combined approach was considered for the first time. The authors proposed a framework that allows these two problems to be addressed jointly. The method proposed is based on a Recurrent Neural Network (RNN) but in an extended way that allows for the integration of multiple systems. The framework contains two Long Short-Term Memory (LSTM) RNNs, one for movement prediction and another for early recognition. The former is designed to accommodate the output of the recognition system. At the same time, the latter uses the predicted sequence of body movements produced by the movement prediction LSTM to output the vector of class probabilities. This work presents as main novelties the justification and empirical evidence showing that movement prediction is beneficial for early recognition and vice-versa. The computation of the probability of an event starts with utilizing the prediction LSTM to generate samples of plausible future observation sequences and subsequently compute the marginalized probability over the sample set. The main benefit that comes from the usage of this LSTM is the preservation of information. The authors were able to outperform several state-of-the-art methods for early recognition and individual systems, obtaining an accuracy of 80% with only 0.4 of observational ratio. Regardless, it presents a few limitations in the movement prediction network. In this LSTM, the results have a margin for improvement given that even with ground truth labels, the average prediction error for the joints on the arms is around 7cm when predicting 200ms into the future of the action sequence. This is the only successful approach found in Computer Vision that combines these two components, revealing an enormous potential to develop a solution that improves the results obtained with this approach and addresses its limitations.

Table 2.1: Approaches and algorithms comparison in CV

Method	**Recognition**	**Prediction**	**Retro-feedback**
[36]	*Yes*	*Yes*	*Yes*
[43]	*No*	*Yes*	*No*
[18]	*No*	*Yes*	*No*
[1]	*Yes*	*No*	*No*
[41]	*Yes*	*No*	*No*
[10]	*Yes*	*No*	*No*
[40]	*Yes*	*No*	*No*
[24]	*Yes*	*No*	*No*
[12]	*No*	*Yes*	*No*
[22]	*Yes*	*No*	*No*
[7]	*Yes*	*No*	*No*
[37]	*Yes*	*No*	*No*
[32]	*Yes*	*No*	*No*
[33]	*Yes*	*No*	*No*
[35]	*Yes*	*No*	*No*
[30]	*No*	*Yes*	*No*
[25]	*No*	*Yes*	*No*
[44]	*No*	*Yes*	*No*
[2]	*No*	*Yes*	*No*

2.2 Movement Prediction and Recognition in Human-Robot Collaboration

This is a research field with a wide range of applications, future scenarios, and potentially a high economic impact. HRC is a research area that comprises classical robotics, cognitive sciences, and psychology [3]. There are four main methods to provide safety in human-robot scenarios: control, motion planning, prediction, and consideration of psychological factors. [15] The proposed work is focused on the motion prediction component of HRC. The early recognition component is used to improve the results obtained in the joint movement prediction to ensure smoothness and safety in the interactions.

"Efficient collaboration requires a common plan for all involved partners. To gain a joint intention, they need to know the intentions of the other team members and what they are doing. Based on that knowledge, a robot can plan its actions that will eventually lead to fulfil the joint intention and reach a common goal"

Bauer et al. [3]

In Table 2.2, all the approaches for human movement prediction and early recognition in the context of HRC are summed up.

2.2.1 Early Recognition

Xia et al. [42] proposes an overall model of vision-based hand gesture recognition for HRC and reviews its technical components: sensor technologies, hand gesture detection, segmentation, and hand gesture classification. Looking at the classification component, the authors talk about template matching as the first action recognition approach where there is a match between an input image and a template. Inside ML, they talk about SVMs - who have a person-independent performance and are reliable with different sizes and complex backgrounds - and Deep Learning, namely CNNs and RNNs. Afterward, they talk about geometric features used for static gesture recognition. The Hidden Markov Model, which is very suitable for describing sequence models, is also mentioned for context-sensitive occasions. Lastly, they talk about Dynamic Time Warping, a pattern-matching technology of nonlinear time normalization. It has small training costs and has advantages in eliminating the time difference between different Spatio-temporal patterns but has poor recognition effect and stability. After this analysis, Deep Learning is presented as the best option considering dynamic scenarios and large amounts of data.

A similar work made by Liu et al. [20] focus K-Nearest Neighbours, Hidden Markov Model, SVMs, Ensemble Method, Dynamic Time Warping, Artificial Neural Networks, and Deep Learning as gesture classification techniques. This paper also shows that Deep Learning is the rising field for gesture classification, alongside Artificial Neural Networks that have a black box nature.

Li et al. [16] proposes a transfer learning-enabled action recognition approach to facilitate reactive robot control in HRC assembly. A transfer learning-based ST-GCN architecture is proposed to learn human actions from these data. Our proposed deep transfer learning network includes an ST-GCN feature extractor, an action classifier, and one domain adaptation module. The feature extractor consists of nine layers of ST-GCN, which achieved by spatial-temporal graph convolution and pooling operations. Subsequent action classifier is completed by concatenating a fully connected layer(FC1) and one output layer. This network presents an improvement on the mean average precision (mAP) when compared with the ST-GCN, obtaining a 95,56% against 86,67%.

Wang et al. [37] presents an approach that takes advantage of multiple cameras while satisfying the constraints due to limited communication bandwidth and processing power. At each time step, the method decides the best camera to use so that a confident recognition decision can be reached as soon as possible. The camera selection problem is seen as a sequential decision process, and the view selection policy is learned using reinforcement learning. They also developed an RNN architecture to account for the unobserved video frames and the irregular intervals between the observed frames. This presents an essential tool when considering the proposed solution's industrial setup. The experiments conducted demonstrated the approach's effectiveness for early recognition of human actions.

2.2.2 Movement Prediction

Liu et al. [21] proposes a framework that combines RNN and Inverse Kinematics to predict human arm motion. A modified Kalman filter (MKF) is applied to adapt the model online. The authors use LSTM and RNN to denote the transition of a single LSTM cell and the N-to-1 prediction, respectively. This method improves the prediction accuracy by approximately 14% compared to the state-of-art on seen situations. Additionally, it stably adapts to unseen situations by keeping the maximum prediction error under 4cm, which is 70% lower than other methods. Furthermore, it is robust to partial occlusions obtaining the same wrist predictions and 20% less variation in the elbow prediction.

Liu et al. [39] uses an LSTM to extract temporal patterns of human motion, automatically outputting the prediction result before motion takes place. This avoids feature extraction due to its end-to-end characteristic. The experiment achieved 83% accuracy of motion prediction when the data length of manipulation motion is 60%. The proposal uses as base the VGG16 CNN and implements the LSTM on top of it. This architecture outperforms DCNNs and RNNs with an accuracy between 95% and 99%, depending on the number of frames extracted from the video.

Zhou et al. [46] presents the implementation of an early turn-taking prediction algorithm using an LSTM. This algorithm is significantly superior to its algorithmic counterparts and is more accurate than the human baseline when little partial input is given (less than 30% of full action). After observing more information, the algorithm can achieve comparable performances as humans with an F1 score of 0.90.

Liu et al. [19] presents a way to model product assembly tasks as a sequence of human motions. The Hidden Markov model is used in the motion sequence to generate a motion transition probability matrix making motion prediction possible. Even though the industrial robot needs to respond in a continuous-time domain, the system models an HRC task as a discrete HMM model. Therefore, only one observation is generated during each motion, making the time between two movements ignored. The validation of this system is based on the success of a conducted case study of an aircraft assembly. There are no concrete relevant metrics presented in the paper.

Tortora et al. [31] proposes an interface capable of anticipating user intention while performing reaching movements on a working bench. The system integrates motion intention and direction prediction. The motion intention is implemented using a 2-class Hidden Markov Model, while the direction prediction uses a Gaussian Mixture Model. The dynamic stopping criteria proposed allows adjusting the trade-off between early anticipation and accuracy. This method outperforms the others by achieving a real-time classification accuracy of 94,3±2,9% after 160.0ms ± 80.0ms.

Zhao et al. [45] presents a modified minimum jerk model (MMJM), containing three input parameters: motion duration, motion destination, and early-stage fingertip trajectory. These unknown parameters are defined by determining the optimal starting time of motion prediction and employing Gaussian process regression models (GPRs). The experimental results show that the MAEs of motion destination prediction is less than 20 mm. The MAE of motion duration prediction is less than 0.05 sec, which is more effective than others in early-stage motion prediction.

Table 2.2: Approaches and algorithms comparison in HRC

Method	**Recognition**	**Prediction**	**Retro-feedback**
[39]	*No*	*Yes*	*No*
[21]	*No*	*Yes*	*No*
[46]	*No*	*Yes*	*No*
[42]	*Yes*	*No*	*No*
[20]	*Yes*	*No*	*No*
[16]	*Yes*	*No*	*No*
[19]	*No*	*Yes*	*No*
[31]	*No*	*Yes*	*No*
[45]	*No*	*Yes*	*No*
Proposed Approach	*Yes*	*Yes*	*Yes*

2.3 Human Motion Datasets

Nowadays, numerous public datasets present excellent characteristics for this type of application. Most of these datasets have labels and other annotations, and some of them provide data for both 2D and 3D approaches. In Table 2.3 some of the most relevant datasets in this context, their content and 3D/2D data availability are presented.

The first dataset that is important to highlight is JHMDB [13]. It consists of 960 video sequences belonging to 21 actions. It contains video and annotation for puppet flow per frame (approximated optimal flow on the person), puppet mask per frame, joint positions per frame, action label per clip, and meta label per clip (camera motion, visible body parts, camera viewpoint, number of people, video quality) [4]. The joint positions available are in 2D, making this dataset an option only for 2D approaches.

Lonescu et al. [11] introduced the Human3.6M dataset. The dataset presents 3.6 million accurate 3D Human poses, acquired by recording five female and six male subjects under four different viewpoints. They use the camera parameters to project the 3D joint positions and obtain accurate 2D pose information. This makes this dataset suitable for both 2D and 3D approaches and an excellent tool to train the framework to become robust to the operator's size and pose.

Kay et al. [14] presented the Kinetics dataset that contains 400 human action classes, with at least 400 video clips for each action. Each clip lasts around 10s, is taken from a different YouTube video, and is human annotated with a single action class. This dataset is limited to 2D applications.

Finn et al. [6] introduced the Robotic Pushing Dataset for video prediction for real-world interactive agents, which consists of 59,000 robot interactions involving pushing motions, including a test set with novel objects. It is divided into one training set and two test sets of previously seen and unseen objects. This dataset aims only at Unsupervised Learning approaches.

Heilbron et al. [9] introduced, in 2015, the ActivityNet dataset that provides samples from 203 activity classes with an average of 137 untrimmed videos per class and 1.41 activity instances per video, for a total of 849 video hours. It has both first-person and third-person videos and has only 2D data available.

Shahroudy et al. [27] introduced the NTU RGB+D dataset. It involves 56,880 samples of 60 action classes collected from 40 subjects. These actions took place under 17 different scene conditions and were captured using three cameras with different horizontal imaging viewpoints. This dataset has depth maps, 3D skeleton joint position, RGB frames, and infrared sequences. This makes the dataset suitable for 2D and 3D approaches and an excellent way to make the framework robust to different viewpoints and operator sizes.

Soomro et al. [28] presented the UCF101 dataset that consists of 101 action classes, over 13000 clips, and 27 hours of video data. These videos are only labeled with action classification and are usable for 2D applications.

Li et al. [17] presents the MSR Action 3D dataset, an action dataset of depth sequences captured by a depth camera. This dataset contains twenty different actions and is only suited for 3D approaches.

Escalera et al. [5] introduced the Montalbano Gesture Dataset, an improved version of the ChaLearn 2013 multi-modal gesture recognition challenge with more ground-truth annotations. It consists of 20 Italian gestures performed by 27 different people with 13858 labeled sequences. It has RGB, Depth Maps, Skeleton, and User mask data available, making it suitable for 2D and 3D applications.

Maurice et al. [23] introduced an Industry-Oriented Dataset for Collaborative Robotics that consists of 13 participants performing six different tasks, with 15 trials per participant. The dataset has three types of labels: general posture, detailed posture, and goal-oriented action. There is 3D data available from the 43 sensors placed in the participants' bodies.

Voulodimos et al. [34] presented a workflow recognition dataset that consists of workers executing industrial workflows. It is divided into two datasets: one with one day of work (5 hours and 10 minutes) where the work cycle is executed 20 times and another with two days of work (15 hours and 30 minutes). It has RGB from the videos and a representation using sixth-order Zernike moments.

Rude et al. [26] presented a benchmark dataset for Depth Sensor based recognition. The data is collected during a work cycle on the factory floor and has eleven different labels. Using a Kinect, 10 points from the upper body were collected, so there is 3D data available.

Table 2.3: Dataset annotation and availability comparison

Dataset	**Annotations and Available Data**	**2D Data Processing**	**3D Data Availability**
[13]	*Action and meta labels per clip, 2D joint positions*	*Yes*	*No*
[11]	*Action labels; Time of flight data, 2D pose information and 3D joint positions*	*Yes*	*Yes*
[14]	*Action labels; RGB images from video sequences*	*Yes*	*No*
[6]	*No labels; RGB images from video sequences*	*Yes*	*No*
[9]	*Action labels; RGB images from video sequences*	*Yes*	*No*
[27]	*Action labels; RGB, Depth map, 3D skeleton data (25 joints), and Infrared*	*Yes*	*Yes*
[28]	*Action labels; RGB images from video sequences*	*Yes*	*No*
[17]	*Action label;Depth maps*	*No*	*Yes*
[5]	*Limb and Gesture Action Annotations; RGB, Depth, User mask, and skeleton available*	*Yes*	*Yes*
[23]	*Posture and Goal-Oriented Action Annotations; RGB and skeleton available*	*No*	*Yes*
[34]	*Action Annotations; RGB and Zernike moments representation available*	*Yes*	*No*
[26]	*Action Annotations; RGB and skeleton available*	*Yes*	*Yes*

Chapter 3

Integrated Framework for HRC

This chapter describes the proposed solution and methodological approaches to face critical research challenges. In essence, this chapter explains what is planed to implement to tackle the defined research problem and fulfill all previously defined functional requirements.

3.1 Proposed Solution

To solve the problem identified in Section 1.3, it is proposed to implement an integrated framework where knowledge is shared between the early recognition and the movement prediction networks, increasing their performance.

This framework has two main components: early recognition and movement prediction networks. These two components should directly impact the performance of each other. In the proposed architecture, the input data is provided to both networks for processing at a given time T. The class probabilities vector output of the early recognition network will then be used to feed the movement prediction, and hopefully improve it by being used as an optional input for that network.

In the movement prediction network, the output data must be delivered in the same format as the input data of these networks. This way, it is possible to feed the recognition network with an input that consists of the concatenation of the past frames with the predicted frames. Using this data as input data to the early recognition network will refine the accuracy of this network.

Regarding the final solution, other considerations had to be made and included in this architecture. Firstly, considering the real-time requirements of this system and the limited computational resources, the CRISP methodology's application to the data being received had to be optimized so that, especially in the feature engineering stage, it can be processed quickly enough.

Additionally, considering the defined robustness requirement, it was essential to train these models with diversified data from various datasets and refine their performance using data acquired

by low-budget cameras and uncontrolled lighting conditions that present operators of different sizes and with different positioning relative to the camera.

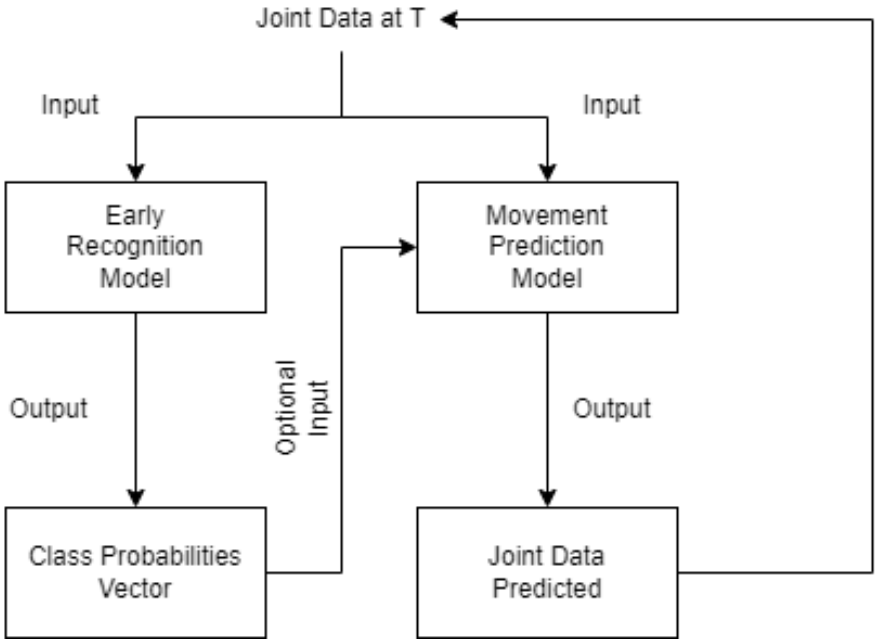

Figure 3.1: General proposed architecture for the integrated framework

In Fig. 3.1, it is possible to see a general architecture of the developed solution. At a certain time T, the system receives an input with a certain number of frames that goes through the early recognition model that provides a class probability vector which, concatenated with the original input, is fed to the movement prediction model. In turn, this model outputs the prediction for the following frames, which are used to increase the amount of relevant data received by the early recognition model.

3.2 Methodology

In order to develop the previously described framework, it was decided to base the work on two different datasets: [5] and [23]. The former has a broader state of the art, allowing for a better comparison of the overall performance of the models. At the same time, the latter was specifically developed for work in collaborative robotics for industries and focused on a single operator at a time, which is this dissertation's use case.

3.2.1 Dataset

Considering the real-time and low-budget requirements of the system, a new dataset was generated. The dataset has available the data and file formats described in Table 3.1, including a total of 132 video sequences from 22 different subjects with 15310 total frames.

Table 3.1: File formats and data available

Data	Format
Videos	*svo*
Skeleton and Hand Keypoints 2D	*csv*
Skeleton and Hand Keypoints 3D	*csv*
Annotations	*csv*

3.2.1.1 Participants

Twenty-two adults participated in the data collection, 13 were males, and 9 were females. Their average age was 26.7 years, the youngest participant was 20, and the oldest was 64. Their average stature was 172.9 cm, varying between 160 cm and 198 cm.

Participants were mainly students and researchers with no particular industrial work experience. Participants were assigned a numerical ID (random number between 1 and 22) to anonymize the data.

3.2.1.2 Experimental Setup

The produced dataset targets actions that are commonly observed in industrial settings. Participants performed three different industry-oriented tasks:

• Grab: Take a screwdriver from the table and return to the position

• Screw: Take the screwdriver to the object and screw

• Drop: Place the screwdriver back into the initial position on the table

Although the items used in the activities were positioned in a way that encouraged the adoption of specific positions, participants were not given instructions on how to perform a given task, i.e., they were never constrained to do any specific movement.

The data was collected from two different angles in each environment and two different environments: sixteen participants in one of them (training scenario) and six in another (validation scenario) to test and improve the model's robustness in these different scenarios.

3.2.1.3 Equipment

A Stereolabs ZED2 in the 1080p mode at 30fps with a resolution of 3840x1080 was used to capture the data. The camera has a depth range of 20cm to 20m, a field of view (FOV) of 110º horizontal, 70º vertical, and 120º diagonal, and neural stereo depth-sensing technology. Additionally, it has two motion sensors (accelerometer and gyroscope) @ 400 Hz and two position sensors (barometer and magnetometer) @ 50 Hz, presenting a 6-DoF visual-inertial stereo simultaneous location and mapping (SLAM) with advanced sensor fusion and thermal compensation. The image sensors have a resolution of dual 4M pixel sensors with 2-micron pixels and a size of 1/3" BSI (backside illumination) sensor with high low-light sensitivity.

3.2.1.4 Annotations

After the data collection, one human annotator manually labeled the motions in all trials, using a taxonomy of actions and postures that were defined beforehand 3.2.

Table 3.2: Taxonomy of actions for the produced dataset

Action	Description
Grab	*Moving an arm towards a target, no object in hand, picking it up and bringing arm back after grabbing.*
Screw	*Rotational screwing movement of the hand.*
Drop	*Placing an object, similar to Grab, but with an object in hand, bringing arm back after releasing.*

3.2.1.5 Skeleton Extraction

After capturing the videos, they were processed to extract relevant information about the participants' poses and motions. In order to achieve this, each video was processed, frame by frame, using the OpenPose and ZED SDK capabilities. With OpenPose, it was possible to extract body and hand keypoints (eighteen points across the body using the COCO Model and twenty-one points in each hand, according to Fig. 3.2).

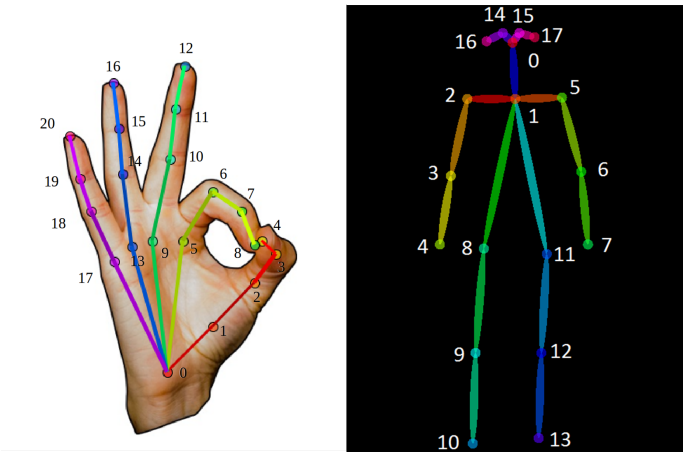

Figure 3.2: Keypoints extracted when using OpenPose

In Fig. 3.3, it is possible to see the extracted keypoints mapped on a frame of the sequences of two different participants. The points obtained with this processing are 2D, corresponding to the location in the RGB image. The entire point cloud was extracted using the ZED SDK to obtain 3D points, and then, for each of the 2D points, the corresponding Z coordinate was obtained.

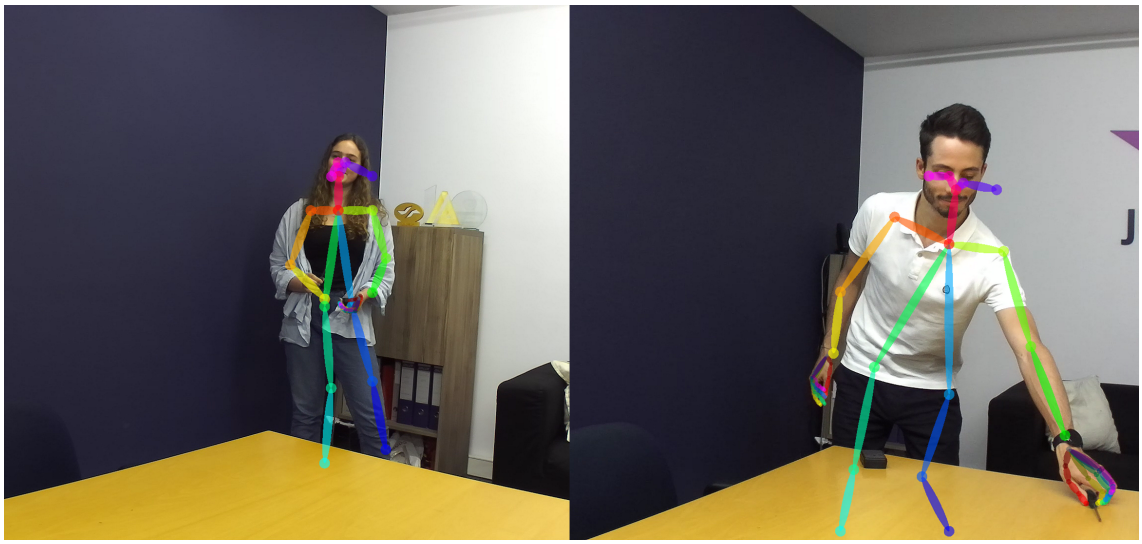

Figure 3.3: Keypoints extracted mapped into the respective frame with two different participants

3.2.2 Recognition and Prediction Framework for Human Action Sequences

All experimental and final models tested were built from scratch, i.e., without resorting to techniques like transfer learning or already developed solutions. This decision was made because this work intends to develop a framework effective for a particular set of conditions that are not focused on in that work: real-time and low-quality data, using OpenPose to detect relevant keypoints. For this reason, a custom solution was implemented, and the usage of datasets commonly referenced in state-of-the-art is only to have a way of comparison with them.

The models were built using Python, with Keras, a deep learning API running on top of TensorFlow's machine learning platform. The isolated models for recognition and prediction were built using the Sequential API, while the final integrated frameworks were built using the Functional API.

The final proposed solution for the integrated framework is based on the results obtained with the Montalbano Gesture Dataset due to some limitations with the others that are detailed in Chapter 4. The system was developed to deal with this dataset's features accurately This network has four LSTM layers: three dedicated to recognition with 1318, 1197, and 988 units, respectively, and one dedicated to the prediction with 988 units. Additionally, there are two dense layers: one after the third recognition LSTM layer with 20 outputs and another after the prediction LSTM with 60 outputs. These dense layers correspond to the class probability vector and one frame's prediction, respectively.

Chapter 4

Experiments and Results

This chapter presents an overview of all experiments conducted on the three selected datasets, from data extraction and preparation to the different implementations of early recognition and movement prediction networks and, finally, the several techniques for integrating these two types of networks. In each section, there is an explanation of difficulties and limitations found during the process and how they were overcome.

4.1 Data Preparation

4.1.1 Montalbano Gesture Dataset

For each participant's action sequence, this dataset has videos with the RGB data, Depth data, and user segmentation mask. Additionally, it has three CSV files: one with the number of frames, fps rate, and a maximum depth value, one with the labels where for each action there is the label, initial and final frame, and another with the skeleton information for each frame.

Skeletons are encoded as a sequence of 20 joints: HipCenter, Spine, ShoulderCenter, Head, ShoulderLeft, ElbowLeft, WristLeft, HandLeft, ShoulderRight, ElbowRight, WristRight, HandRight, HipLeft, KneeLeft, AnkleLeft, FootLeft, HipRight, KneeRight, AnkleRight, and FootRight. Each of these joints has nine values corresponding to World Coordinates (in 3D), Rotation values, and pixel coordinates. A script was developed to extract and organize this information.

Each video sequence might have only one or multiple tasks being performed, so, for this solution, the sequences were divided by action, making it a multiclass problem instead of a multilabel one. The world coordinates of all 20 joints were used to represent each frame. As seen in Table 4.1, the dataset was well balanced, so no further data processing was performed.

Table 4.1: Montalbano gesture dataset label distribution

Label	Samples	Label	Samples
1	*330*	11	*340*
2	*342*	12	*328*
3	*336*	13	*336*
4	*342*	14	*345*
5	*335*	15	*331*
6	*398*	16	*350*
7	*335*	17	*348*
8	*367*	18	*333*
9	*353*	19	*322*
10	*344*	20	*335*

4.1.2 Industry-Oriented Dataset

This dataset has available, for each participant's action sequence, among other data, videos with the RGB data and three CSV files: one with participant's features, one with the annotations for each action, and another with the markers information for each frame. There are 43 markers distributed across the participant's head, torso, right arm, left arm, right leg, and left leg. Each of these markers has available 3D world coordinates. Regarding annotations, there were three different types of annotation available: general posture, detailed posture, and goal-oriented action. Considering the type of tasks defined in each one, the annotations for the goal-oriented actions were chosen. Additionally, there were annotations available from three independent people, but considering the reliability analysis in [23], a simplification was made, and only annotator number one was considered. Similar to the previous dataset, a script was developed to extract and organize this information.

After the first experiments, it was noticeable that there were performance issues, specifically with the learning process for some labels, as can be seen in Fig. 4.6. Taking a closer look at the dataset distribution (Table 4.2), it was possible to understand that the dataset was imbalanced. SMOTE, an oversampling technique, was applied to overcome this issue using the imbalanced-learn python library. This technique takes an under-sampled class sample, plots lines between them, and randomly choose a point in the line to be that class's new synthetically generated sample. Given the results shown in Fig. 4.7, it was possible to understand that the technique did not significantly impact the performance.

Table 4.2: Industry Dataset label distribution for training

Label	Samples	Label	Samples
Reach	*978*	Pick	*786*
Place	*799*	Release	*921*
Carry	*748*	Fine Manipulation	*1128*
Screw	*446*	Idle	*782*

Considering the sample distribution and similarity between some of the labels in the dataset, it was decided to remove some unnecessary classes to simplify the problem. The Idle, Screw, Carry, and Reach labels were removed. The Idle label was removed because it had a low impact considering this framework's goal. The Screw label was removed due to the low number of samples and because it was a particular case of Fine Manipulation, leading to less label reliability. The Reach label was removed due to its similarity to the Release label, given that there are no markers in the fingers to help determine if the hand is closed or not and clarify the difference between the labels. Finally, the Carry label was removed because it happens in-between two other labels, and there was a high disagreement rate in the transition of tasks between the annotators. Additionally, considering the tasks being evaluated, the markers for the right and left legs were also removed from the used dataset. The results after these changes can be seen in Fig. 4.9, representing a significant increase in performance. After this, no other processing of the data was performed.

4.1.3 Live Dataset

For the live framework, instead of using extracted data stored in CSV files as in the previous datasets, the actual video files (in SVO format) were processed to simulate the real-time constraints. Initially, the process used was precisely the one described in Section 3.2.1.5 . Afterward, some body keypoints, considered irrelevant for the tasks being performed, were removed to reduce processing and prediction times. Thus, the models were trained using 21 points from each hand plus 10 points from the body (torso and arms). As also mentioned in Section 3.2.1, this dataset is perfectly balanced, so no further processing was required. In order to evaluate if the system was optimized enough to be able to process frames in real-time in a machine with low computational resources, the processing and prediction time was measured running everything using CPU only. The average time was 0,0823 seconds, meaning that the system can make predictions at approximately 12 frames per second. If a higher rate is desired, it is possible to sacrifice some performance and extract only body keypoints (changing to a more complex body model with 25 keypoints), which takes only an average of 0,0163 seconds, allowing a rate of 60 frames per second.

4.2 Early Recognition

4.2.1 Montalbano Gesture Dataset

The first model developed was a simple LSTM architecture that aimed to understand the initial behavior of the data and how to improve the network. In this version, there was only one LSTM layer with 60 units and one Dense Layer as the output layer with only one unit.

As expected for a network this simple, its performance was below the desired values, as can be seen in Fig. 4.1 and Fig. 4.2, achieving an accuracy of only 5,69% .

In order to optimize this network, it was used the Keras Tuner library that performs hyperparameter tuning, i.e., selects the right set of hyperparameters for a particular ML application. The

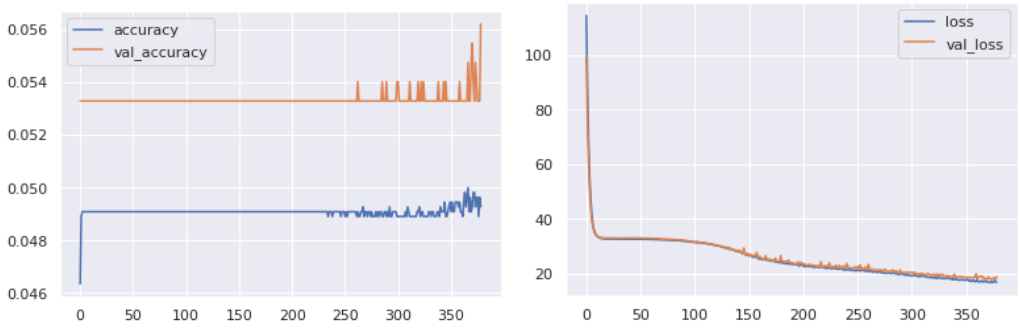

Figure 4.1: Accuracy and loss function of the Montalbano Gesture Dataset first Recognition Model

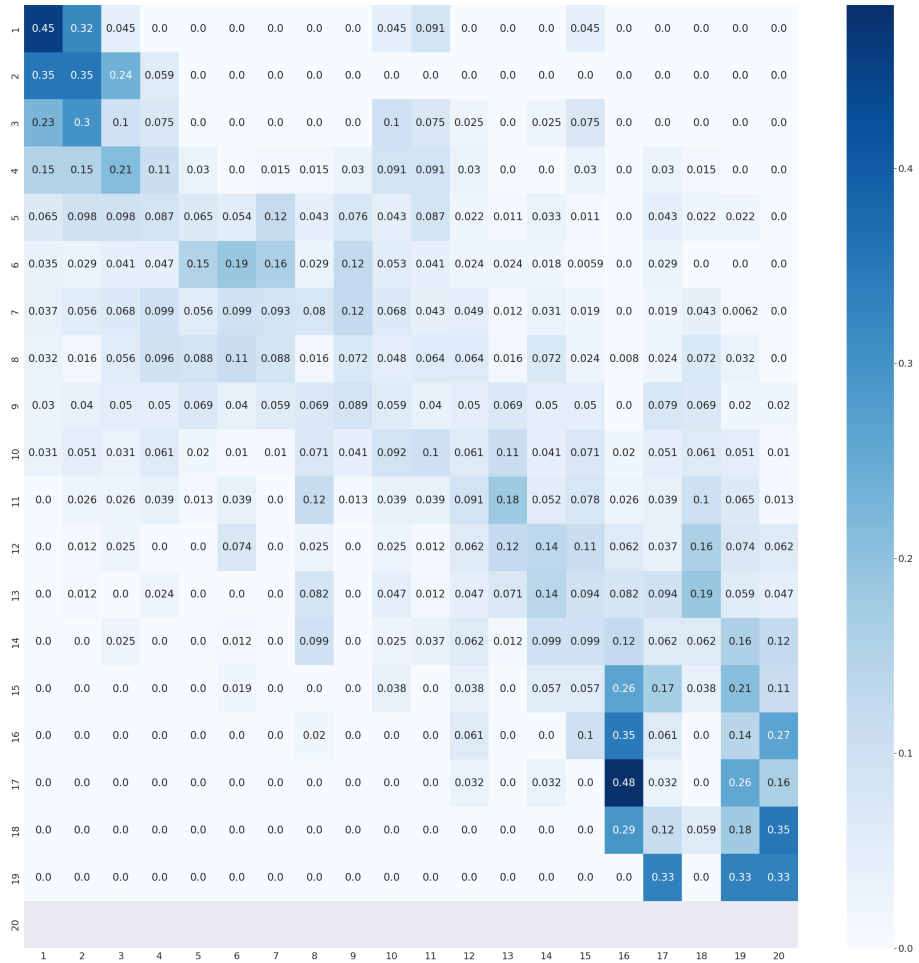

Figure 4.2: Confusion Matrix of the Montalbano Gesture Dataset first Recognition Model

search was conducted using Bayesian optimization, an informed search method. It learns from previous iterations through a probabilistic model, mapping hyperparameters to their corresponding score probability. The hyperparameters and respective possible intervals were chosen in this search according to Table 4.3.

Additionally, label data was restructured by changing the integer representing the class to a

Table 4.3: Hyperparameters for Bayesian Optimization Search in the Gesture Dataset and Early Recognition Model

Hyperparameter	**Description**	**Values**
Number of Layers	*Number of LSTM layers*	*Integer between 1 and 4*
Learning Rate	*Learning rate for the Adam Optimizer*	*Choice between 0.1, 0.01, 0.001 and 0.0001*
Units Layer I	*Number of neurons for LSTM layer I*	*Integer between*
Dropout	*Dropout rate*	*Float between 0.0 and 0.5*

categorical variable, i.e., a vector with the size of the number of existing labels where if that index corresponds to the label, it is set to 1, and, otherwise, it is set to 0. The loss function was also changed to a Categorical Cross Entropy function, frequently used in multiclass classification problems like this one.

With these changes and the output generated by the hyperparameter tuning, the model evolved into 3 LSTM layers with 469, 936, and 1127 units, respectively, and a Dense layer with a number of units equal to the number of existing labels, i.e., 21. The model was trained for 423 epochs until the validation loss stabilized and the training was stopped. The training and validation loss evolution can be seen in Fig. 4.3 . This model achieved a 78,47% accuracy using the first 16 frames of a given sequence.

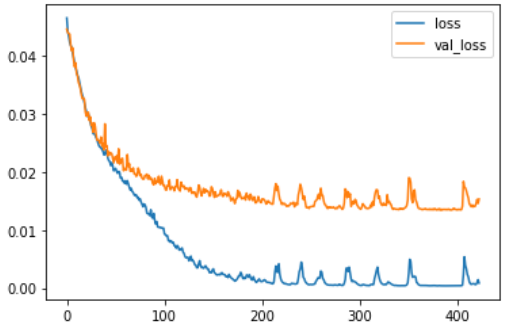

Figure 4.3: Training and validation loss for the recognition model in the Gesture Dataset

4.2.2 Industry Dataset

When training the network obtained for the Gesture Dataset with the data extracted in this dataset, we found that the training loss would converge to zero. However, the validation loss would increase during the training, as can be seen in Fig. 4.4.

Looking at this behavior, it was apparent that there was an overfitting problem, i.e., the model was too closely aligned to the training data and could not generalize to new data. Using, once again, the Keras Tuner library, it was attempted to find a suitable way to solve this problem by testing two different standard solutions: adding dropout layers and increasing the depth of the network.

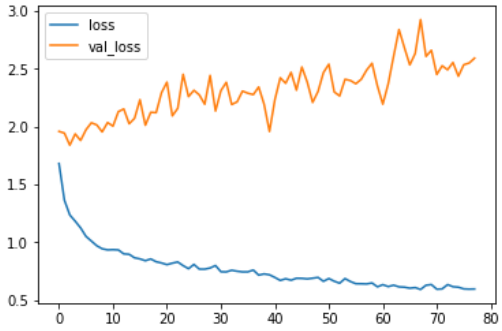

Figure 4.4: Training and validation loss for the recognition model in the Industry Dataset with overfitting problem

Initially, 40 trials with different hyperparameter combinations of values were run, according to the information in Table 4.4.

Table 4.4: Hyperparameters for Bayesian Optimization Search in the Industry Dataset and Early Recognition Model

Hyperparameter	**Description**	**Values**
Number of Layers	*Number of LSTM layers*	*Integer between 1 and 4*
Learning Rate	*Learning rate for the Adam Optimizer*	*Choice between 0.001 and 0.0001*
Units Layer I	*Number of neurons for LSTM layer I*	*Integer between 129 and 3000*
Dropout I	*Dropout rate for Dropout layer I*	*Float between 0.0 and 0.5*
Dropout	*Dropout rate for Dropout layer after last LSTM*	*Float between 0.0 and 0.8*

The results of this search suggested a new model with 3 LSTM layers with 3000, 3000, and 129 units, respectively. Additionally, it added a Dropout layer with a rate of 0.5 between the second and third layers.

Training this new network with the data for 62 epochs was capable of achieving only 25,52% accuracy on the validation data and had the loss behavior that is observable in Fig. 4.5.

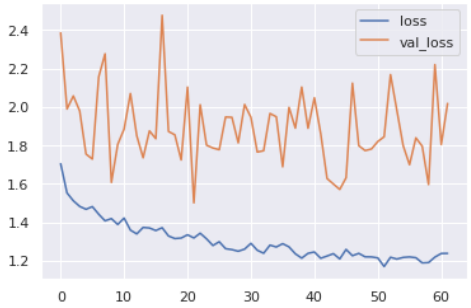

Figure 4.5: Training and validation loss for the recognition model in the Industry Dataset after hyperparameter tuning

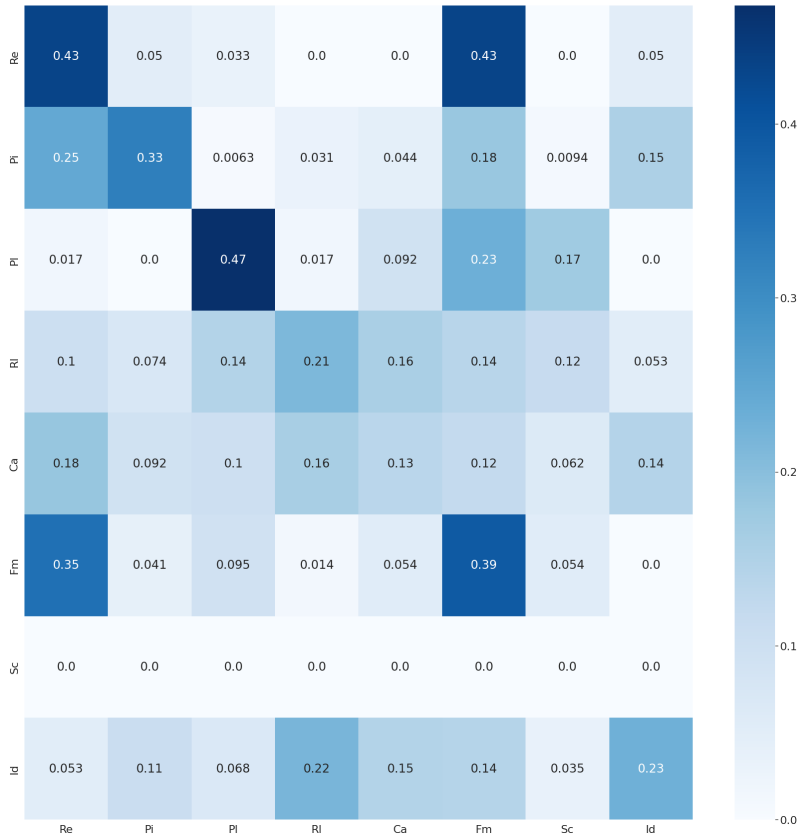

Figure 4.6: Confusion Matrix of the Industry Dataset Recognition Model

Looking at the confusion matrix in Fig. 4.6, it was clear that this dataset had a problem with the Screw action. As described in Section 4.1.2, an oversampling technique was applied, obtaining a new dataset with 1128 samples for each label. This technique led to a slight improvement in the model's performance, achieving an accuracy of 29,03%. The loss and accuracy evolution for this train can be seen in Fig. 4.7 and the confusion matrix in Fig. 4.8 .

Figure 4.7: Training and validation accuracy and loss for the recognition model in the Industry Dataset after oversampling with SMOTE

As explained in Section 4.1.2, the next step was to remove some labels and body markers. With the same architecture as in previous steps, this model achieved an accuracy of 52,27% which

Figure 4.8: Confusion Matrix of the Industry Dataset Recognition Model after oversampling with SMOTE

is an improvement of 104,82% in performance. The accuracy and loss evolution can be seen in Fig. 4.9 and the confusion matrix in 4.10. The F1-score, precision and recall can be found in Table 4.5.

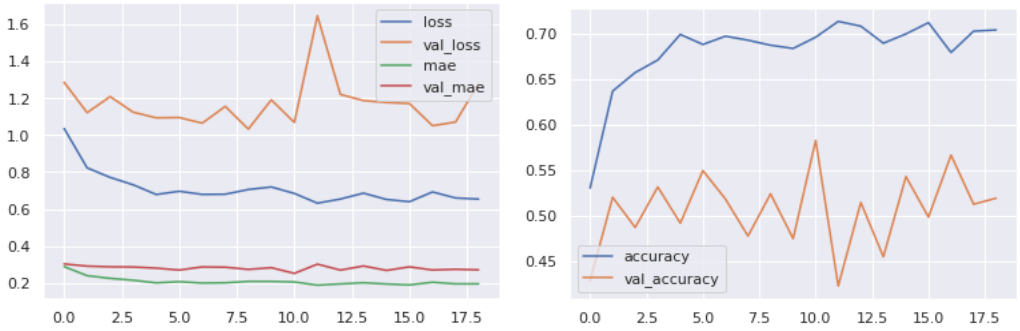

Figure 4.9: Training and validation accuracy and loss for the recognition model in the Industry Dataset after reducing labels and markers

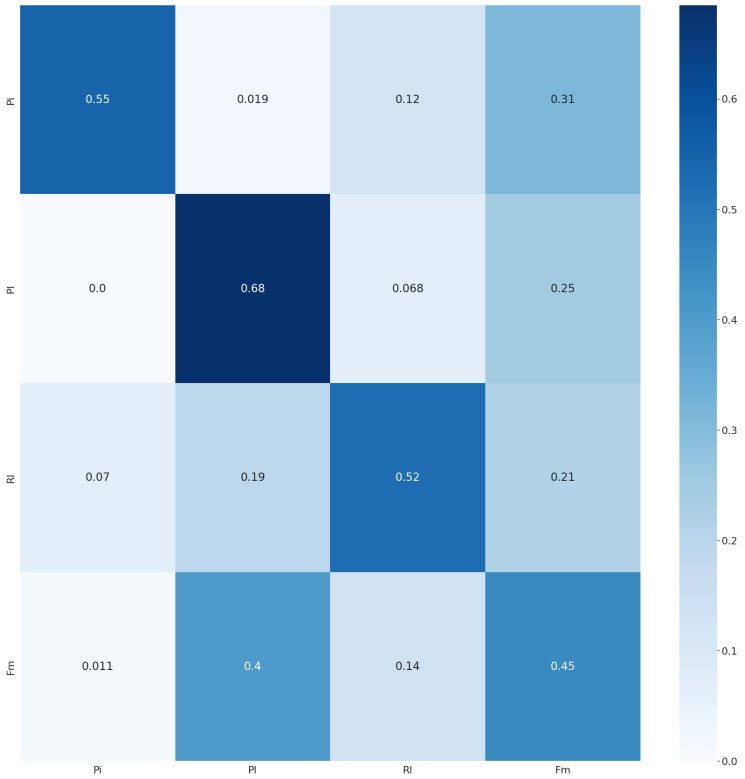

Figure 4.10: Confusion Matrix of the Industry Dataset Recognition Model after reducing labels and markers

Table 4.5: Metrics for the Recognition Model in Industry Dataset

Label	**Precision**	**Recall**	**F1-Score**	**Support**
Pi	*0.55*	*0.88*	*0.67*	*226*
Pl	*0.68*	*0.22*	*0.33*	*232*
Rl	*0.52*	*0.68*	*0.59*	*272*
Fm	*0.45*	*0.37*	*0.40*	*326*
Accuracy			*0.52*	*1056*
Macro Avg.	*0.55*	*0.54*	*0.50*	*1056*
Weighted Avg.	*0.54*	*0.52*	*0.49*	*1056*

4.2.3 Live Dataset

As previously mentioned, initially, the model was trained using all available data. The model had three stacked LSTM layers of 180, 180, and 379 units and a Dense output layer using a softmax activation function with three units (number of distinct labels). This model achieved an accuracy of 66,67% and had a loss evolution that can be seen in Fig. 4.11.

After some body keypoints were removed, as mentioned in Section 4.1.3, this same model was trained with the new input shape achieving similar performance, i.e., the accuracy of 61,11%. The confusion matrix of this model can be seen in Fig. 4.12 and loss evolution in Fig. 4.13.

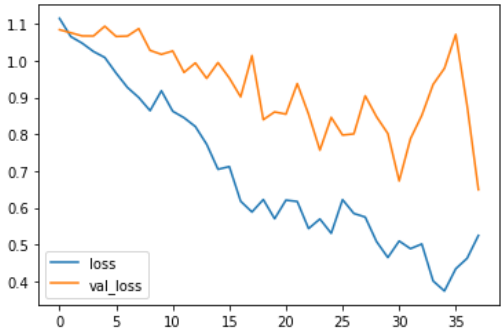

Figure 4.11: Training and validation loss for the recognition model in the Live Dataset with all keypoints available

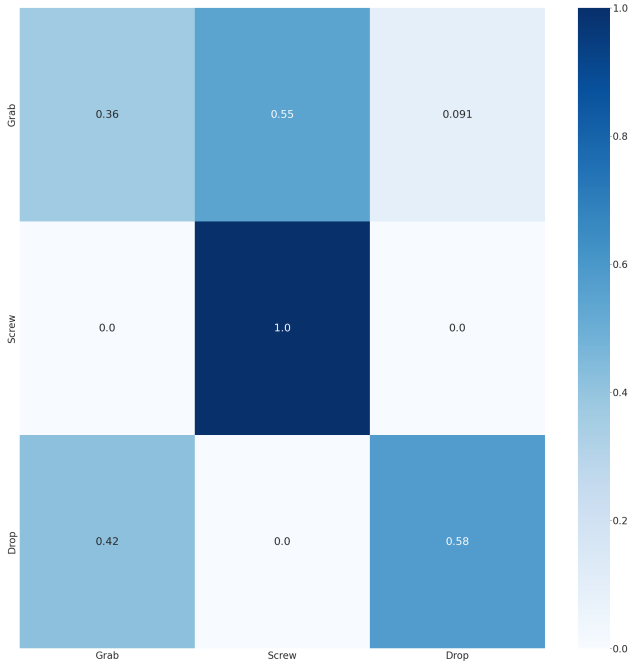

Figure 4.12: Confusion Matrix of the Live Dataset Recognition Model

Figure 4.13: Training and validation loss for the recognition model in the Live Dataset with only torso and hand keypoints available

After this, an hyperparameter tuning session was conducted in order to optimize the model parameterization and achieve the best configuration. This session, similarly to the one conducted for the industry dataset, varied the parameters according to Table 4.4. In total, 40 distinct trials were run in this first search. The best result was an accuracy on the validation data of 63,89% with a model composed of one Dense Layer and three stacked LSTM layers with 1193, 1539, and 874 units, respectively. The loss evolution for the training of this model can be seen in Fig. 4.14, its confusion matrix in Fig. 4.15 and the F1 score, recall and precision in Table 4.6

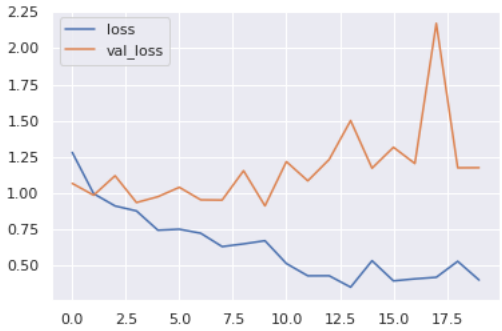

Figure 4.14: Training and validation loss for the recognition model in the Live Dataset after hyperparameter optimization

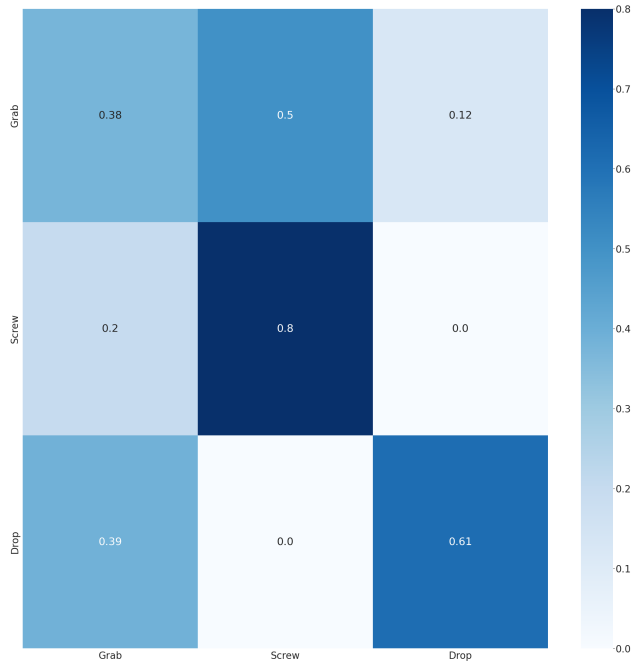

Figure 4.15: Confusion Matrix of the Live Dataset Recognition Model after hyperparameter tuning

Table 4.6: Metrics for the Recognition Model with the Live Dataset

Label	Precision	Recall	F1-Score
Grab	*0.38*	*0.25*	*0.30*
Screw	*0.80*	*0.67*	*0.73*
Drop	*0.61*	*0.92*	*0.73*
Accuracy			*0.61*
Macro Average	*0.60*	*0.61*	*0.59*
Weighted Average	*0.60*	*0.61*	*0.59*

4.3 Movement Prediction

4.3.1 Montalbano Gesture Dataset

The first version of the movement prediction network was already the output of the Keras Tuner, with the hyperparameter's intervals as described in Table 4.7.

Using its output, a model was built with one LSTM layer with 920 units and one Dense layer with the number of outputs equal to a frame's shape, 60 in this case.

Table 4.7: Hyperparameters for Bayesian Optimization Search in the Montalbano Gesture Dataset for the Movement Prediction Model

Hyperparameter	Description	Values
Number of Layers	*Number of LSTM layers*	*Integer between 1 and 6*
Units Layer I	*Number of neurons for LSTM layer I*	*Integer between 60 and 3000*
Dropout I	*Dropout rate for Dropout layer I (after each LSTM)*	*Float between 0.0 and 0.8*

The optimizer chosen was RMSprop, a gradient-based optimization technique developed as a stochastic technique for mini-batch learning. It uses a moving average of squared gradients to normalize the gradient, decreasing the step for large gradients to avoid exploding and increasing the step for small gradients to avoid vanishing. This means that the learning rate changes over time.

As a loss function, a root mean squared error function was implemented, which evaluates how far predictions fall from ground truth using euclidean distance. This is a very commonly used loss function for numerical predictions.

With this architecture, the model could predict the next frame with a base of 15 frames with the accuracy of 57,59% and the behavior seen in Fig. 4.16.

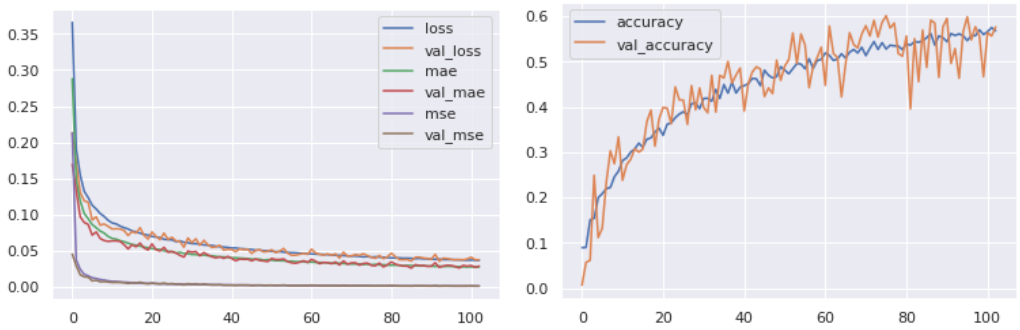

Figure 4.16: Training and validation accuracy, loss, MSE and MAE for the prediction model in the Gesture Dataset

4.3.2 Industry-Oriented Dataset

Considering the results obtained using only one LSTM layer for prediction shown in the previous section, that same architecture was tested with this dataset. For this configuration with one LSTM layer with 920 units and one Dense Layer with 81 outputs, the final accuracy of 8,14% was obtained.

Another test was made with 1127 units instead of 920, obtaining the same results. This last configuration's loss and accuracy evolution can be seen in Fig. 4.17.

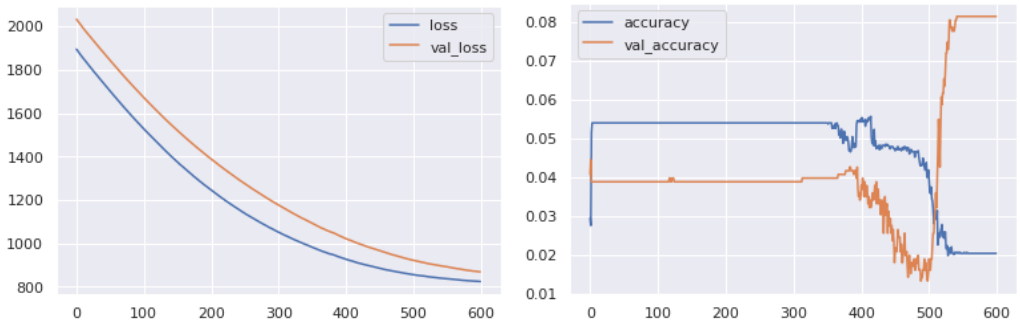

Figure 4.17: Training and validation accuracy and loss for the prediction model in the Industry Dataset

After this, it was attempted to train a more complex network. This network has 4 LSTM layers with 152, 320, 1500, and 920 units, respectively. Additionally, a Dropout layer with a 0.7 rate between layers 3 and 4 and another with a 0.2 rate between layer 4 and the Dense layer was added.

As seen in Fig. 4.18, the accuracy remained very similar throughout most of the training and ended up dropping to 1,04%, an even worse performance than the previous network.

This dataset was frequently considered very challenging to process and obtain good results from work analyzed during research. To improve the results for this type of data, other architectures based on CNNs instead of LSTMs, for example, should be considered. Considering the primary purpose of this dissertation, it was chosen to proceed with the current results instead of

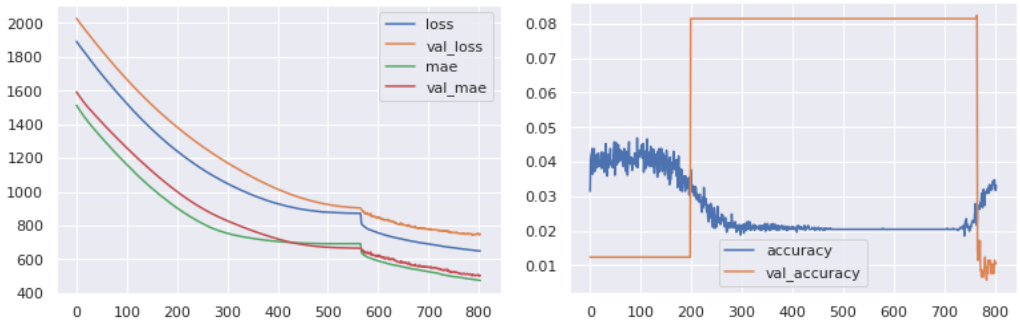

Figure 4.18: Training and validation accuracy and loss for a second prediction model in the Industry Dataset

dedicating a significant amount of effort to developing a solution designed specifically for a type of data that was not the one intended for the final solution.

4.3.3 Live Dataset

The prediction network for the live dataset brought an unusual problem. Four different architectures were trained using this data but presented similar loss behavior and the same final accuracy of 22,22%.

The first architecture had only one LSTM layer with 920 units and one Dense layer with linear activation. The optimizer was RMSProp, and the loss function was the root mean squared error.

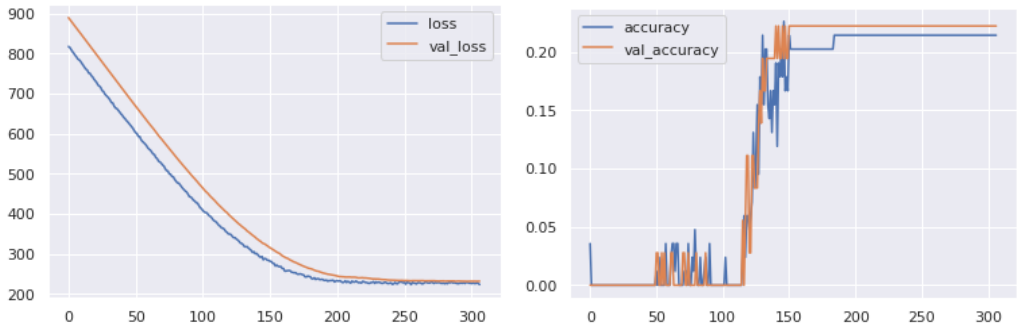

Figure 4.19: Training and validation accuracy and loss for the first prediction model in the Live Dataset

The second one had four LSTM layers with 120, 120, 120, and 3000 units, respectively. Additionally, there was a Dropout layer between layers 1 and 2, 3 and 4, and after 4 with 0.5, 0.5, and 0.8 rates. In the end, there was a Dense Layer with linear activation. The optimizer was Adam, with a learning rate of 0.0001, and MSE was the loss function.

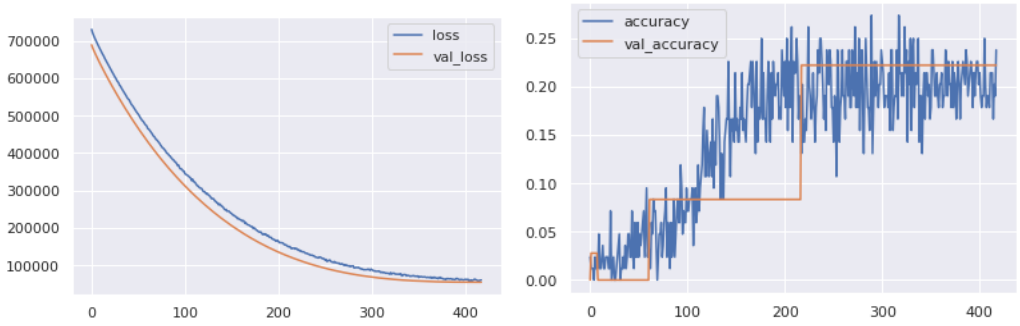

Figure 4.20: Training and validation accuracy and loss for the second prediction model in the Live Dataset

The third had a similar structure but with 4000 units instead of 3000. The learning rate for the Adam optimizer changed to 0.001, and the loss function changed to root mean squared error.

Figure 4.21: Training and validation accuracy and loss for the third prediction model in the Live Dataset

The fourth and last one also had four LSTM layers but with 920, 3000, 2500, and 920 units. The dropout layers remained the same, as well as the changes to the optimizer and loss function from the third network.

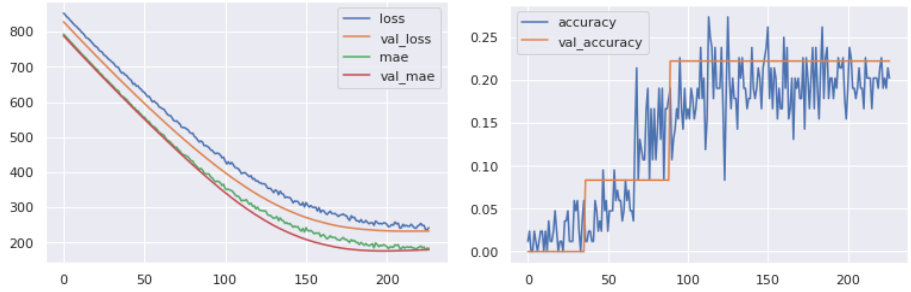

Figure 4.22: Training and validation accuracy and loss for the fourth prediction model in the Live Dataset

In an attempt to solve this issue, different approaches were tested. One of the most common

recommendations for these cases is to reduce the learning rate and remove early stopping techniques. Each approach was tested individually and simultaneously, without changing the results. Another recommendation followed without any improvement in the results was to use a callback called ReduceLROnPlateau that reduces the learning rate when some metric stops improving. Finally, the last recommendation found was regarding dataset size and quality.

Given that increasing the dataset was not possible, it was decided to experiment with different shuffles between training and validation data, given the two different environments in which data acquisition occurred. This approach was, once again, not capable of solving the problem found. It was then decided to proceed with this network to the following steps, despite not being ideal.

4.4 Combined Model

When combining the two networks, a framework with one input and two outputs was created as explained in Fig. 3.1. Multiple different approaches were tested to understand which one had more impact on the final results of the framework:

1. **Approach A**: Feeding the final hidden and cell states of the recognition network as the initial state to the prediction network.
2. **Approach B**: Feeding the final hidden and cell states of the prediction network as the initial state to the recognition network.
3. **Approach C**: Feeding the recognition network with an input that concatenates past frames with the prediction network's predictions.
4. **Approach D**: Feeding the prediction network with two distinct inputs concatenated: the frame sequence and the predicted label

In order to be able to extract the final states from one network and feed them to the other, it was necessary to transition from Keras Sequential API to their Functional API. Considering this, the network's performance might present slight variations from the previously presented in the following sections.

4.4.1 Approach A

4.4.1.1 Montalbano Gesture Dataset

The first version of the integrated framework for this dataset was obtained by combining the final networks described in Section 4.2.1 and Section 4.3.1. It achieved an accuracy of 53,84% on the recognition network and of 58,47% on the prediction network.

These first results were unexpected for the recognition component, given that they represent a significant drop in performance. Despite that, the slight increase in the prediction results was already a sign of positive influence. The loss evolution on these two output layers can be seen in

Fig. 4.23, and the evaluation metrics and confusion matrix for the recognition component can be found in Table 4.8 and Fig. 4.24, respectively.

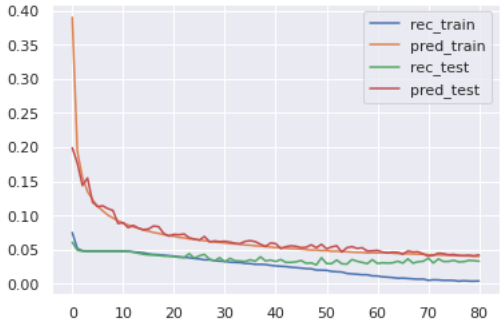

Figure 4.23: Training and validation loss on the integrated framework in the Gesture Dataset

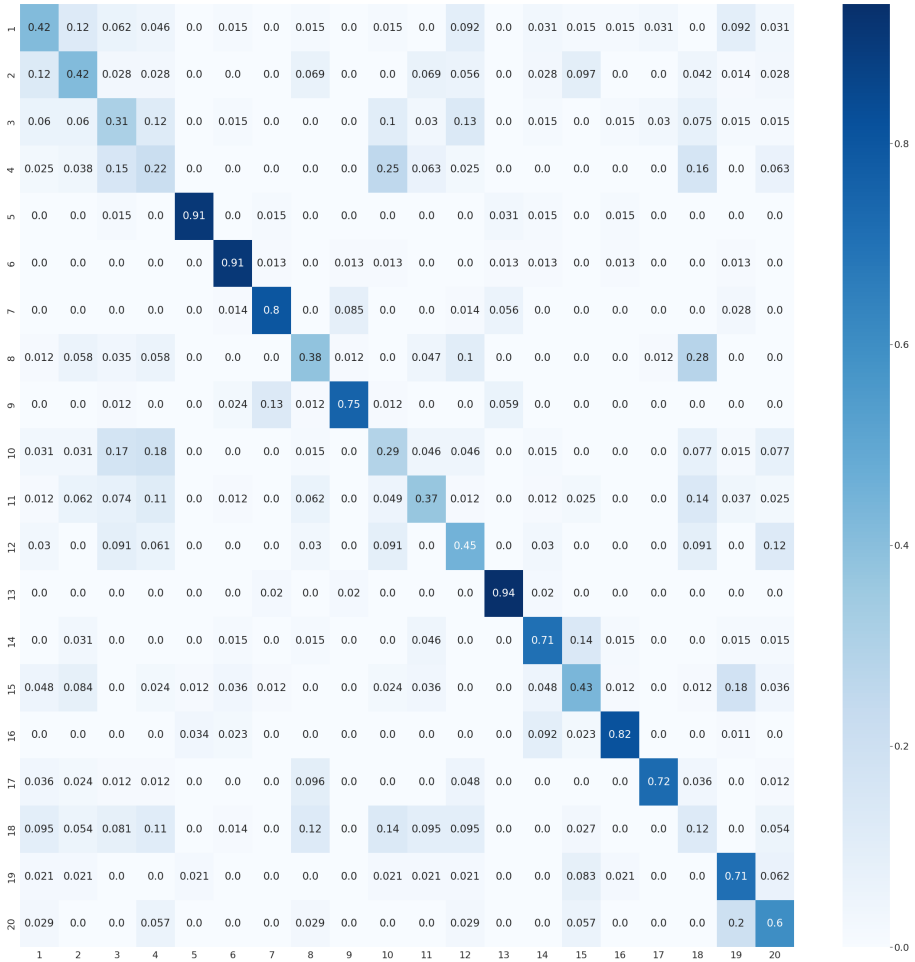

Figure 4.24: Confusion Matrix for recognition results on the integrated framework in the Gesture Dataset

After running a Bayesian Optimization Search for the recognition component of the integrated model, a new configuration was tested using 1318, 1197 and 988 units for the three LSTM layers

Table 4.8: Metrics for the Recognition Results of the Integrated Framework in Gesture Dataset

Label	Precision	Recall	F1-Score	Support
1	*0.42*	*0.43*	*0.42*	*63*
2	*0.42*	*0.41*	*0.41*	*73*
3	*0.31*	*0.30*	*0.30*	*71*
4	*0.22*	*0.24*	*0.23*	*71*
5	*0.91*	*0.92*	*0.91*	*64*
6	*0.91*	*0.84*	*0.88*	*83*
7	*0.80*	*0.79*	*0.80*	*72*
8	*0.38*	*0.50*	*0.43*	*66*
9	*0.75*	*0.88*	*0.81*	*73*
10	*0.29*	*0.28*	*0.28*	*69*
11	*0.37*	*0.48*	*0.42*	*63*
12	*0.45*	*0.24*	*0.31*	*63*
13	*0.94*	*0.79*	*0.86*	*58*
14	*0.71*	*0.67*	*0.69*	*69*
15	*0.43*	*0.55*	*0.49*	*65*
16	*0.82*	*0.91*	*0.86*	*78*
17	*0.72*	*0.92*	*0.81*	*65*
18	*0.12*	*0.12*	*0.12*	*77*
19	*0.71*	*0.47*	*0.56*	*73*
20	*0.60*	*0.39*	*0.47*	*54*
Accuracy			*0.56*	*1370*
Macro Avg.	*0.56*	*0.56*	*0.55*	*1370*
Weighted Avg.	*0.56*	*0.56*	*0.56*	*1370*

for recognition and 988 units for the LSTM layer responsible for prediction. With this architecture, the model had an accuracy of 74,96% in recognition and an 80,51% accuracy in prediction, with an evolution as can be seen in Fig. 4.25. The loss and MAE evolution can be seen in Fig. 4.26. The confusion matrix in Fig. 4.27 and the Recall, Precision and F1-Score in Table 4.9

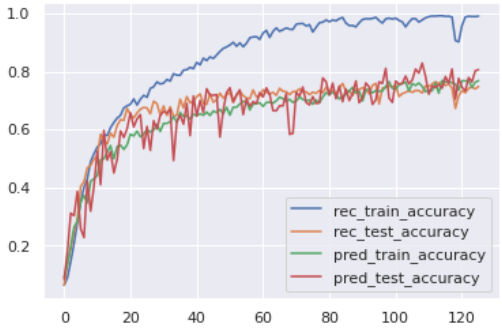

Figure 4.25: Training and validation accuracy for the integrated framework in the Gesture Dataset after hyperparameter tuning

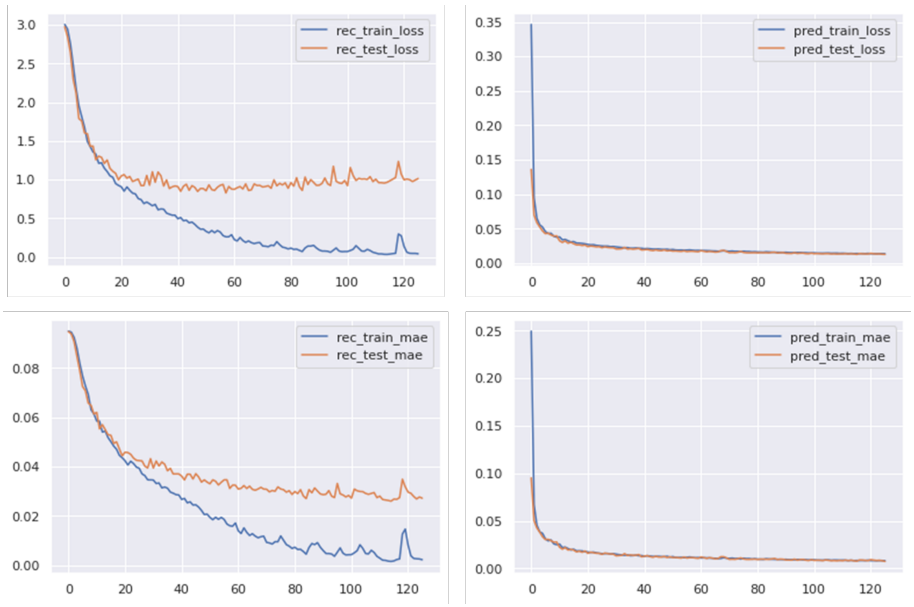

Figure 4.26: Training and validation loss and MAE for the integrated framework in the Gesture Dataset after hyperparameter tuning

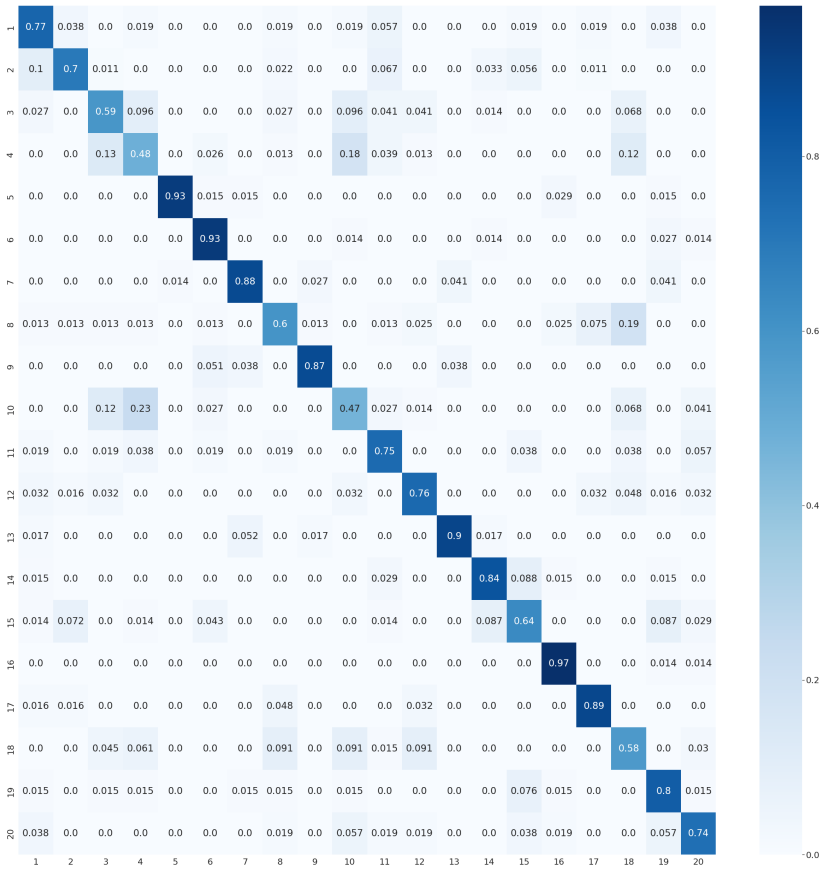

Figure 4.27: Confusion Matrix for recognition results on the integrated framework in the Gesture Dataset after hyperparameter tuning

Table 4.9: Metrics for the Recognition Results of the Integrated Framework in Gesture Dataset after hyperparameter tuning

Label	Precision	Recall	F1-Score	Support
1	*0.77*	*0.65*	*0.71*	*63*
2	*0.70*	*0.86*	*0.77*	*73*
3	*0.59*	*0.61*	*0.60*	*71*
4	*0.48*	*0.52*	*0.50*	*71*
5	*0.93*	*0.98*	*0.95*	*64*
6	*0.93*	*0.83*	*0.88*	*83*
7	*0.88*	*0.89*	*0.88*	*72*
8	*0.60*	*0.73*	*0.66*	*66*
9	*0.87*	*0.95*	*0.91*	*73*
10	*0.47*	*0.49*	*0.48*	*69*
11	*0.75*	*0.63*	*0.69*	*63*
12	*0.76*	*0.75*	*0.75*	*63*
13	*0.90*	*0.90*	*0.90*	*58*
14	*0.84*	*0.83*	*0.83*	*69*
15	*0.64*	*0.68*	*0.66*	*65*
16	*0.97*	*0.91*	*0.94*	*78*
17	*0.89*	*0.85*	*0.87*	*65*
18	*0.58*	*0.49*	*0.53*	*77*
19	*0.80*	*0.73*	*0.76*	*73*
20	*0.74*	*0.72*	*0.73*	*54*
Accuracy			*0.75*	*1370*
Macro Avg.	*0.75*	*0.75*	*0.75*	*1370*
Weighted Avg.	*0.75*	*0.75*	*0.75*	*1370*

4.4.1.2 Industry-oriented Dataset

For this first approach, an architecture with 3 LSTM layers with 1193, 1539, and 988 units for recognition and one LSTM layer with 988 units for prediction was defined. Additionally, there were two Dense layers, one for label output and one for frame prediction output.

Before testing with this configuration, a test was run where the 3 LSTM layers had 3000, 3000, and 129 units, just as described in Section 4.2.2. However, this network took much more time to train, and, after completion, the final performance was the same as the one proposed here. For this reason, it was chosen to use a lighter network, optimizing computational times.

This configuration obtained an accuracy of 30,87% for recognition and 8,14% for prediction. The accuracy evolution for both networks can be seen in Fig. 4.28, the loss and MAE evolution in Fig. 4.29, the F1-score, precision and recall in Table 4.10, and the confusion matrix in Fig. 4.30.

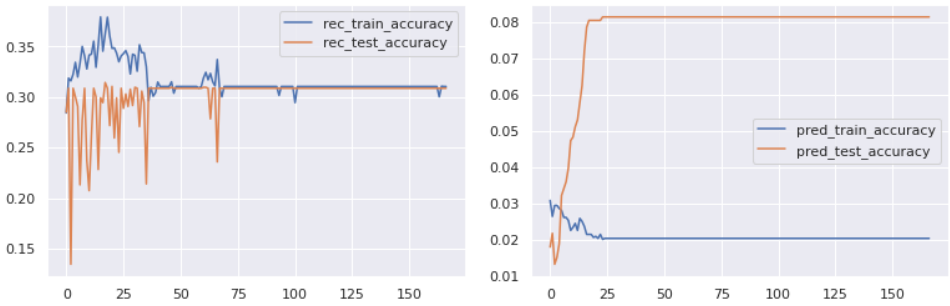

Figure 4.28: Training and validation accuracy for the integrated framework in the Industry Dataset

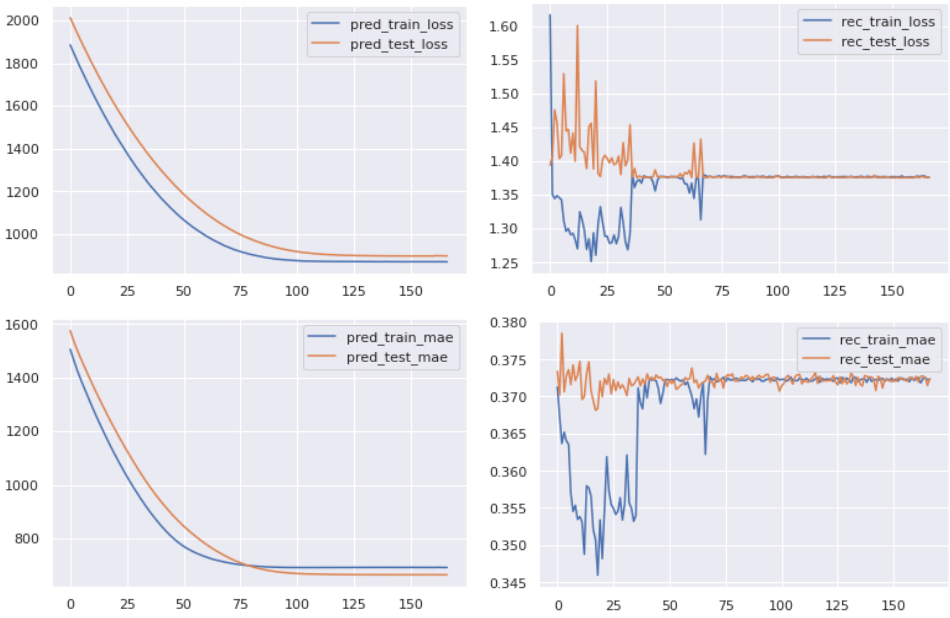

Figure 4.29: Training and validation loss and MAE for the integrated framework in the Industry Dataset

Table 4.10: Metrics for the Recognition Model with the Industry Dataset

Label	**Precision**	**Recall**	**F1-Score**	**Support**
Pi	*0.00*	*0.00*	*0.00*	*226*
Pl	*0.00*	*0.00*	*0.00*	*232*
Rl	*0.00*	*0.00*	*0.00*	*272*
Fm	*0.31*	*1.00*	*0.47*	*326*
Accuracy			*0.31*	*1056*
Macro Avg.	*0.08*	*0.25*	*0.12*	*1056*
Weighted Avg.	*0.10*	*0.31*	*0.15*	*1056*

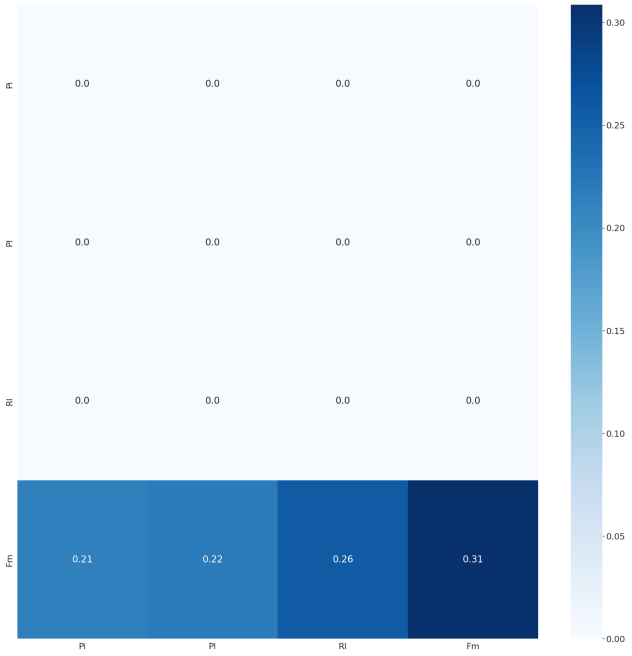

Figure 4.30: Confusion Matrix for recognition results on the integrated framework in the Industry Dataset

4.4.1.3 Live Dataset

For this first approach, an architecture with 3 LSTM layers with 100, 100, and 516 units for recognition and one LSTM layer with 516 units for prediction was defined. Additionally, there were two Dense layers, one for label output and one for frame prediction output.

This configuration obtained an accuracy of 52,78% for recognition and 25% for prediction. The accuracy evolution for both networks can be seen in Fig. 4.31, the loss and MAE evolution in Fig. 4.32, the F1-score, precision and recall in Table 4.11, and the confusion matrix in Fig. 4.33.

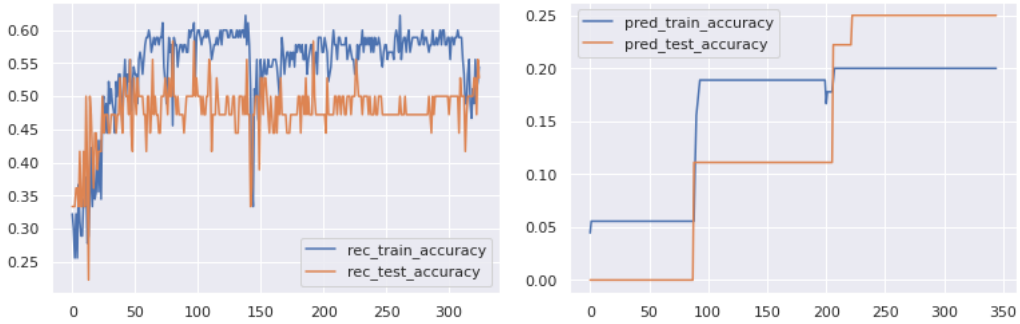

Figure 4.31: Training and validation accuracy for the integrated framework in the Live Dataset

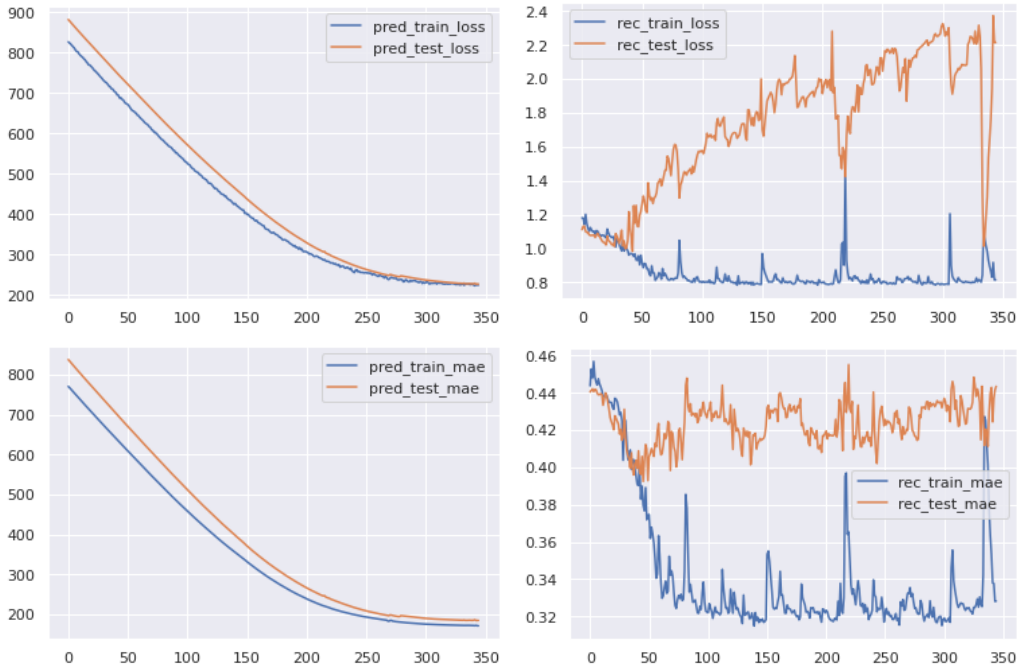

Figure 4.32: Training and validation loss and MAE for the integrated framework in the Live Dataset

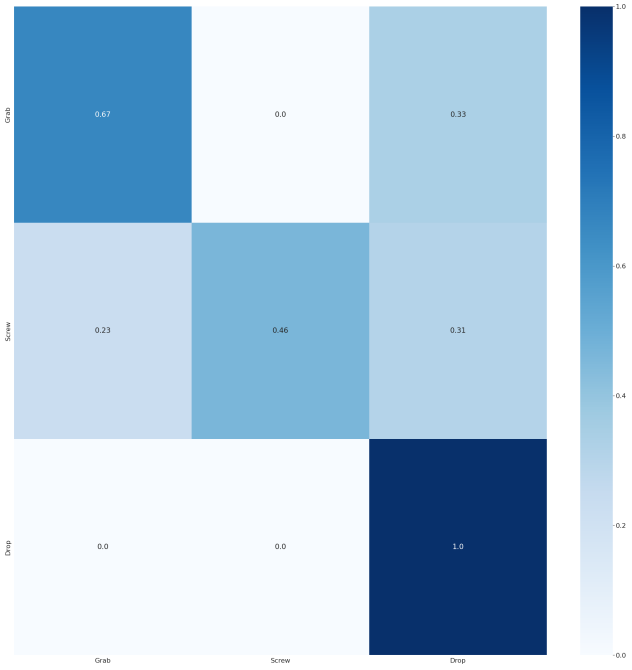

Figure 4.33: Confusion Matrix for recognition results on the integrated framework in the Live Dataset

Table 4.11: Metrics for the Recognition Model with the Live Dataset

Label	Precision	Recall	F1-Score
Grab	*0.67*	*0.50*	*0.57*
Screw	*0.46*	*1.00*	*0.63*
Drop	*1.00*	*0.08*	*0.15*
Accuracy			*0.53*
Macro Average	*0.71*	*0.53*	*0.45*
Weighted Average	*0.71*	*0.53*	*0.45*

4.4.2 Approach B

4.4.2.1 Montalbano Gesture Dataset

For this approach, the architecture from Section 4.4.1.1 was inverted by starting to train the prediction network and feeding the final states as initial states for the recognition network. The number of units in the prediction network had to be changed from 988 to 1318 to assure compatibility. The recognition network, the optimizer, and loss functions remained unchanged. This configuration reached an accuracy of 73,58% in recognition and 75,33% in prediction. The lack of impact in the recognition performance might be due to the fact that the prediction was only for one frame and the network has only one layer, so the impact on the states might not be significant. The loss and MAE evolution can be seen in Fig. 4.34, confusion matrix in Fig. 4.36, accuracy in Fig. 4.35, and metrics in Table 4.12.

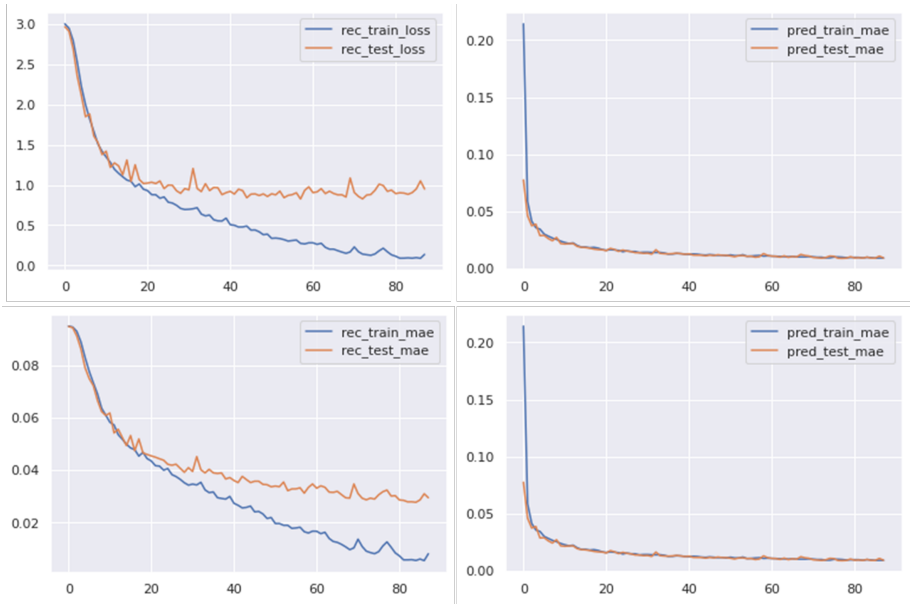

Figure 4.34: Training and validation loss and MAE for the integrated framework in the Gesture Dataset using prediction final states to initialize recognition network

Table 4.12: Metrics for the Integrated Framework in Gesture Dataset when using prediction LSTM states to initialize recognition

Label	**Precision**	**Recall**	**F1-Score**	**Support**
1	*0.76*	*0.71*	*0.74*	*63*
2	*0.71*	*0.71*	*0.71*	*73*
3	*0.54*	*0.55*	*0.55*	*71*
4	*0.43*	*0.39*	*0.41*	*71*
5	*0.91*	*0.98*	*0.95*	*64*
6	*0.96*	*0.81*	*0.88*	*83*
7	*0.86*	*0.89*	*0.88*	*72*
8	*0.58*	*0.68*	*0.62*	*66*
9	*0.87*	*0.90*	*0.89*	*73*
10	*0.47*	*0.54*	*0.50*	*69*
11	*0.64*	*0.67*	*0.65*	*63*
12	*0.75*	*0.65*	*0.69*	*63*
13	*0.90*	*0.90*	*0.90*	*58*
14	*0.80*	*0.83*	*0.81*	*69*
15	*0.57*	*0.75*	*0.65*	*65*
16	*0.96*	*0.94*	*0.95*	*78*
17	*0.86*	*0.85*	*0.85*	*65*
18	*0.61*	*0.57*	*0.59*	*77*
19	*0.88*	*0.73*	*0.80*	*73*
20	*0.75*	*0.67*	*0.71*	*54*
Accuracy			*0.74*	*1370*
Macro Avg.	*0.74*	*0.74*	*0.74*	*1370*
Weighted Avg.	*0.74*	*0.74*	*0.74*	*1370*

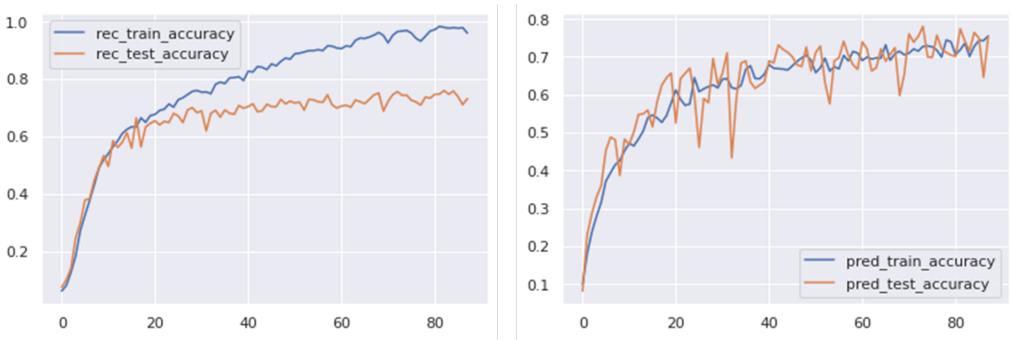

Figure 4.35: Training and validation accuracy for the integrated framework in the Gesture Dataset using prediction final states to initialize recognition network

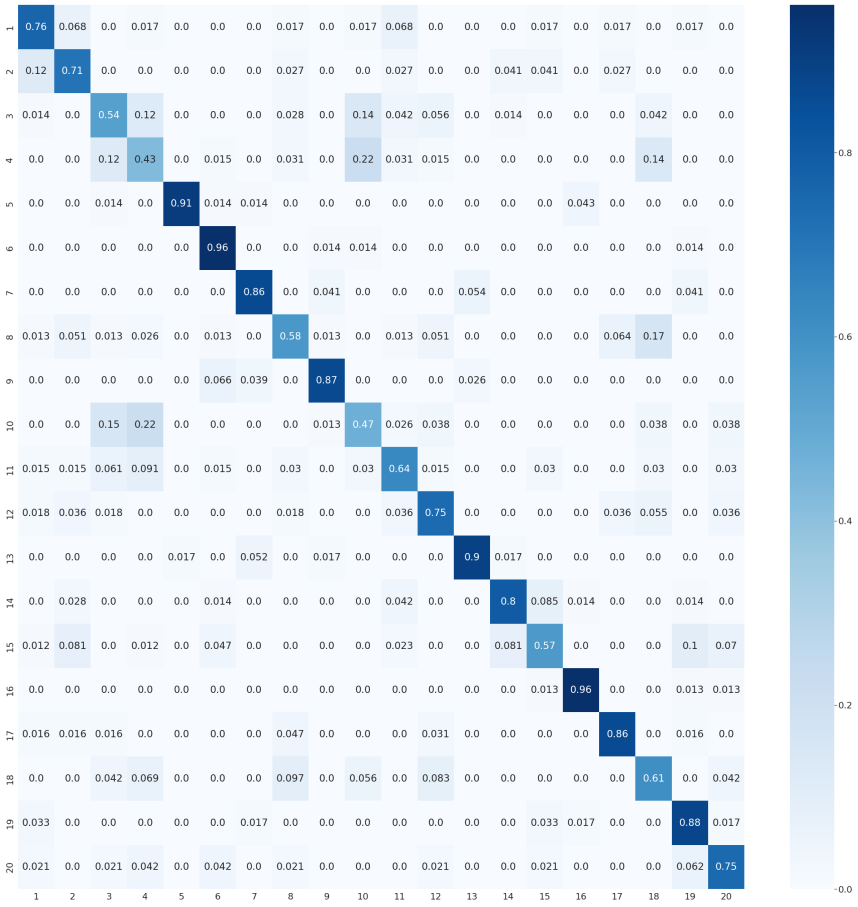

Figure 4.36: Confusion Matrix for the integrated framework in the Gesture Dataset using prediction final states to initialize recognition network

4.4.2.2 Industry-oriented Dataset

In this approach, the architecture from Section 4.4.1.2 was inverted having an initial LSTM layer with 1193 units for prediction. This configuration obtained an accuracy of 30,87% for recognition and 8,14% for prediction. The accuracy evolution can be seen in Fig. 4.37, the confusion matrix in Fig. 4.38, the F1-score, precision and recall in Table 4.13, and the loss and MAE in Fig. 4.39.

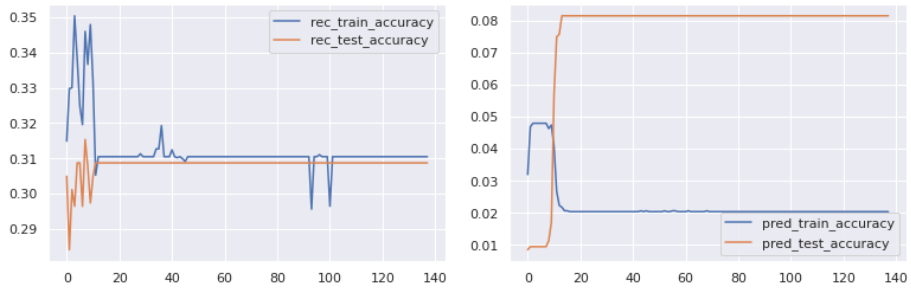

Figure 4.37: Training and validation accuracy for the integrated framework in the Industry Dataset using prediction states to initialize recognition

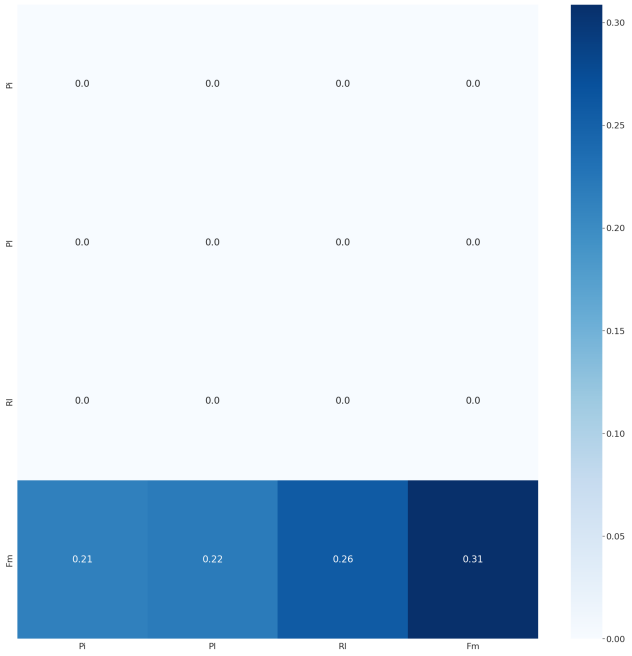

Figure 4.38: Confusion Matrix for recognition results on the integrated framework in the Industry Dataset using prediction states to initialize recognition

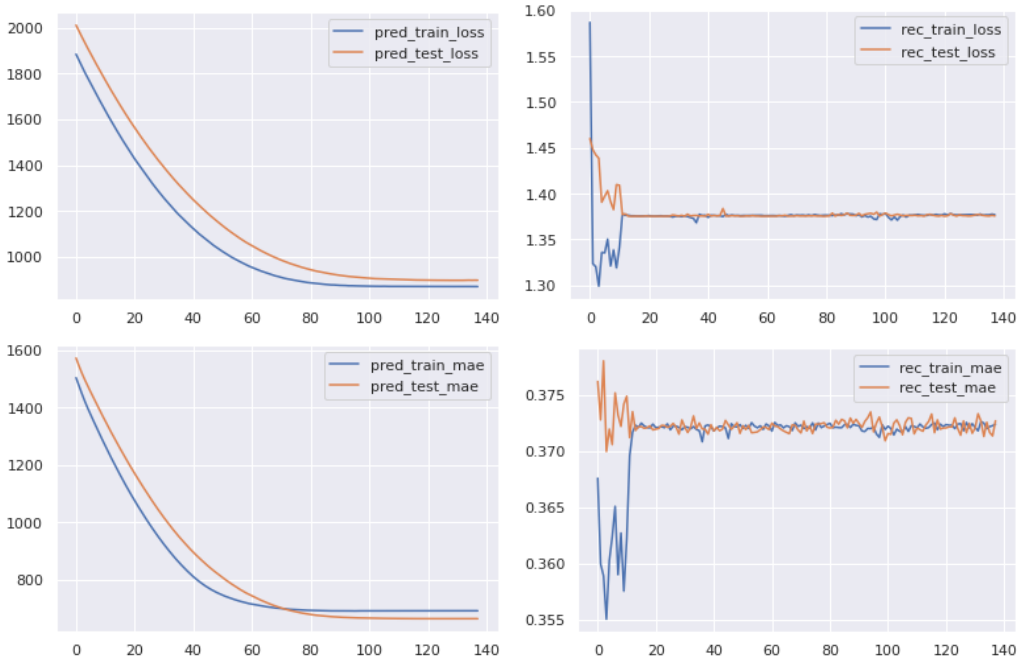

Figure 4.39: Training and validation loss and MAE for the integrated framework in the Industry Dataset using prediction states to initialize recognition

Table 4.13: Metrics for the Recognition Model with the Industry Dataset using prediction states to initialize recognition

Label	Precision	Recall	F1-Score	Support
Pi	*0.00*	*0.00*	*0.00*	*226*
Pl	*0.00*	*0.00*	*0.00*	*232*
Rl	*0.00*	*0.00*	*0.00*	*272*
Fm	*0.31*	*1.00*	*0.47*	*326*
Accuracy			*0.31*	*1056*
Macro Avg.	*0.08*	*0.25*	*0.12*	*1056*
Weighted Avg.	*0.10*	*0.31*	*0.15*	*1056*

4.4.2.3 Live Dataset

In this second approach, the architecture from Section 4.4.1.3 was inverted, starting by an LSTM layer with 100 units for prediction and then the 3 LSTM layers with 100, 100 and 516 units each. With this configuration, the model achieved an accuracy of 50,00% for recognition and 22,22% for prediction. The accuracy evolution for both networks can be seen in Fig. 4.41, the loss and MAE evolution in Fig. 4.40, the F1-score, precision and recall in Table 4.14, and the confusion matrix in Fig. 4.42.

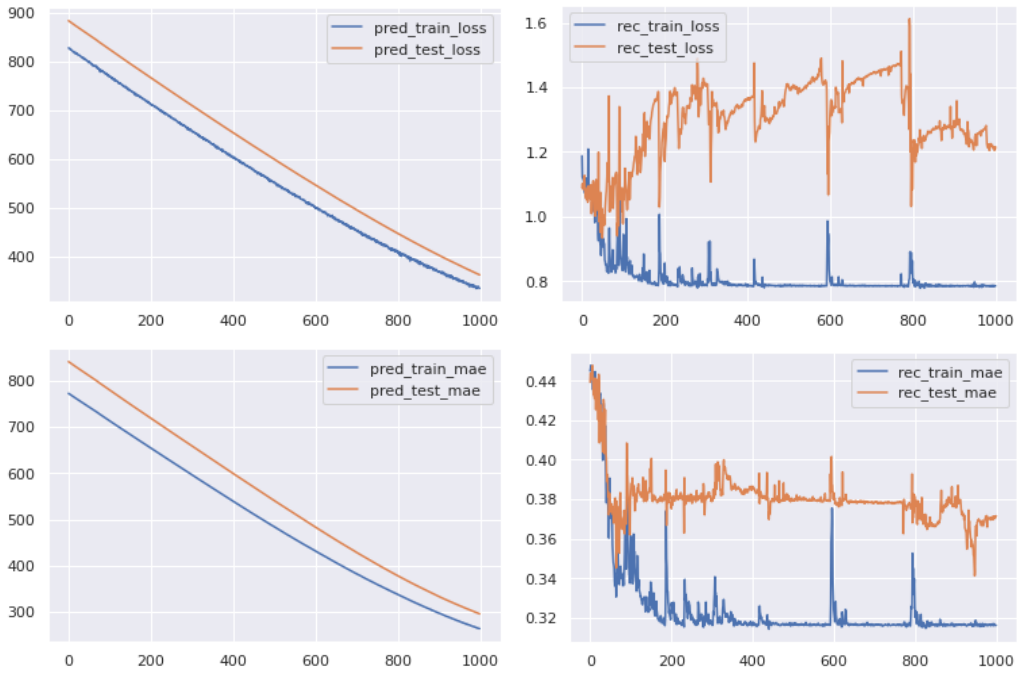

Figure 4.40: Training and validation loss and MAE for the integrated framework in the Live Dataset using prediction states to initialize recognition

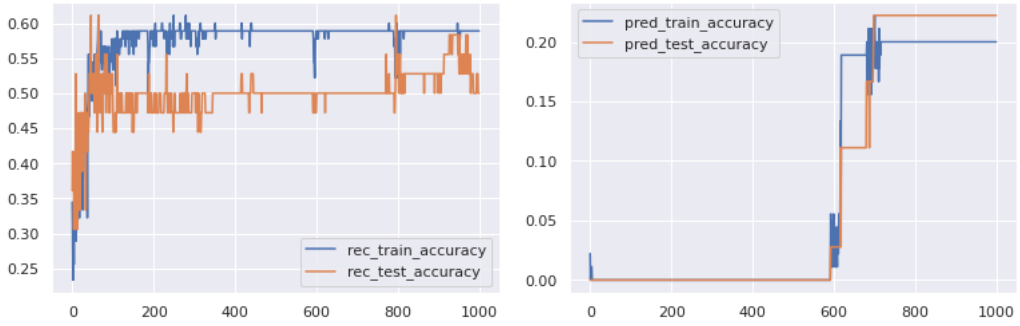

Figure 4.41: Training and validation accuracy for the integrated framework in the Live Dataset using prediction states to initialize recognition

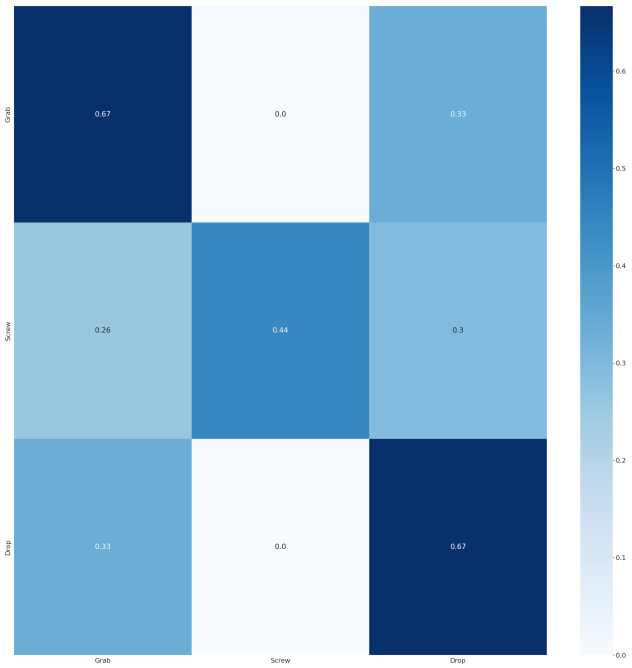

Figure 4.42: Confusion Matrix for recognition results on the integrated framework in the Live Dataset using prediction states to initialize recognition

Table 4.14: Metrics for the Live Dataset using prediction states to initialize recognition

Label	**Precision**	**Recall**	**F1-Score**
Grab	*0.67*	*0.33*	*0.44*
Screw	*0.44*	*1.00*	*0.62*
Drop	*0.67*	*0.17*	*0.27*
Accuracy			*0.50*
Macro Average	*0.59*	*0.50*	*0.44*
Weighted Average	*0.59*	*0.50*	*0.44*

4.4.3 Approach C

4.4.3.1 Montalbano Gesture Dataset

In this approach, the network was maintained exactly as in the previous step but instead of using hidden and cell states, the predicted frame was concatenated with the initial input and fed to the recognition network.

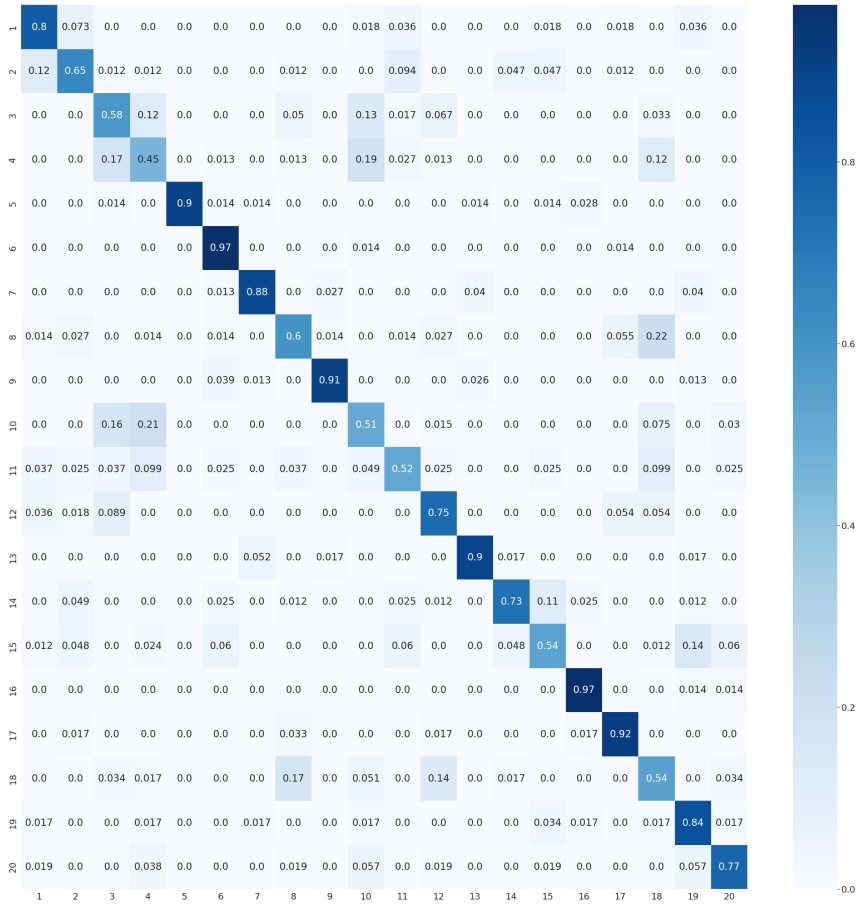

Figure 4.43: Confusion Matrix for the integrated framework in the Gesture Dataset when concatenating predicted frame to input for recognition

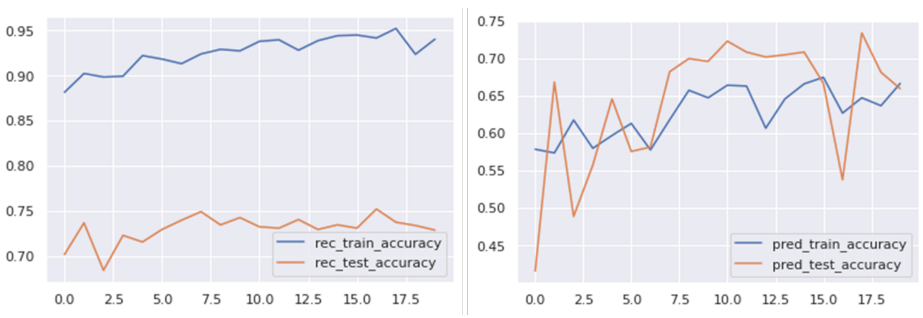

Figure 4.44: Training and validation accuracy for the integrated framework in the Gesture Dataset when concatenating predicted frame to input for recognition

In this experiment, the prediction network achieved 65,91% accuracy whilst the recognition network achieved 72,85%. The metrics can be seen in Table 4.15, loss in Fig. 4.45, confusion matrix in Fig. 4.43, and accuracy in Fig. 4.44.

Table 4.15: Metrics for the Integrated Framework in Gesture Dataset when concatenating predicted frame to input for recognition

Label	**Precision**	**Recall**	**F1-Score**	**Support**
1	*0.80*	*0.70*	*0.75*	*63*
2	*0.65*	*0.75*	*0.70*	*73*
3	*0.58*	*0.49*	*0.53*	*71*
4	*0.45*	*0.48*	*0.47*	*71*
5	*0.90*	*1.00*	*0.95*	*64*
6	*0.97*	*0.81*	*0.88*	*83*
7	*0.88*	*0.92*	*0.90*	*72*
8	*0.60*	*0.67*	*0.63*	*66*
9	*0.91*	*0.95*	*0.93*	*73*
10	*0.51*	*0.49*	*0.50*	*69*
11	*0.52*	*0.67*	*0.71*	*63*
12	*0.75*	*0.67*	*0.71*	*63*
13	*0.90*	*0.90*	*0.90*	*58*
14	*0.73*	*0.86*	*0.79*	*69*
15	*0.54*	*0.69*	*0.60*	*65*
16	*0.97*	*0.92*	*0.95*	*78*
17	*0.92*	*0.85*	*0.88*	*65*
18	*0.54*	*0.42*	*0.47*	*77*
19	*0.84*	*0.67*	*0.75*	*73*
20	*0.77*	*0.76*	*0.77*	*54*
Accuracy			*0.73*	*1370*
Macro Avg.	*0.74*	*0.73*	*0.73*	*1370*
Weighted Avg.	*0.74*	*0.73*	*0.73*	*1370*

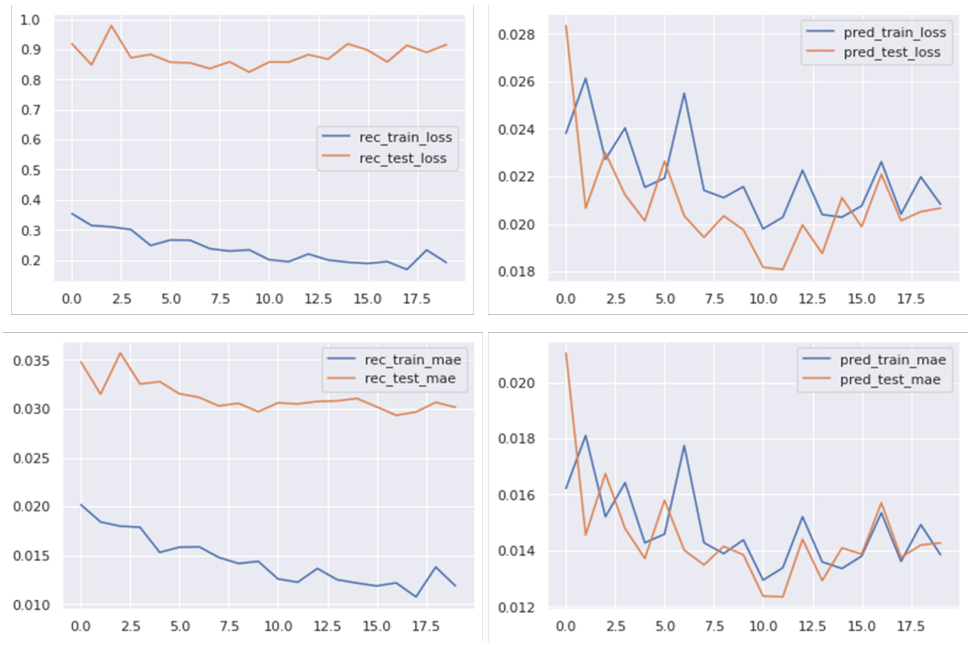

Figure 4.45: Training and validation loss and MAE for the integrated framework in the Gesture Dataset when concatenating predicted frame to input for recognition

4.4.3.2 Industry-oriented Dataset

In this third approach, the architecture from Section 4.4.1.2 was inverted, having an initial LSTM layer with 1193 units for prediction and then 3 LSTM layers with 1193, 1539, and 988 units for recognition. This configuration obtained an accuracy of 30,87% for recognition and 8,14% for prediction.

The accuracy evolution for both networks can be seen in Fig. 4.48, the loss and MAE evolution in Fig. 4.46, the F1-score, precision and recall in Table 4.16, and the confusion matrix in Fig. 4.47.

Table 4.16: Metrics for the Recognition Model with the Industry Dataset concatenating predicted frame with input frames

Label	**Precision**	**Recall**	**F1-Score**	**Support**
Pi	*0.00*	*0.00*	*0.00*	*226*
Pl	*0.47*	*0.04*	*0.07*	*232*
Rl	*0.00*	*0.00*	*0.00*	*272*
Fm	*0.30*	*0.97*	*0.46*	*326*
Accuracy			*0.31*	*1056*
Macro Avg.	*0.19*	*0.25*	*0.13*	*1056*
Weighted Avg.	*0.20*	*0.31*	*0.16*	*1056*

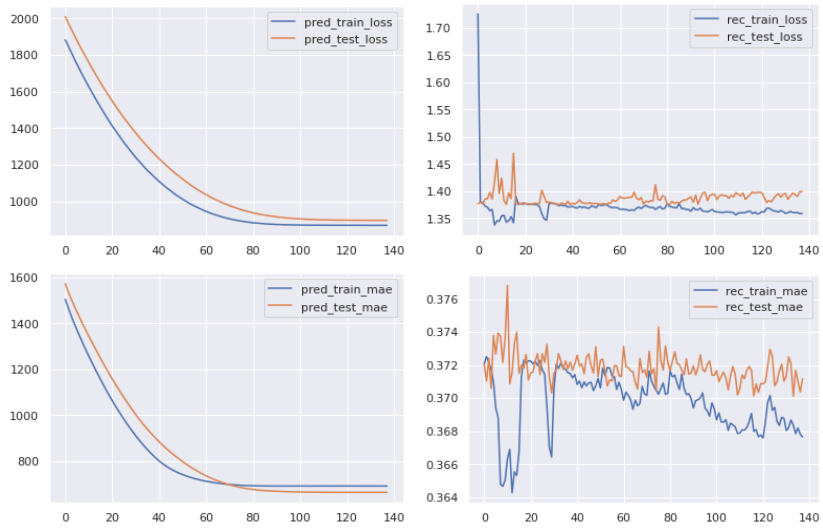

Figure 4.46: Training and validation loss and MAE for the integrated framework in the Industry Dataset concatenating predicted frame with input frames

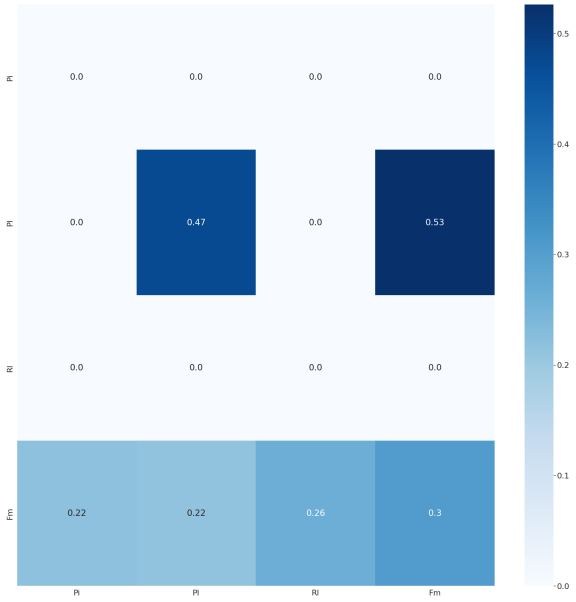

Figure 4.47: Confusion Matrix in the Industry Dataset concatenating predicted frame with input

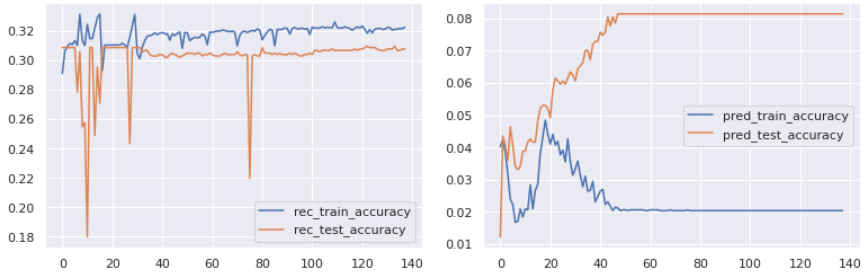

Figure 4.48: Accuracy in the Industry Dataset concatenating predicted frame with input

4.4.3.3 Live Dataset

The architecture for this approach was exactly the one described in Section 4.4.2.3 but instead of using hidden and cell states, the actual predicted frame was concatenated with the input frames and fed to the recognition network. With a prediction accuracy of 22,22% the recognition accuracy was 41,67%. The accuracy evolution for both networks can be seen in Fig. 4.50, the loss and MAE evolution in Fig. 4.51, the F1-score, precision and recall in Table 4.17, and the confusion matrix in Fig. 4.49.

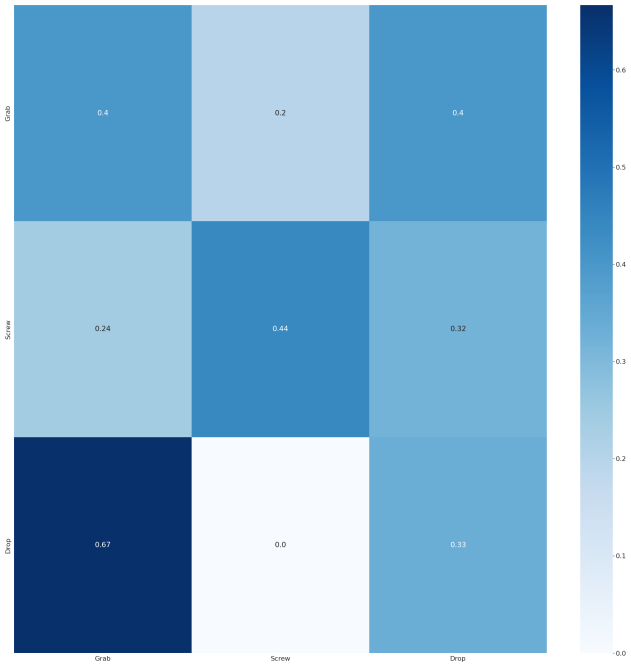

Figure 4.49: Confusion Matrix for recognition results on the integrated framework in the Live Dataset concatenating predicted frame with input frames

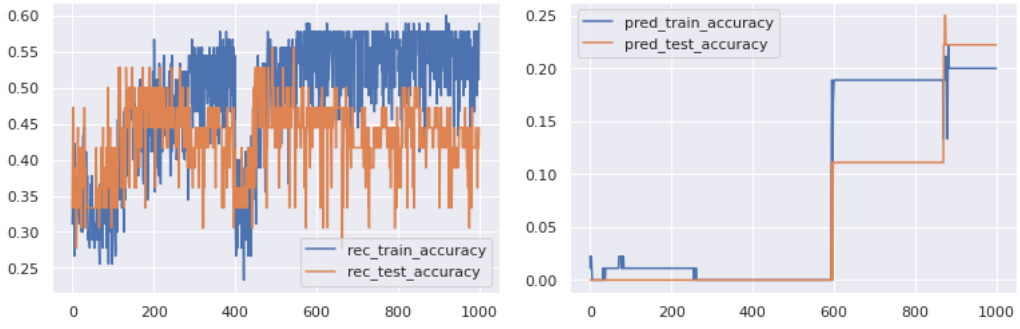

Figure 4.50: Training and validation accuracy for the integrated framework in the Live Dataset concatenating predicted frame with input frames

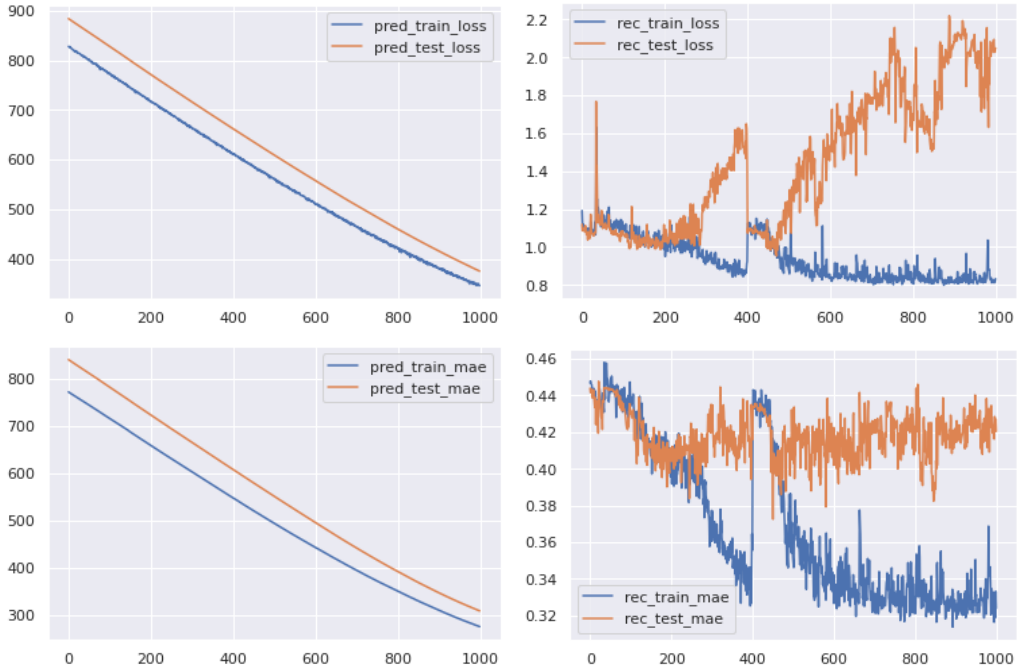

Figure 4.51: Training and validation loss and MAE for the integrated framework in the Live Dataset concatenating predicted frame with input frames

Table 4.17: Metrics for the Recognition Model with the Live Dataset concatenating predicted frame with input frames

Label	**Precision**	**Recall**	**F1-Score**
Grab	*0.40*	*0.17*	*0.24*
Screw	*0.44*	*0.92*	*0.59*
Drop	*0.33*	*0.17*	*0.22*
Accuracy			*0.42*
Macro Average	*0.39*	*0.42*	*0.35*
Weighted Average	*0.39*	*0.42*	*0.35*

4.4.4 Approach D

4.4.4.1 Montalbano Gesture Dataset

For this approach, the architecture from the first one was recovered, but instead of using final states to initialize the prediction network, we concatenated the label with the frames input. With this configuration, the recognition network obtained a 72,85% accuracy, and the prediction obtained an accuracy of 84,23%. The confusion matrix can be seen in Fig. 4.52, and the metrics in Table 4.18.

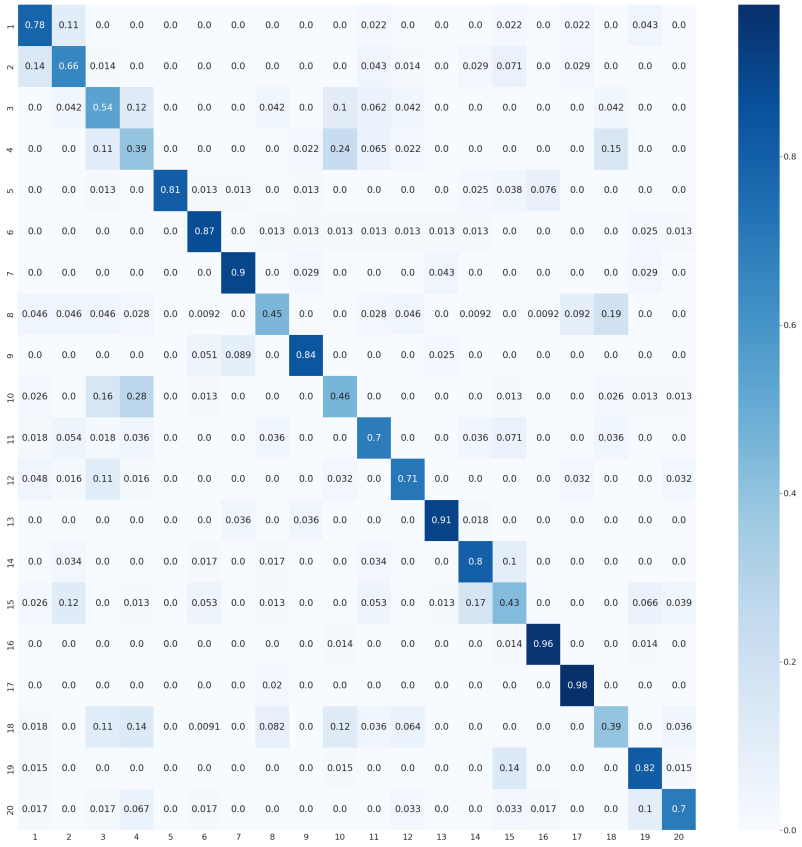

Figure 4.52: Confusion Matrix for the integrated framework in the Gesture Dataset concatenating recognition label to the prediction network input

4.4.4.2 Industry Dataset

In this fourth approach, the initial architecture from Section 4.4.1.2 was reproduced but concatenating the label to the input frames instead of using states. This time the recognition network had an accuracy of 30,87% and the prediction of 8,14%. The accuracy evolution for both networks can be seen in Fig. 4.53, the loss and MAE evolution in Fig. 4.54, the F1-score, precision and recall in Table 4.19, and the confusion matrix in Fig. 4.55.

Figure 4.53: Training and validation accuracy for the integrated framework in the Industry Dataset concatenating predicted label with input frames

Table 4.18: Metrics for the Integrated Framework in Gesture Dataset when using prediction LSTM states to initialize recognition

Label	**Precision**	**Recall**	**F1-Score**	**Support**
1	*0.78*	*0.57*	*0.66*	*63*
2	*0.66*	*0.63*	*0.64*	*73*
3	*0.54*	*0.37*	*0.44*	*71*
4	*0.39*	*0.25*	*0.31*	*71*
5	*0.81*	*1.00*	*0.90*	*64*
6	*0.87*	*0.83*	*0.85*	*83*
7	*0.90*	*0.86*	*0.88*	*72*
8	*0.45*	*0.74*	*0.56*	*66*
9	*0.84*	*0.90*	*0.87*	*73*
10	*0.46*	*0.51*	*0.48*	*69*
11	*0.70*	*0.62*	*0.66*	*63*
12	*0.71*	*0.70*	*0.70*	*63*
13	*0.91*	*0.88*	*0.89*	*58*
14	*0.80*	*0.68*	*0.73*	*69*
15	*0.43*	*0.51*	*0.47*	*65*
16	*0.96*	*0.90*	*0.93*	*78*
17	*0.98*	*0.77*	*0.86*	*65*
18	*0.39*	*0.56*	*0.46*	*77*
19	*0.82*	*0.74*	*0.78*	*73*
20	*0.70*	*0.78*	*0.74*	*54*
Accuracy			*0.69*	*1370*
Macro Avg.	*0.70*	*0.69*	*0.69*	*1370*
Weighted Avg.	*0.70*	*0.69*	*0.69*	*1370*

Table 4.19: Metrics for the Recognition Model with the Industry Dataset concatenating predicted label with input frames

Label	**Precision**	**Recall**	**F1-Score**	**Support**
Pi	*0.00*	*0.00*	*0.00*	*226*
Pl	*0.00*	*0.00*	*0.00*	*232*
Rl	*0.00*	*0.00*	*0.00*	*272*
Fm	*0.31*	*1.00*	*0.47*	*326*
Accuracy			*0.31*	*1056*
Macro Avg.	*0.08*	*0.25*	*0.12*	*1056*
Weighted Avg.	*0.10*	*0.31*	*0.15*	*1056*

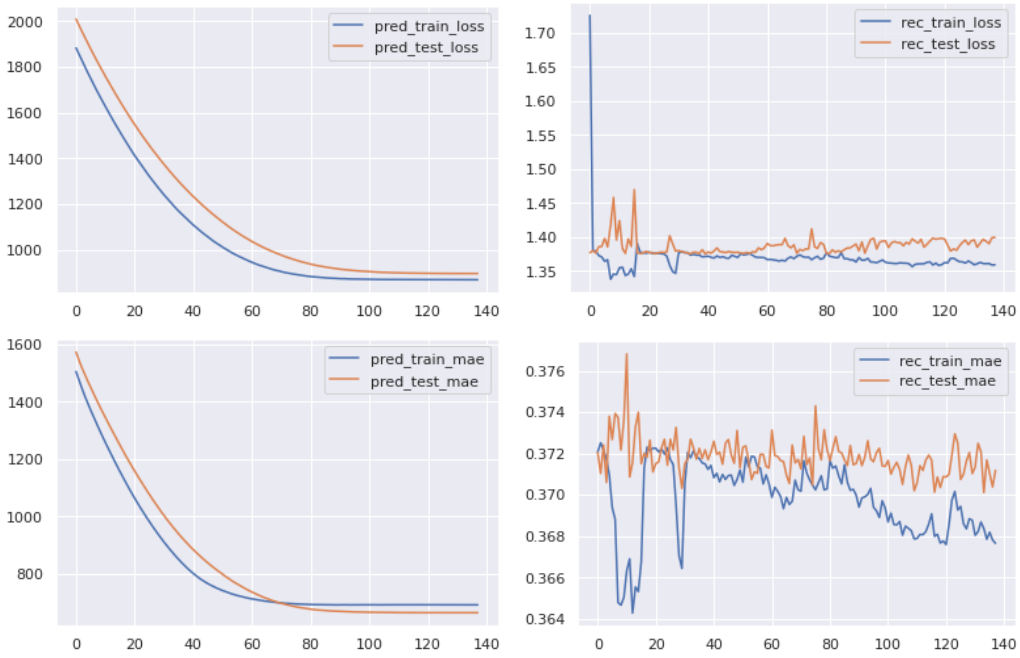

Figure 4.54: Training and validation loss and MAE for the integrated framework in the Industry Dataset concatenating predicted label with input frames

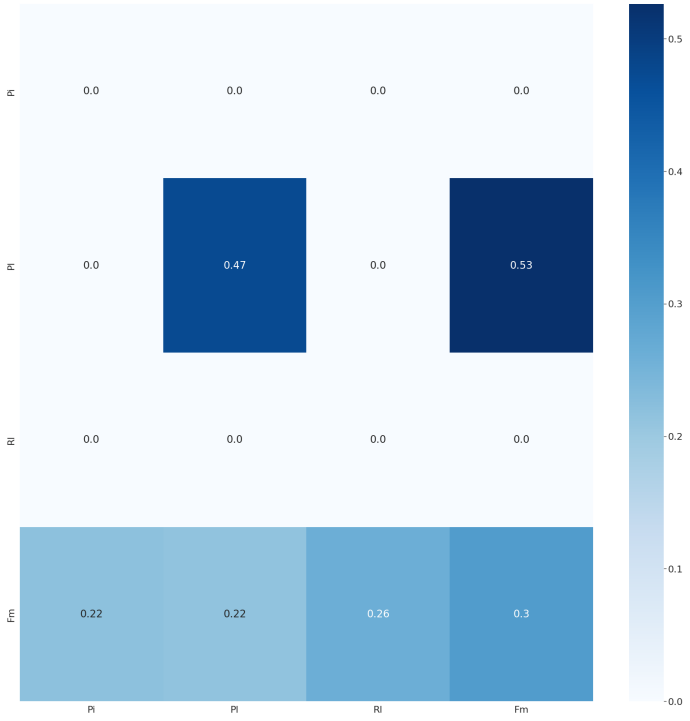

Figure 4.55: Confusion Matrix for recognition results on the integrated framework in the Industry Dataset concatenating predicted label with input frames

4.4.4.3 Live Dataset

For this last approach, the architecture from Section 4.4.1.3 was recovered but instead of using the hidden and cell states, the label predicted was concatenated to the input frames and fed to the prediction network. This time the recognition network had an accuracy of 44,44% and the prediction of 25%. The accuracy evolution for both networks can be seen in Fig. 4.56, the loss and MAE evolution in Fig. 4.57, the F1-score, precision and recall in Table 4.20, and the confusion matrix in Fig. 4.58.

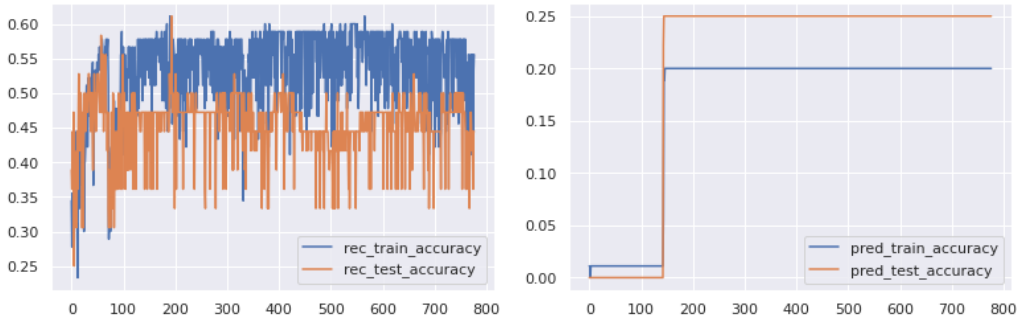

Figure 4.56: Training and validation accuracy for the integrated framework in the Live Dataset concatenating predicted label with input frames

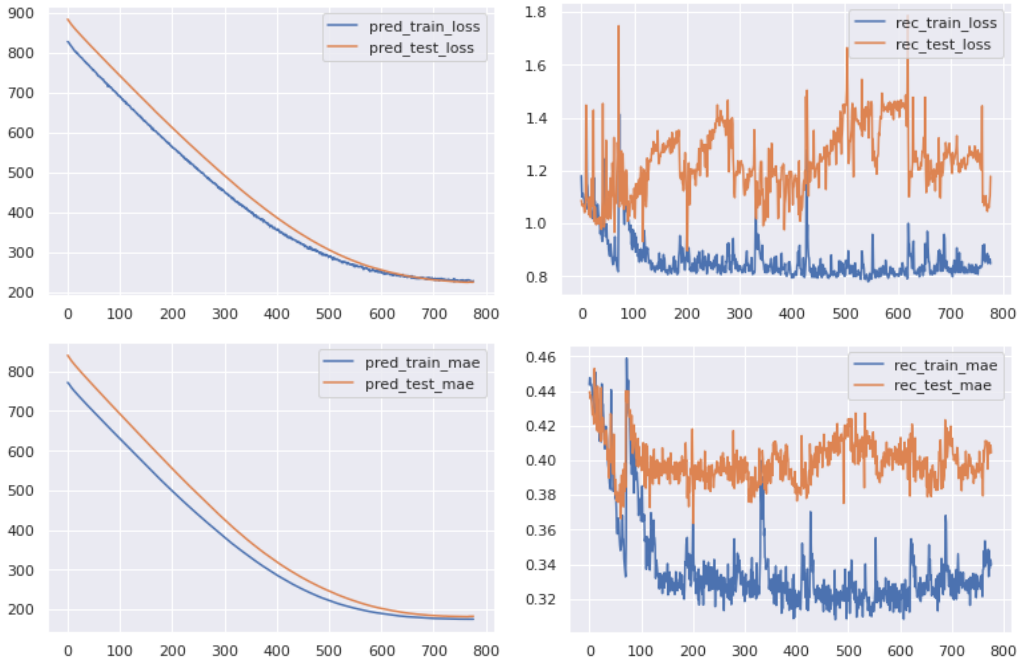

Figure 4.57: Training and validation loss and MAE for the integrated framework in the Live Dataset concatenating predicted label with input frames

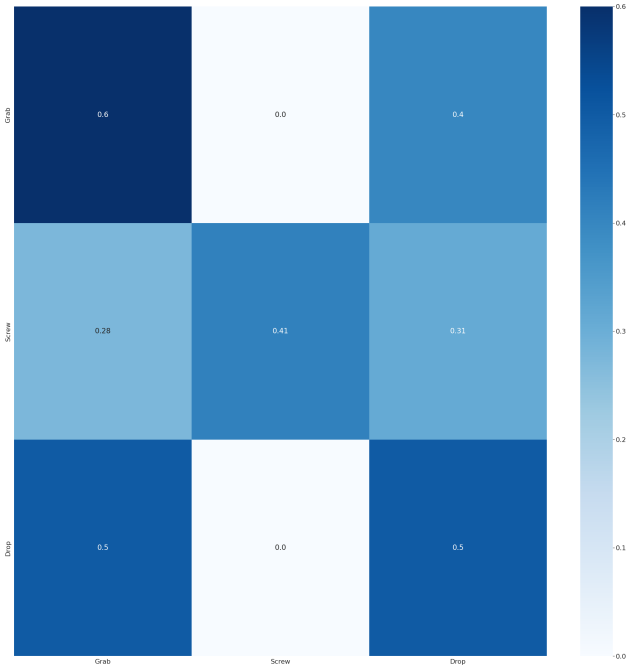

Figure 4.58: Confusion Matrix for recognition results on the integrated framework in the Live Dataset concatenating predicted label with input frames

Table 4.20: Metrics for the Recognition Model with the Live Dataset concatenating predicted label with input frames

Label	**Precision**	**Recall**	**F1-Score**
Grab	*0.60*	*0.25*	*0.35*
Screw	*0.41*	*1.00*	*0.59*
Drop	*0.50*	*0.08*	*0.14*
Accuracy			*0.44*
Macro Average	*0.50*	*0.44*	*0.36*
Weighted Average	*0.50*	*0.44*	*0.36*

Table 4.21: Integrated Framework approaches performance comparison

Method	Early Recognition			Movement Prediction		
	Gesture	Industry	Live	Gesture	Industry	Live
Early Recognition	*72,85%*	*30,87%*	*52,78%*	-	-	-
Movement Prediction	-	-	-	*75,33%*	*08,14%*	*22,22%*
Recognition final states initialize prediction	*74,96%*	*30,87%*	*52,78%*	*80,51%*	*08,14%*	*25,00%*
Prediction final states initialize recognition	*73,58%*	*30,87%*	*50,00%*	*75,33%*	*08,14%*	*22,22%*
Concatenate predicted frames with input	*72,85%*	*30,87%*	*41,67%*	*65,91%*	*08,14%*	*22,22%*
Concatenate predicted label with input	*72,85%*	*30,87%*	*44,44%*	*84,23%*	*08,14%*	*25,00%*

Considering the results seen in Table 4.21, it is possible to understand there is, in fact, a relevant impact on performance when we integrate these two networks.

Looking at the impact of prediction networks on the performance of early recognition, the results have to be analyzed with two critical factors in mind. The first one is that the network only predicts one frame in the conducted experiments, so the amount of information added is small and might not be relevant enough to have the impact it could have if a higher percentage of frames were predicted. The second factor to consider is that for the Industry and Live datasets, due to the problems already explained in Section 4.3.2 and Section 4.3.3, the performance for prediction is unusually low. This can lead to the impact on recognition performance to be a negative one instead of a positive, given that we are injecting inaccurate data that might confuse the model training instead of clarifying it.

In turn, the impact of early recognition networks in prediction is more straightforward and relevant. It is possible to observe that, although both methods have an impact, concatenating the predicted label with the input frame sequences has a really important impact on the overall performance.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This dissertation approached the problem of developing a real-time framework that simultaneously can recognize an activity being performed and predict the operator's movement for the next milliseconds. The developed framework relies strongly on OpenPose to extract the necessary information about the operator to make the predictions. The outcomes for the first models were very poor, not surpassing the test accuracy of 5,69% in recognition. After much research about LSTMs, optimization tools, and overfitting and oversampling techniques, the results improved significantly, reaching a final version that achieved 72,85% in recognition and 84,23% in prediction.

Extensive research is also presented regarding the current state of Computer Vision and HRC on action recognition and movement prediction, either simultaneously or separately. This research allowed a better understanding of how this topic is usually approached and some of its applications. However, most of the work found had no experimental data in live datasets, making it difficult to compare the results. In work researched that focus on industrial tasks, actions like grab, drop and screw are usually misrecognized due to the level of detailed data required to understand the action correctly. Additionally, they are actions with small and precise movements with a low positional variation.

Considering the four approaches for integrating the two networks, it was possible to conclude that all of them had some level of impact on system performance, either when the recognition network was used to provide inputs to the prediction or vice-versa. When the performance of the initial network is worst, it harms the other component's performance. However, when it is able to achieve accuracy above 50%, it is sufficient to help improve the overall structure. Comparing the approaches of initializing a layer with the final states of the other and feeding the output as input to the other one, the latter revealed itself to be more impactful. For this reason, this was the approach chosen for the final solution.

One of the main limitations of this work was the size of the live dataset. Considering that the other datasets did not have the same type of data (different number of joints and positioned in different places), the network had to be trained from scratch with this dataset. The data acquisition originated only 132 action sequences, 44 for each label. This significantly limits how much the model can extract and learn from the data.

5.2 Future Work

Considering that our most significant limitation was regarding live dataset size, the first thing to do in future work is to expand the amount of data collected. This can be done by increasing the number of trials for each participant and the number of participants. With this, we would have a good base for understanding the actions and making more precise predictions, particularly regarding the prediction network.

Another relevant approach that should be explored in future work is applying OpenPose to the videos in the Montalbano Gesture and the Industry-Oriented Datasets. This would allow us to expand the available data for training as well as work as a tool to understand if OpenPose is, in fact, a reliable method for keypoint calculation or if other frameworks should be considered.

After this, an important study was whether it was possible to reduce the number of keypoints extracted after expanding training data, maintaining good performance.

Additionally, it would be interesting to experiment with the implementation of CNN-based networks and compare them with the developed LSTM-based ones. In the research made, this was usually reported as an excellent alternative to LSTMs for some cases.

References

[1] Mohammad Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson. Encouraging lstms to anticipate actions very early. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 280–289, 2017.

[2] Benjir Islam Alvee, Sadia Nasrin Tisha, and Amitabha Chakrabarty. Application of machine learning classifiers for predicting human activity. In *2021 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pages 39–44, 2021.

[3] Andrea Bauer, Dirk Wollherr, and Martin Buss. Human-robot collaboration: a survey. *I. J. Humanoid Robotics*, 5:47–66, 03 2008.

[4] Ujjal Kr Dutta, Mehrtash Harandi, and Chellu Chandra Sekhar. Unsupervised deep metric learning via orthogonality based probabilistic loss. *IEEE Transactions on Artificial Intelligence*, 1(1):74–84, Aug 2020.

[5] Sergio Escalera, Xavier Baró, Jordi Gonzàlez, Miguel Bautista, Meysam Madadi, Miguel Reyes, Víctor Ponce-López, Hugo Jair Escalante, Jamie Shotton, and Isabelle Guyon. Chalearn looking at people challenge 2014: Dataset and results. pages 459–473, 03 2015.

[6] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction, 2016.

[7] Antonino Furnari and Giovanni Maria Farinella. Rolling-unrolling lstms for action anticipation from first-person video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):4021–4036, 2021.

[8] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[9] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–970, 2015.

[10] Minh Hoai and Fernando De la Torre. Max-margin early event detectors. *International Journal of Computer Vision*, 107(2):191–202, 2014.

[11] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014.

[12] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 5308–5317, 2016.

[13] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *International Conf. on Computer Vision (ICCV)*, pages 3192–3199, December 2013.

[14] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset, 2017.

[15] Daniel Kofer, Christian Bergner, Christian Deuerlein, Ronald Schmidt-Vollus, and Peter Heß. Human–robot-collaboration: Innovative processes, from research to series standard. *Procedia CIRP*, 97:98–103, 2021. 8th CIRP Conference of Assembly Technology and Systems.

[16] Shufei Li, Junming Fan, Pai Zheng, and Lihui Wang. Transfer learning-enabled action recognition for human-robot collaborative assembly. *Procedia CIRP*, 104:1795–1800, 2021. 54th CIRP CMS 2021 - Towards Digitalized Manufacturing 4.0.

[17] Wanqing Li, Zhengyou Zhang, and Zicheng Liu. Action recognition based on a bag of 3d points. pages 9 – 14, 07 2010.

[18] Bowen Liu, Yu Chen, Shiyu Liu, and Hun-Seok Kim. Deep learning in latent space for video prediction and compression. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 701–710, 2021.

[19] Hongyi Liu and Lihui Wang. Human motion prediction for human-robot collaboration. *Journal of Manufacturing Systems*, 44:287–294, 2017. Special Issue on Latest advancements in manufacturing systems at NAMRC 45.

[20] Hongyi Liu and Lihui Wang. Gesture recognition for human-robot collaboration: A review. *International Journal of Industrial Ergonomics*, 68:355–367, 11 2018.

[21] Ruixuan Liu and Changliu Liu. Human motion prediction using adaptable recurrent neural networks and inverse kinematics. *IEEE Control Systems Letters*, 5(5):1651–1656, 2021.

[22] Shugao Ma, Leonid Sigal, and Stan Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1942–1950, 2016.

[23] Pauline Maurice, Adrien Malaisé, Clélie Amiot, Nicolas Paris, Guy-Junior Richard, Olivier Rochel, and Serena Ivaldi. Human movement and ergonomics: An industry-oriented dataset for collaborative robotics. *The International Journal of Robotics Research*, 38(14):1529–1537, 2019.

[24] Vladimir Pavlovic, James M Rehg, and John MacCormick. Learning switching linear models of human motion. In *NIPS*, volume 2, page 4, 2000.

[25] John Phillips, Julieta Martinez, Ioan Andrei Bârsan, Sergio Casas, Abbas Sadat, and Raquel Urtasun. Deep multi-task learning for joint localization, perception, and prediction. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4677–4687, 2021.

[26] Don J. Rude, Stephen Adams, and Peter A. Beling. A benchmark dataset for depth sensor based activity recognition in a manufacturing process. *IFAC-PapersOnLine*, 48(3):668–674, 2015. 15th IFAC Symposium onInformation Control Problems inManufacturing.

[27] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis, 2016.

[28] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild, 2012.

[29] Richard Szeliski. *Computer Vision - Algorithms and Applications.* Texts in Computer Science. Springer, 2011.

[30] Krasimir Tonchev, Agata Manolova, Radostina Petkova, and Vladimir Poulkov. Human skeleton motion prediction using graph convolution optimized gru network. In *2021 XXX International Scientific Conference Electronics (ET)*, pages 1–5, 2021.

[31] Stefano Tortora, Stefano Michieletto, Francesca Stival, and Emanuele Menegatti. Fast human motion prediction for human-robot collaboration with wearable interface. In *2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 457–462, 2019.

[32] Vinh Tran, Niranjan Balasubramanian, and Minh Hoai. Progressive knowledge distillation for early action recognition. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 2583–2587, 2021.

[33] Vinh Tran, Yang Wang, Zekun Zhang, and Minh Hoai. Knowledge distillation for human action anticipation. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 2518–2522, 2021.

[34] Athanasios Voulodimos, Dimitrios Kosmopoulos, Georgios Vasileiou, Emmanuel Sardis, Anastasios Doulamis, Vassileios Anagnostopoulos, Constantinos Lalos, and Theodora Varvarigou. A dataset for workflow recognition in industrial scenes. In *2011 18th IEEE International Conference on Image Processing*, pages 3249–3252, 2011.

[35] Boyu Wang and Minh Hoai. Back to the beginning: Starting point detection for early recognition of ongoing human actions. *Computer Vision and Image Understanding*, 175:24–31, 2018.

[36] Boyu Wang and Minh Hoai. Predicting body movement and recognizing actions: An integrated framework for mutual benefits. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, pages 341–348, 2018.

[37] Boyu Wang, Lihan Huang, and Minh Hoai. Active vision for early recognition of human actions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1078–1088, 2020.

[38] Peng Wang, Hongyi Liu, Lihui Wang, and Robert X. Gao. Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. *Cirp Annals-manufacturing Technology*, 67:17–20, 2018.

[39] Peng Wang, Hongyi Liu, Lihui Wang, and Robert X. Gao. Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. *Cirp Annals-manufacturing Technology*, 67:17–20, 2018.

[40] Yang Wang and Minh Hoai. Improving human action recognition by non-action classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2698–2707, 2016.

[41] Yunfeng Wang, Wengang Zhou, Qilin Zhang, Xiaotian Zhu, and Houqiang Li. Weighted multi-region convolutional neural network for action recognition with low-latency online prediction. In *2018 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, pages 1–6, 2018.

[42] Zanwu Xia, Qujiang Lei, Yang Yang, Hongda Zhang, Yue He, Weijun Wang, and Minghui Huang. Vision-based hand gesture recognition for human-robot collaboration: A survey. In *2019 5th International Conference on Control, Automation and Robotics (ICCAR)*, pages 198–205, 2019.

[43] Maosheng Ye, Tongyi Cao, and Qifeng Chen. Tpcn: Temporal point cloud networks for motion forecasting. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11313–11322, 2021.

[44] Yan Zhang, Michael J. Black, and Siyu Tang. We are more than our joints: Predicting how 3d bodies move. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3371–3381, 2021.

[45] Jing Zhao, Shiqiu Gong, Biyun Xie, Yaxing Duan, and Ziqiang Zhang. Human arm motion prediction in human-robot interaction based on a modified minimum jerk model. *Advanced Robotics*, 35(3-4):205–218, 2021.

[46] Tian Zhou and Juan P. Wachs. Early prediction for physical human robot collaboration in the operating room, 2017.