Transparent control of 1-DOF exoskeleton using LSTM torque prediction and FMG weight estimation

Guillermo Gutiérrez Bea, Malthe Boelskift, Louis Ildal, Emil Lunau Bentsen & Nikolaos Gkloumpos
Electronic Systems 8th semester from Aalborg University

August 1, 2024

Abstract: Upper-limb active exoskeletons hold significant potential in alleviating musculoskeletal strain during load-bearing activities. Effective real-time assistance requires precise prediction of the necessary support torque for upcoming movements. This study implements a novel approach to predicting such torque by utilizing LSTM models trained on inverse dynamics calculated torque from real data collected from an exoskeleton. The prediction is combined with admittance control to synchronize the exoskeleton's movements with the user's for seamless assistance and load management. Using Force Myography (FMG) data, we train a model to classify a finite set of loads, whose value is used in the inverse dynamics after the initialization. The results of our simulations suggest that this approach has potential, but requires further experimental testing to validate the approach.

Index Terms—Exoskeleton, Predictive control, LSTM, Force Myography (FMG), Inverse Dynamics, Admittance Control

1 Introduction and purpose

The interaction between humans and machines has never been more relevant[1]. This interaction can take multiple forms but this research paper will focus on the control of exoskeletons. Exoskeletons are wearable devices designed to enhance the wearers capabilities. They serve various purposes ranging from rehabilitation and strain relief to strength augmentation [2][3].

There exists active and passive exoskeletons, with the difference being that active ones have motorised joints (or some other form of actuator) while the passive ones do not. Exoskeletons usually focus on specific muscle groups, but versions exist that encompass the whole user either augmenting or supporting all of their movements[3][4].

The shoulder area is the most common muscle group that gets overloaded (typically due to over-head work). It is also the muscle group which can be expected to take the most damage over time due to it's large range of motion [5], which is why there have been multiple attempts at making exoskeletons mitigating this issue [2][3][1]. This paper aims to design a control system for an upper limb controlled exoskeleton, which can detect if the user carries a load or not, in order to provide the appropriate level of assistance.

To effectively control an active exoskeleton, detecting the wearer's intended movements as well as the payload weight is imperative [6][7][8]. Over the years a variety of sensors have been used for both of these purposes [9]. For this paper, a rotary encoder is used to measure motion of the users elbow joint and a sensor band using Force Myography (FMG) is used to sense the payload weight. The armband used contains 8 Force Sensitive Resistors (FSR), which are measuring the force exerted on them by the triceps and biceps. Figure 1 illustrates the system as a whole and it's concept, with a test subject wearing the FMG band and exoskeleton. As discussed, part of this paper aims to build a system which can provide an appropriate level of assistance. This is done by predicting future movements based on previous ones. To complete that objective an LSTM (Long Short Term Memory neural network) is utilised. LSTM is a type of recurrent neural network that performs extremely well when tasked with solving sequence prediction problems [10], making it an ideal choice for our movement prediction mechanism (in figure 1 it can be seen how LSTM is part of the system). However, it should be noted that the effectiveness of an LSTM (or any neural network) in making accurate predictions is highly dependent on the quality of the training data. Our solution integrates inverse dynamics alongside

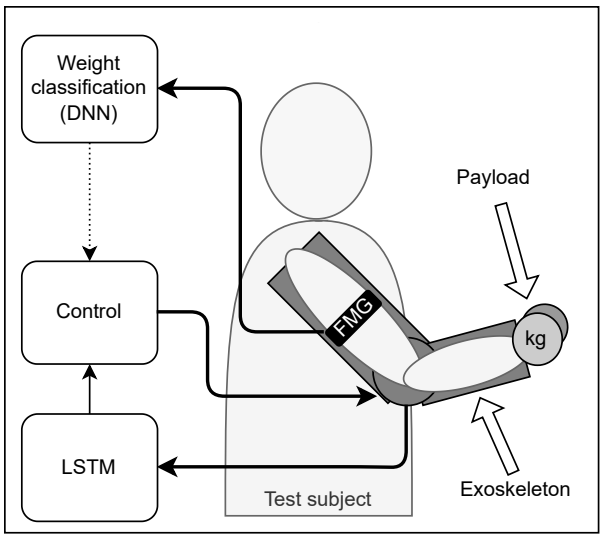


Figure 1: *Concept diagram of test subject wearing FMG band and exoskeleton, with system parts sketched*

the LSTM to provide a solution both for motion prediction and load estimation. The rotary encoder data is used by the LSTM to predict future movements by the user. By integrating inverse dynamics principles, we seek to develop control for an exoskeleton with the ability to dynamically adjust its support in response to the user's movements, thereby minimising strain and maximising comfort during various activities. Additionally, by measuring the muscle expansion during flexion the system can accurately estimate the payload weight which can then be given to the control system to compensate for it. We have achieved the following contributions

- Experimental validation of LSTM for torque prediction
- Implementation of LSTM network on a microcontroller
- Design and Implementation of an admittance controller based upon predicted compensation torque values.

We will begin by surveying related work in this field. Subsequently, we will introduce the notation and definitions essential for framing the problem context accurately, and define preliminaries necessary for understanding this paper. Following this, we will develop a control algorithm designed to accurately predict torque exerted on exoskeleton joints, using inverse dynamics principles and machine

learning techniques. A payload classifier will be designed to classify the weight during the initialisation.

Finally, we will detail the simulations and experiments conducted and showcase the results obtained from evaluating the performance of our system. Through these experiments, we aim to demonstrate the effectiveness of our approach in classifying whether a person is carrying a load, based on sensor data from the FMG band, and, with the developed control algorithm, provide seamless interaction with the exoskeleton.

2 Related work

In this section, we will have a look at some related work. In many ways, this paper aims to build upon some of the work seen in this section.

The primary inspiration comes from the paper: "Towards Data-driven predictive control of upper-body exoskeletons for payload carrying"[11]. Specifically, their data-driven approach for predicting user intentions to improve transparency. The research paper conducts analyses of different prediction methods for active exoskeletons. These include techniques for predicting joint torques based on future joint positions and dynamical models, as well as the prediction of future joint torques from past calculated torques. LSTM-based neural networks were employed, having emerged as a prominent choice for modeling temporal dependencies and predicting future motion in such scenarios [11]. However, the training data used was generated from motion capture data to train the prediction models, whereas for the work done in this paper, we have trained the models on data captured using a single joint exoskeleton.

Furthermore, this research extends some of the ideas proposed in "Robust Payload Recognition Based on Sensor-Over-Muscle-Independence Deep Learning for the Control of Exoskeletons" [12]. This paper has explored various sensing techniques and pre-processing algorithms to improve the reliability of FMG data interpretation. These include methods for organising irregular sensory data into regular patterns, such as the sensor over muscle independence (SOMI) pre-processing algorithm presented in the paper. By mitigating the effects of sensor displacement and orientation changes, these algorithms aim to enhance the resilience of FMG based classification systems. The research incorporates Machine learning techniques, including deep neural networks (DNNs), support vector machines (SVMs), k-nearest neighbors (KNN), and decision trees (DT), which have been employed for FMG data classification. Researchers have investigated the performance of these algorithms in recognising human intention from FMG signals. Recent efforts have focused on developing lightweight DNN architectures tailored to FMG data, aiming to achieve high classification accuracy while minimising computational complexity [12].

This research paper aims to integrate some of these known methods, into a more complete control system trained on data from a real exoskeleton instead of a simulated model.

3 Preliminaries

In this section, we will begin by introducing the notations and definitions relevant to our use of control methods and machine learning techniques within the context of exoskeletons. Next, we will present the formalisation of the problem we aim to solve: Transparent control of 1-DOF exoskeletons. Following this, we will briefly discuss the concepts used in our Neural Network approach which are independent of the control process.

The reader should be familiar with Advanced control engineering, Advanced Engineering mathematics, and have knowledge of programming and machine learning.

In table 3 an overview of some different definitions and notations can be seen

Table 1: *Notations and Definitions*

Abbreviations	Definition
FMG	Force Myography
EMG	Electro Myography
RMG	Radio Myography
LSTM	Long Short-Term Memory
DOF	Degrees Of Freedom
FSR	Force Sensitive Resistor
SOMI	Sensor Over Muscle Independence
DNN	Deep Neural Network
RNN	Recurrent Neural Network
SVM	Support Vector Machine
KNN	K-Nearest Neighbors
DT	Decision Tree
RMS	Root Mean Squared
RMSE	Root Mean Squared Error
MAE	Mean Absolute Error

Variables	Definition
$X_{FMG}(t)$	FMG input data at time t
$F_i(t)$	Force measurement from sensor i at time t
$L(t)$	Load status categories at time t
$\tau_{(t-h\rightarrow t)}$	Torque history from $t-h$ to t
$\tau_{(t+1\rightarrow t+N)}$	Predicted future torque from $t+1$ to $t+N$
$\hat{\tau}_{(t+1\rightarrow t+N)}$	Estimated future torque from $t+1$ to $t+N$
M	Mapping function in LSTM
τ_c	Calculated torque
J_E	Inertia of the exoskeleton
J_p	Inertia of the payload
$\ddot{\theta}$	Angular acceleration
m_{link}	Mass of the link
L_{COM}	Distance to the center of mass
m_p	Mass of the payload
L_{link}	Length from joint to payload
$Y(s)$	Admittance transfer function
B	Desired inertia
D	Desired damping
K	Desired stiffness
T	Sampling time
Nm	Newton meter

3.1 Problem Formalisation

Firstly, we look to define the weight classification part of our system. The input $X_{FMG}(t) = \{F_0(t), F_1(t), \ldots, F_8(t)\}$ denotes the 8-column data at time t. The output is all the possible categories the data is classified into $L(t) = \{L_0, L_1, \ldots, L_N\}$ for N categories.

The goal is to classify the load status L based on the force measurements X_{FMG} at time t. In this project, the data was classified into

four categories depending on whether the arm was at rest, flexed with no load, and flexed with two different loads.
A feedforward neural network is used to map the FMG preprocessed data X_{FMG} to the load status L at time t.

The torque prediction task can be formulated as:

Torque prediction input:

$$\tau_{(t-h\rightarrow t)} = \tau_{t-h}, \ldots, \tau_{t-1}, \tau_t \tag{1}$$

where h represents the history length of the past timestamps and is associated to a calculated torque using inverse dynamics on the angle and angular acceleration data.

Torque prediction output:

$$\tau_{(t+1\rightarrow t+N)} = \tau_{t+1}, \ldots, \tau_{t+N-1}, \tau_{t+N} \tag{2}$$

The LSTM will be trained to behave as a mapping function M from the input sequence of past torques to the output sequence of future torques.

$$\hat{\tau}_{(t+1\rightarrow t+N)} = M(\tau_{(t-h\rightarrow t)}) \tag{3}$$

4 Methods

This section will describe the methods used in this paper. As seen in figure 2, a variety of different methods are used, from the initial payload estimation to the ongoing control of the exoskeleton.
The payload estimation is only done once in the initialisation. The payload estimate is fed into the inverse dynamics, which uses this value to produce torque prediction, used in the control of the exoskeleton.

4.1 Payload classification with DNN and SOMI

The payload classification is achieved by collecting data from an FMG-band, which is then trained by a neural network to be able to classify the data into four different categories: rest, flexed without load, flexed with 1.23kg load and flexed with 2.31kg load.

Before training the neural network the data was normalised using the SOMI algorithm, which was mentioned in the section Related Work 2, developed in a related paper [12]. SOMI standardises the data with a 4-step algorithm such that the data is independent of the FMG-band placement. [12].
The neural network is a 5-layer network with the number of neurons in each layer being 512, 256, 128, 32, and 4 respectively. For all layers, except for the output layer, the ReLU function is used as an activation function. A softmax activation function was used for the output layer.

4.2 Torque calculation with Inverse Dynamics for 1-DOF exoskeleton

To provide an input that can be used in LSTM prediction Inverse Dynamics was utilised. The model used for the 1-DOF exoskeleton is as follows.

$$\tau_c = (J_E + J_p)\ddot{\theta} + (m_{link}L_{COM} + m_pL_{link})gsin(\theta) \tag{4}$$

where J_E is the inertia of the exoskeleton, J_p is the inertia of the payload and $\ddot{\theta}$ is the angular acceleration. $\ddot{\theta}$ is measured using an encoder on the joint motor, J_p is calculated using the weight classification given by the DNN and J_E is treated as a constant. The second term of the equation refers to the torque produced by gravity on the link and on the payload and this torque is a function of the angular position (defined as 0 degrees when the arm is resting vertically pointing down).m_{link} is the mass of the link and L_{COM} is the distance to its center of mass. L_{link} is the length from the joint to the payload.

4.3 Torque prediction with LSTM

To achieve transparency, a predictive approach is used. Specifically, the future torques produced by the combination of the exoskeleton's motion and the payload's weight are estimated to be able to correct them on time. An LSTM is used to make predictions due to its ability to capture patterns in sequential data and its mechanism to capture long-term dependencies on the data [10]. As proposed in [11], the LSTM predicts the N future torques $\tau_c(t+1 \rightarrow t+N)$ from h previous torques $\tau_c(t-h \rightarrow t)$.

To produce multiple predictions a feedforward layer was connected to the output of the LSTM which included N neurons. Since the predictor is designed to run in a micro-controller, a light LSTM model of two neurons was implemented. The feedforward layer,

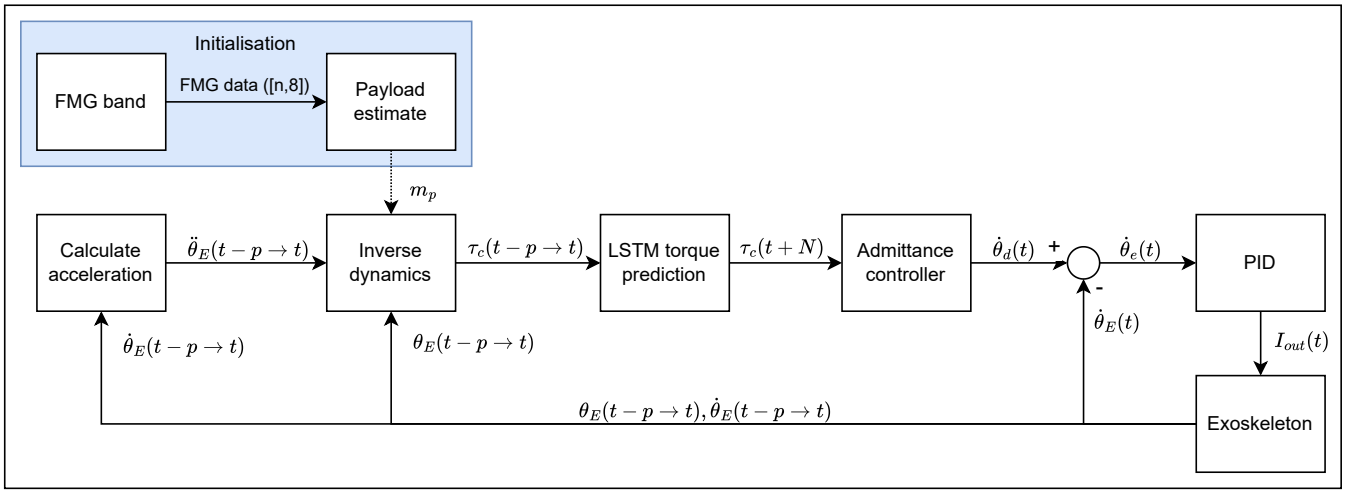


Figure 2: *Block Diagram of Admittance control for a 1DOF exoskeleton*

consisting of N neurons with an atan activation function, takes the last output of the LSTM as input to produce N outputs. To facilitate the training, the input data and target outputs where scaled form -1 to 1. This means that the input data is expected to be scaled from -1 to 1 and therefore a scaling function was implemented and an inverse scaling function in the output to revert the scale back to the original scaling. Differencing was also applied to the signal to make the time series' mean stationary.

Based on the partial autocorrelation of the differenced torque dataset and through trial and error, we determined the optimal number of lags required for the prediction. The partial autocorrelation shows the correlation between a point and a lagged point by k removing the effect of any correlations due to the terms at shorter lags[13]; this served as a starting point to determine which lags to include. In the end, three lags seemed to make an optimal trade-off between complexity and accuracy; according to our experience.

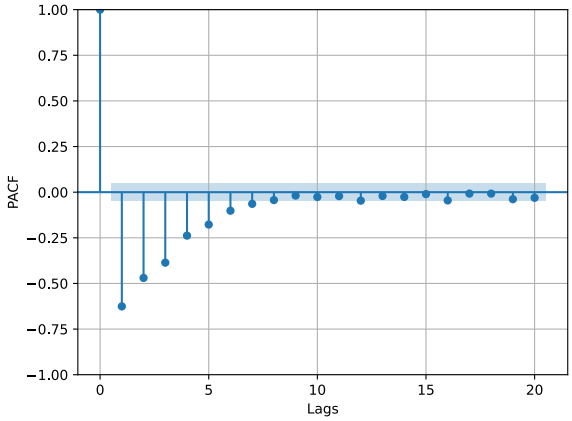


Figure 3: *Partial autocorrelation function of the differenced torque dataset.*

4.4 Admittance Controller Design

An admittance controller is a common type of controller used in robotics. It allows the robot, in this case the exoskeleton, to follow the external force applied to the exoskeleton seen as a torque. The typical equation for an admittance controller is:

$$Y(s) = \frac{\dot{\theta}(s)}{\tau(s)} = \frac{1}{B \cdot s + D + K \cdot s^{-}1} \tag{5}$$

Here the parameters B, D and K are desired inertia, dampening and stiffness respectively. These, along with the input torque, are used in a virtual object to calculate the angular velocity. This value is a set-point for the PID controller as seen in figure 2. Here it is compared with feedback from the actual angular velocity from the exoskeleton. This seeks to minimise the error between the angular velocity from the virtual object and the one measured at the encoder on the exoskeleton.

The transfer function was implemented as 1st order IIR in the Direct I form; which yields the following equation:

$$\dot{\theta}[n] = -\frac{-2B + DT}{2B + DT}\dot{\theta}[n-1] + \frac{T}{2B + DT}(\tau[n] + \tau[n-1]) \tag{6}$$

Where T is the sampling time and the stiffness is set to zero.

5 Experiments & Results

This section describes the dataset, the experiments conducted, and how our system performed, and compares the results to the control of the exoskeleton without our system. We will then delve into some noteworthy results to highlight its strengths and weaknesses.

5.1 Payload Classification Test

In order to be able to get some input data for our control system, a payload classification test was performed to get some data to train the neural network with. This section will describe the process behind the test and gathering of the data.

It's important to understand that variations in data distributions between two measurements or tests conducted on the same individual can be comparable to the differences observed when measuring across different individuals. In figure 4 an example of some data gathered from the FMG band can be seen. It can be noted how different the data look for different people.

Data Collection and Preprocessing Our data is collected using the FMG band with 8 sensors inside the band, with a MATLAB application that facilitates the calibration of the FMG sensors, data collection, and selection of the data to be saved. The data is recorded in sessions of approximately 200 seconds each. The data is sampled at around 300 Hz, generating approximately 60.000 samples per 200-second session.

We conducted 15 measurement sessions, three for each of the participants in the research group, one without load, one with a light load and lastly one with a heavier load. The static part of the data was saved and labeled. Later on, all the data files from in each category from the different participants were concatenated into arrays and shuffled. Then the shuffled data was split into training and test sets.

5.2 Payload Classification results

The classifier is trained to distinguish FMG data in four different situations: resting (R), arm flexed with no load (NL), flexed with a 1.23kg load (L1), and flexed with a 2.31 load (L2). After classifying the test dataset, the following confusion matrix is obtained:

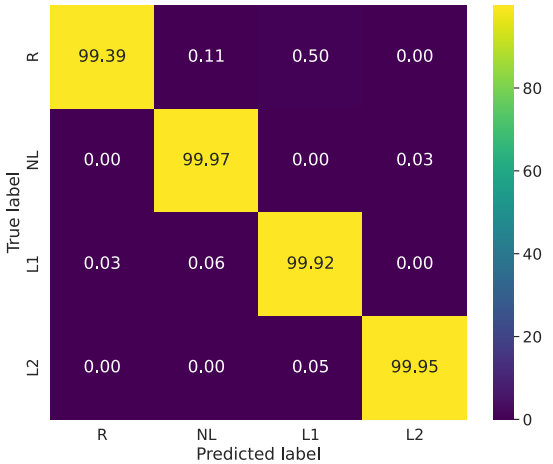


Figure 5: *Confusion matrix from classifying the test dataset*

The confusion matrix shows the rate of predicted labels against the true label of data. This means that the x-axis corresponds to the

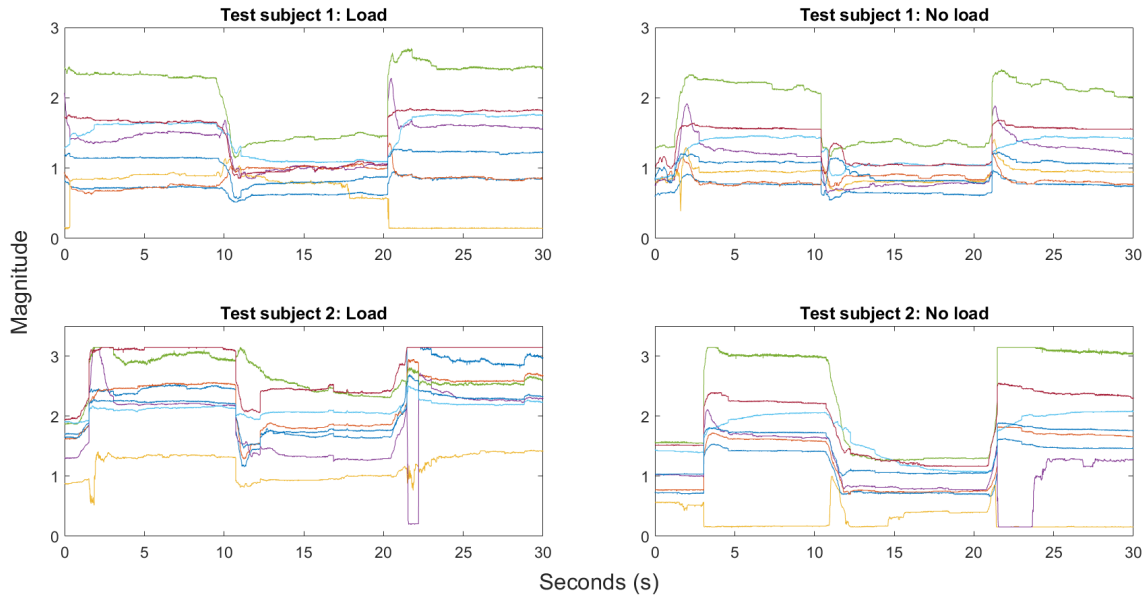


Figure 4: *FMG data for two different test subjects*

predicted label and the y-axis to the actual label of the data; this tool can help visualising where the model goes wrong, which data points were misclassified, and the rate of it. The results show the high accuracy of the model.

The model was validated with cross-validation. By training the model for five epochs in a 5-fold cross-validation we consistently obtained an average accuracy of > 98, which seems consistent with the values from the confusion matrix.

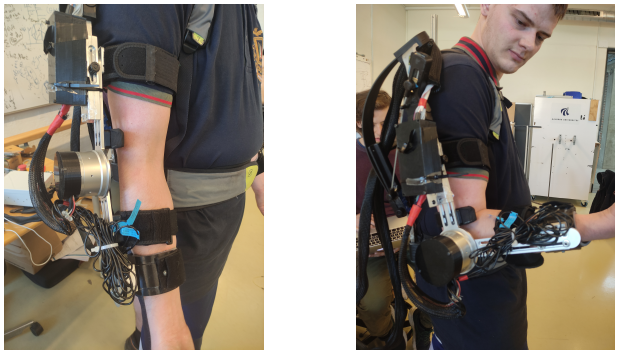
5.3 Transparency test

To evaluate the transparency of the exoskeletons control algorithm the interaction torque was measured using a load cell placed at the wrist of the exoskeleton. The torque was then measured during flexion and extension of the arms with the control algorithm in both on and off states. These test where performed with 0kg payload and 1.1 kg payload.In figure 6 the exoskeleton can be seen in the middle of testing the transparency.

5.4 Torque prediction test

To evaluate the prediction of the embedded LSTM, a test was performed with and without the controller connected. The test with the controller disabled is performed to see the prediction without any influence on the motion to get a clean graph with the prediction against the current calculated value. One group member wore the upper limb exoskeleton, flexing and extending his arm multiple times 6a, while another recorded the incoming data. The prediction is comprised of the fifth step-prediction of the torque. The results can be found in figure 10 and 11.

The LSTM was implemented in the Adafruit ESP32 feather together with the controller's code. Initially, the TensorFlow model underwent conversion to TensorFlow Lite, a lightweight version oriented to mobile and embedded systems. Subsequently, full integer quantisation was applied, reducing all tensors, including parameters, activations, inputs, and outputs, to 8-bit integers (Int8). This reduces the model size significantly as well as ensuring compatibility with the library used EloquentTensorflow32.



(a) *Exoskeleton during extension.* **(b)** *Exoskeleton during flexion*

Figure 6: *Exoskeleton mounted on subject during data collection.*

5.5 Torque prediction results

The simulation results in the following root mean squared error (RMSE) for five predictions into the future in the test set: 0.000046, 0.000045, 0.000043, 0.000045, 0.000045. The figure 7 shows the one step prediction against a five-step shifted version of the original signal for comparison.

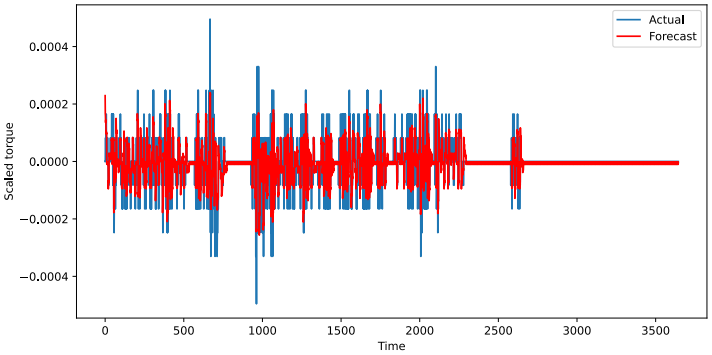


Figure 7: *Torque disregarding the contribution of gravity and without a payload.*

In the embedded LSTM, only one output could be successfully obtained from the LSTM model. Another requirement was the need to do a full integer quantisation of the LSTM model. This led to the need to create a quantised model with only one output. The following figure shows the prediction of the original tensorflow model and the prediction of the quantised model for comparison.

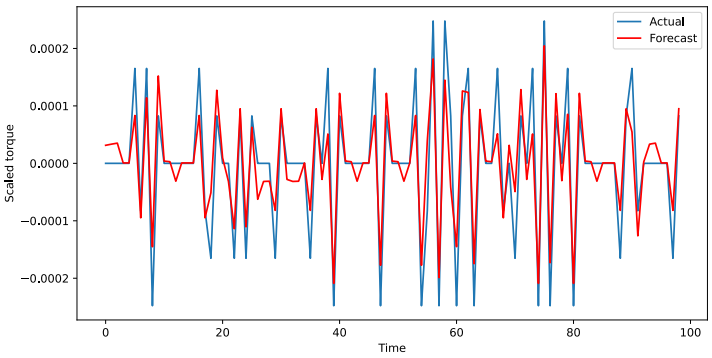


Figure 8: *One step prediction model. The actual signal is shifted to the left for better comparison.*

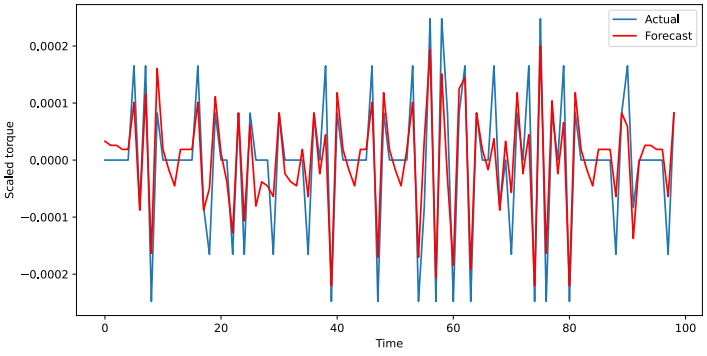


Figure 9: *Quantised one-step prediction model. The actual signal is shifted to the left for better comparison.*

The experiment results using the real exoskeleton yield the following graphs. Where figure 10 shows the prediction against the calculation given that the controller was disabled. Even though the prediction seams to follow the calculation it shows an underlying issue with the LSTM implementation since the prediction is not taking place. The same result can be viewed in the figure 11 where the controller was enabled. This points out the need for further troubleshooting of the prediction code.

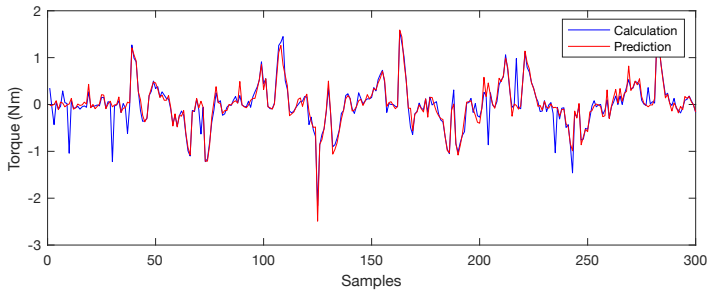


Figure 10: *Plot from the experiment where the controller was disabled. The figure shows the comparison between the calculated τ_c against the prediction of the embedded LSTM.*

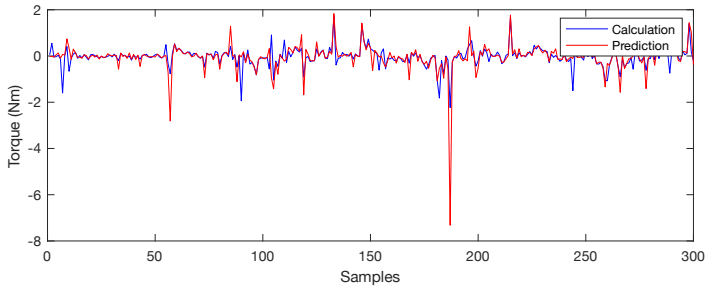


Figure 11: *Plot from the experiment where the controller was enabled. The comparison between the calculated τ_c against the prediction of the embedded LSTM is shown.*

This realisation led the efforts to further analysis on the predictions that were taking place on the embedded LSTM. It seemed that either the predictions or the scaling and differencing preprocessing was causing the problems. To pinpoint the problem, several tests were conducted. The first conclusive test showed that although the performance of the prediction seems to be degraded compared to the expected performance of the quantised model, it does predict one step into the future as expected. The comparison can be made between the figures 9 and 12.

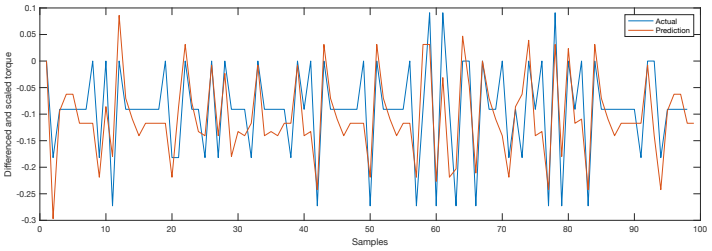


Figure 12: *One-step prediction of the quantised model against the actual signal performed by the embedded LSTM. The actual signal is pre-differenced and scaled and inference was performed directly on it. The prediction is shifted one step to the right for better comparison.*

The next potential failure point is the differencing and scaling functions. Due to time constraints, the issue was not resolved.

5.6 Transparency test results

The transparency test resulted in the interaction torque graphs seen on figures 13, 14,15 and 16. The difference between the controller being disabled and enabled is difficult to distinguish from the figures. In testing what was generally observed was that the control algorithm would only help you once you are already in motion resulting in similar peak interaction forces regardless of whether the controller was enabled or disabled. The difference is mostly while in motion where the RMS interaction torque is noticeably lower when the controller is enabled.

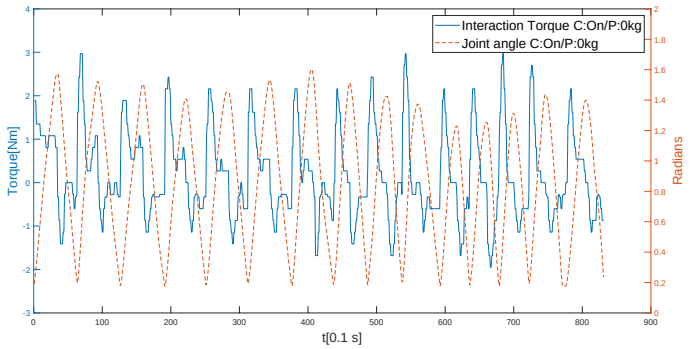


Figure 13: *Interaction torque and joint angle for transparency test with controller enabled and 0kg payload. RMS Interaction Torque:1.0332 Nm.*

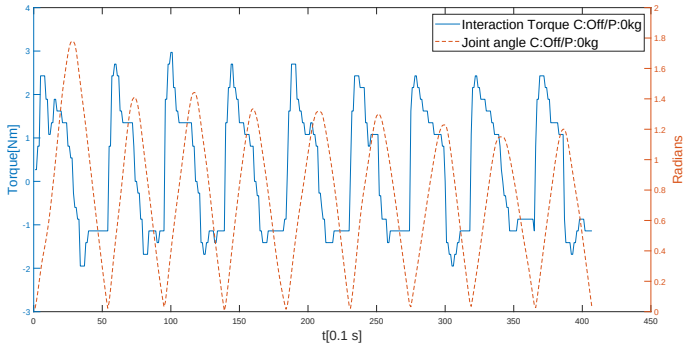


Figure 14: *Interaction torque and joint angle for transparency test with controller disabled and 0kg payload. RMS Interaction Torque:1.4432 Nm.*

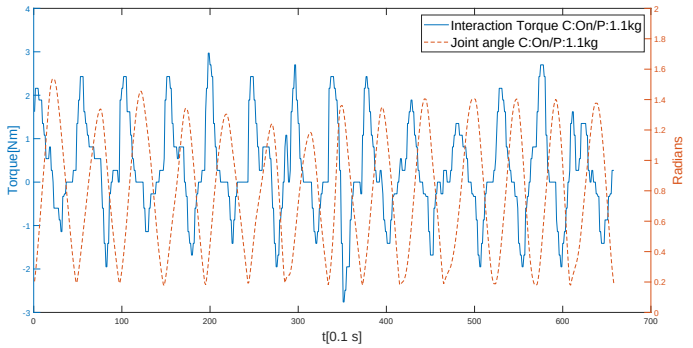


Figure 15: *Interaction torque and joint angle for transparency test with controller enabled and 1.1kg payload. RMS Interaction Torque:1.1536 Nm.*

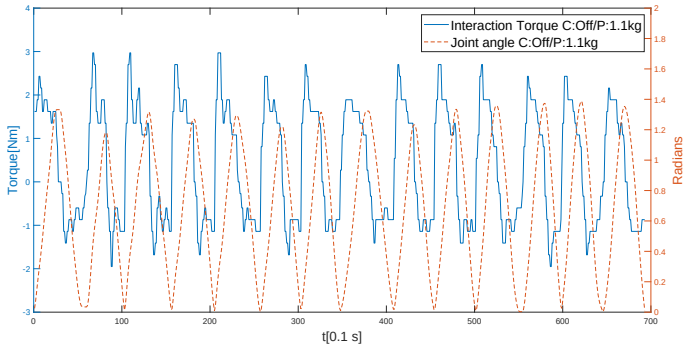


Figure 16: *Interaction torque and joint angle for transparency test with controller disabled and 1.1kg payload. RMS Interaction Torque: 1.3941 Nm.*

6 Discussion

This section aims to discuss how the methods used in the paper turned out and reflect upon how things could have been done differently to achieve a better outcome.

6.1 Sensor data

In the weight classification of the project, we chose to utilise FMG as our way of collecting data. This was convenient, as it was an already proven method[12]. Alternative ways of collecting data were considered, for example: EMG (electromyography) and RMG (radiomyography). These two methods could provide data by measuring the muscles directly, whereas FMG can only measure the external force inflicted on the band[6].
A disadvantage of FMG is the fact that a FMG-band is needed to collect data. The band itself has physical limits to size and number of FMG force sensors on it. The physical limitations of EMG and RMG are fewer than FMG, which could make them more convenient alternatives.

For this particular study, the weight classification was not done in real time, but was based on data collected previous to the control of the exoskeleton. This means that there exists a potential for improvement by changing the weight estimation to a real time prediction.

6.2 Weight classifier

The weight prediction method requires the user to lift the load until it is classified and sent to the controller. This can be seen as a limitation in some scenarios where compensation is required from the start. Moreover, every time the user changes the amount of load is carrying the initialisation has to be repeated to classify the new load.

6.3 Transparency test

The embedded LSTM was found to have some flaws causing the torque prediction to malfunction. As a result, the transparency test results cannot be taken as experimental proof of the controller. In fact, since the prediction was not successful, the control input was essentially the current torque to compensate τ_c calculated from the inverse dynamics which provided some assistance if the admittance control parameters were fine-tuned but with the risk of instability.

6.4 Torque prediction

LSTM was chosen as the way to make the torque prediction due to the popularity and promising results that LSTM has provided as an RNN method for the past few years. Many other predicting methods with neural networks struggle to predict long-term dependencies well, but this is one of LSTM's biggest strengths. LSTM copes with long-term dependencies very well and is at its most effective in sequence prediction such as a time series, which is also the case in this paper.
One of the biggest challenges associated with using LSTM was the complexity that it adds to an RNN. Both the computational cost, which leads to longer training time, and the difficulties interpreting the model led to various issues. Making real-time predictions with LSTM requires enough computational power for it to be insufficient with the smallest microcontrollers available. As size and weight are important factors to consider for strength augmenting exoskeletons, this could be a concern for real-life implementation.

Since the quantized LSTM model deployed on the ESP32 lost some performance in the microcontroller, the issue must be investigated to improve the predictions done on it. The preprocessing functions must be extensively tested since there is most likely a problem with them.

6.5 Future work

The control mechanisms showcased in this paper have showcased possible improvements to exoskeleton control. To fully take advantage and expand upon this work we believe there are a few aspects of the system that can be iterated upon.

The issues with the embedded LSTM must be addressed. This could involve making quantization-aware training instead the the technique used, which is post-training quantization. This approach would involve training the model having in mind that quantization will take place and therefore trying to minimize the loss of information or accuracy that can arise from the quantisation process. Furthermore, the differencing and scaling functions must be extensively tested to find the problem that is causing the unexpected results.

On the sensing side, as mentioned in the discussion section, using other types of sensors to measure muscle activation is an area that could yield significant improvements for the algorithm. Further more, integrating those sensor in the motion system would enable faster actuation of the system enhancing its transparency.

Finally, exploring different machine learning techniques could yield further improvements. Specifically, reinforcement learning (RL) appears to be a strong candidate due to its ability to optimise control strategies through trial and error, continually improving performance based on feedback from the environment [14]. That way it can adjust to individual user movements, providing personalised assistance [15]. Its real-time decision-making capability is ideal for making immediate adjustments to optimise support[14][15]. Furthermore, RL can handle the complexities of exoskeleton control by learning sophisticated policies that account for the nuances of human movement and mechanical constraints [14][15].

7 Conclusion

In this work, we developed a control system for an upper-limb exoskeleton. This technique leverages force myography (FMG) data for weight estimation, inverse dynamics, and an LSTM-based neural network to predict future joint torque requirements. The simulations show promising results but due to some implementation issues on the embedded LSTM, the test results for the torque prediction are not conclusive to either support or discard the method. Further testing and troubleshooting are needed.

Acknowledgements

This research group would like to thank our project supervisors Ming Shen and Shaoping Bai for their great guidance throughout the project. Additionally, we would like to thank PhD students Abdullah Tahir, and Yu Zhu for their contributions to our work.

References

[1] M. A. Gull, T. Bak, and S. Bai, "Dynamic modeling of an upper limb hybrid exoskeleton for simulations of load-lifting assistance," *Proc IMechE Part C: J Mechanical Engineering Science*, vol. 0, no. 0, p. 14, 2021.

[2] M. R. U. Islam and S. Bai, "Payload estimation using force myography sensors for control of upper-body exoskeleton in load carrying assistance," *Modeling, Identification and Control*, vol. 40, no. 4, p. 10, 2019.

[3] S. Bai, M. R. Islam, K. Hansen, J. Nørgaard, C.-Y. Chen, and G. Yang, "A semi-active upper-body exoskeleton for motion assistance," in *Wearable Robotics: Challenges and Trends*, J. C. e. a. Moreno, Ed. Springer Nature Switzerland AG, 2022, p. 5.

[4] A. K. Godiyal, M. Mondal, S. D. Joshi, and D. Joshi, "Force myography based novel strategy for locomotion classification," *IEEE Transactions on Human-Machine Systems*, vol. 48, no. 6, p. 10, Dec. 2018.

[5] G. Terry and T. Chopp, *Functional anatomy of the shoulder*, https://pubmed.ncbi.nlm.nih.gov/16558636/, Jul. 2000.

[6] Z. Zhang and E. C. Kan, "Novel muscle monitoring by radiomyography (rmg) and application to hand gesture recognition," *IEEE Transactions on Biomedical Engineering*, p. 10, 2023.

[7] E. Fujiwara, Y. T. Wu, and C. K. Suzuki, "Assessment of hand posture and grip force by optical fiber force myography sensor," in *2020 IEEE 2nd Global Conference on Life Sciences and Technologies (LifeTech)*, IEEE, Kyoto, Japan: IEEE, Mar. 2020, p. 2.

[8] X. Jiang, L.-K. Merhi, Z. Xiao, and C. Menon, "Exploration of force myography and surface electromyography in hand gesture classification," *Medical Engineering & Physics*, vol. 42, p. 11, 2017.

[9] U. Zakia and C. Menon, "Detecting safety anomalies in phri activities via force myography," *Bioengineering*, vol. 10, no. 3, p. 21, 2023.

[10] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review*, vol. 53, p. 28, 2020.

[11] A. O. Souza, J. Grenier, F. Charpillet, P. Maurice, and S. Ivaldi, "Towards data-driven predictive control of active upper-body exoskeletons for load carrying," in *International Conference on Advanced Robotics and its Social Impacts*, Berlin, Germany, Jun. 2023.

[12] A. Tahir, Z. An, S. Bai, and M. Shen, "Robust payload recognition based on sensor-over-muscle-independence deep learning for the control of exoskeletons," *IEEE Transactions on Circuits and Systems—II: Express Briefs*, vol. 70, no. 9, p. 5, Sep. 2023.

[13] P. S. Cowpertwait and A. V. Metcalfe, *Introductory Time Series with R*, 1st ed. New York: Springer, 2009, p. 81.

[14] M. Abdallah, M. Saad, R. Fareh, Y. Kali, and M. Bettayeb, "Reinforcement learning human inverse kinematics of an upper limb exoskeleton robot," in *2023 Advances in Science and Engineering Technology International Conferences (ASET)*, 2023, pp. 1–6.

[15] M. Oghogho, M. Sharifi, M. Vukadin, C. Chin, V. K. Mushahwar, and M. Tavakoli, "Deep reinforcement learning for emg-based control of assistance level in upper-limb exoskeletons," in *2022 International Symposium on Medical Robotics (ISMR)*, 2022, pp. 1–7.