Leveraging Large Language Models for Automated Feature Extraction and Model Training in EMG-Based Motion Decoding

Anany Dwivedi[1], Bonnie Guan[2], Gustavo J. G. Lahr[3], Mitchell A. Head[4], Jemma L. König[5], Mahonri W. Owen[6], Minas Liarokapis[2], and Albert Bifet[7]

***Abstract*— Feature extraction and model training are critical steps in developing machine learning models for electromyography (EMG) based motion decoding. Traditionally, these processes require domain expertise and programming knowledge to implement signal processing algorithms with optimized model training pipelines. In this work, we investigate the feasibility of using Large Language Models (LLMs) to automate both the extraction of features from EMG data and the development of machine learning models for decoding human motion with minimal human intervention. More specifically, we compare LLM extracted features and their corresponding motion decoding models against those developed using manually developed code. Our results indicate that LLM extracted features and their corresponding trained models achieve performance comparable to traditional methods, demonstrating the potential of accelerating research and scientific investigations with AI-driven biosignal processing. This study highlights LLMs' capabilities and limitations in replacing manual coding for developing muscle-machine interfaces and provides insights into their integration into biomedical signal analysis workflows.**

I. Introduction

Over the past few decades, robotic and mechatronic devices have been increasingly integrated into our daily lives, requiring the development of intuitive interfaces for effective human-machine interaction. Traditional interaction methods, such as buttons, hand-held controllers, and vision- or voice-based systems, are often inconvenient or cumbersome in practical and social contexts [1], [2]. To address these challenges, Muscle Machine Interfaces (MuMIs), which utilize different types of biosignals (e.g., electromyography (EMG) signals, lightmyography signals, forcemyography signals, etc.) to decode human intentions, have emerged as a promising alternative.

[1]A. Dwivedi is with Te Ipu o te Mahara Artificial Intelligence Institute, the University of Waikato, NZ. E-mail: anany.dwivedi@waikato.ac.nz

[2] B. Guan and M. Liarokapis are with the New Dexterity Research Group, Department of Mechanical and Mechatronics Engineering, The University of Auckland, NZ. E-mail: bgua324@aucklanduni.ac.nz, minas.liarokapis@auckland.ac.nz

[3] G. Lahr is with Hospital Israelita Albert Einstein, São Paulo, Brazil. E-mail: gustavo.lahr@einstein.br

[4] M. Head is with Te Kotahi Research Institute, University of Waikato, NZ. E-mail: mitchell.head@waikato.ac.nz

[5] J. König is with the School of Computing and Mathematical Sciences, University of Waikato, NZ.jemma.konig@waikato.ac.nz

[6] M. Owen is with the School of Engineering, University of Waikato, NZ. mahonri.owen@waikato.ac.nz

[7] A. Bifet is a Professor of Artificial Intelligence and the Director of Te Ipu o te Mahara Artificial Intelligence Institute at the University of Waikato, NZ, and a Professor of Big Data at Data, Intelligence and Graphs (DIG) LTCI, Télécom Paris, IP Paris. albert.bifet@waikato.ac.nz

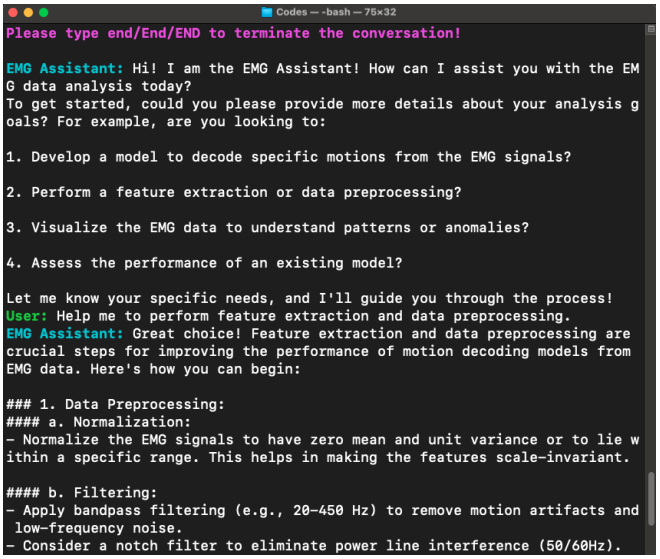

Fig. 1. Illustration of interaction with large language model (GPT-4o) for seeking assistance with Electromyography (EMG) signal analysis.

These hands-free interfaces enable natural and immersive interactions with various devices, making them suitable for applications such as teleoperation of robotic arms [3] and control of prosthetic devices [4]. Machine learning-based MuMIs can be categorized into classification-based and regression-based approaches [5]–[7]. Classification-based methods identify discrete user intentions, such as specific gestures or tasks, while regression-based methods provide continuous predictions of motion trajectories. Both approaches rely on a common processing pipeline that includes data acquisition, signal filtering, feature extraction, and intention decoding [8]. Effective filtering and feature extraction are critical for reducing signal noise, capturing relevant information, and meeting real-time application constraints. Feature extraction methods for EMG signals are typically classified into Time Domain (TD), Frequency Domain (FD), and Time-Frequency Domain (TFD) categories [9]. Among these, TD features, such as Root Mean Square (RMS), Zero Crossings (ZC), and Waveform Length (WL), have shown consistent performance for creating decoding models in various previous studies [10]–[12]. For real-time applications, the choice of features and their segment length significantly affects the performance of machine learning models [13].

Implementing feature extraction for EMG-based intention decoding with strict constraints can be a complex task that requires expertise in both signal processing and programming. Unlike structured datasets, EMG signals are inherently noisy, non-stationary, and influenced by various physiological factors such as muscle fatigue and sweating as well as by physical factors like electrode placement and movement artifacts [14]. Extracting meaningful features, such as RMS, WL, and ZC, involves applying mathematical transformations, windowing techniques, and filtering methods to capture relevant information while minimizing noise [15]. Writing efficient and reproducible code for this process demands significant effort, particularly for researchers unfamiliar with biosignal processing and/or new to programming. The complexity increases further when integrating feature extraction into machine learning pipelines, where incorrect implementation or suboptimal parameter selection can negatively impact classification accuracy and model generalization [16].

Large Language Models (LLMs) have demonstrated remarkable capabilities in understanding and generating human-like text, enabling them to perform a variety of tasks, including providing programming assistance. Models such as OpenAI's GPT-4 [17] have been trained on extensive datasets comprising natural language and code, allowing them to generate code snippets, provide explanations, and assist in debugging across multiple programming languages. This proficiency has been used in tools like GitHub's Copilot [18], which aids developers by suggesting code completions and generating entire functions based on contextual cues.

Recent studies have systematically evaluated the programming skills of LLMs. For instance, a comprehensive evaluation of models like GPT-4 and Gemini AI assessed their code generation capabilities, highlighting their strengths and areas for improvement in producing functional code [19]. Furthermore, the L2CEval benchmark systematically evaluated LLMs' language-to-code generation capabilities across various tasks, offering a comprehensive understanding of their performance in code generation [20]. Yu et al. [21] explored methods to fine-tune LLMs for enhanced code generation performance. Their study provides insights into improving the accuracy and efficiency of generated code through targeted fine-tuning approaches. As a result of the proficiency of LLMs in programming tasks, researchers have started exploring their applicability to biomedical signal analysis. A study by Chan et al. [22] demonstrated the application of LLMs in interpreting physiological signals, including electroencephalograms and electrocardiograms. Their system processes biomedical data and converts it into textual descriptions, which a fine-tuned ChatGPT model then analyzes to offer appropriate interpretations.

In this work, we compare the performance of LLM-generated and manually coded feature extraction techniques for EMG-based motion decoding. The goal is to assess whether LLMs can automate EMG feature extraction with minimal human intervention and without the need for explicit programming. To achieve this, we extract three widely used EMG features—RMS, WL, and ZC—and use them to train machine learning models for motion decoding. The extracted features and corresponding models are evaluated against those developed using manually written code, serving as the baseline. The baseline results are derived from our previous work [7], directly comparing LLM-assisted and traditional feature extraction approaches.

Fig. 2. Electrode placement positions for EMG data collection on the right arm-hand system. The double dot with the connected line represents a double-differential EMG electrode.

The rest of the paper is organized as follows: Section II provides details on the dataset, including the executed tasks and EMG electrode placements. Section III outlines the feature extraction methods and motion decoding models used in the baseline study and their application in this work. Section IV describes the prompting strategies used to guide LLMs in feature extraction and motion decoding. Section V presents and analyzes the results, followed by Section VI, which summarizes the findings and concludes the paper.

II. Dataset

This work employs the dataset used in our previous work [7] that presented a learning scheme for EMG-based decoding of object motions in the execution of dexterous, in-hand manipulation tasks. The dataset comprises experiments performed by 11 able-bodied participants (age = 26 ± 4, hand length = 176 ± 25), five males (hand length = 186 ± 15) and six females (hand length = 166 ± 6). The study was approved by the University of Auckland Human Participants Ethics Committee (UAHPEC) with the reference number #019043. Further details regarding the dataset can be found in [7].

During data collection, participants were asked to execute 3D equilibrium point manipulation tasks with three different object: a Rubik's cube, a chips can, and a custom cube with off-centered mass. Three manipulation motions (pitch, roll, and yaw) were performed with each of the objects. Regarding the data collection protocol, each motion recording included a rest period, followed by five motion repetitions per trial. Ten trials per motion and object were recorded with adequate breaks between trials so as to minimize muscle fatigue.

Myoelectric signals were collected from 16 muscles across the human hand and forearm. Double differential EMG electrodes were placed on the skin to ensure comprehensive coverage of relevant muscle groups. The electrode placement on the muscles can be seen in Fig. 2. For further details, including precise details on acquisition systems, electrode placements, experimental protocols, and muscle group summaries, please refer to [7].

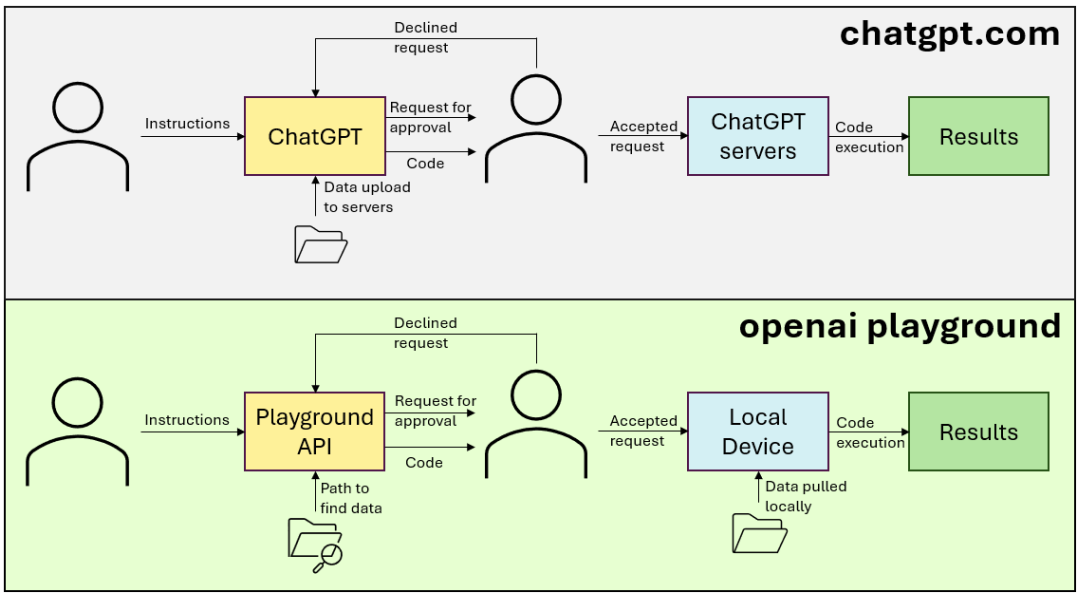

Fig. 3. Two potential approaches for using ChatGPT in EMG feature extraction and analysis. The top section represents usage of ChatGPT.com, where data and instructions are uploaded to OpenAI's servers, and code is executed remotely after user approval. The bottom section depicts the OpenAI Playground API approach, where data remains on the local device, offering enhanced control over sensitive information while maintaining similar interaction and execution steps. Both approaches involve user instructions, code approval, and result generation.

III. Feature Extraction and Motion Decoding

A. Feature Extraction

The previous study [7] employed a 200 ms sliding window with a 10 ms increment to segment the data, ensuring a balance between real-time constraints and biases and variance [9]. Three time-domain features were extracted: RMS for signal power estimation, WL for capturing signal complexity, and ZC for estimating muscle fatigue [10], [12], [23]. The RMS value is one of the most commonly used values in the time domain. It represents the square root of the average power of the signal for the given time period. The RMS value is defined as:

$$RMS = \sqrt{\frac{1}{N}\Big(\sum_{k=1}^{N}(x_k)^2\Big)}, \tag{1}$$

where N is the size of the window applied to the data. WL measures the complexity of the signal. It is a measurement of the amplitude of the waveform, frequency, and duration in a single parameter and is defined as:

$$WL = \sum_{k=1}^{N} |\Delta x_k|, \tag{2}$$

where $\Delta x_k = x_k - x_{k-1}$. ZC represents the number of times the signal crosses the zero value in a given time period. ZC can be used to estimate muscle fatigue [24]. The count of ZC is incremented when:

$$\begin{gathered} x_k < 0 \quad \&\& \quad x_{k+1} > 0 \\ \| \| \\ x_k > 0 \quad \&\& \quad x_{k+1} < 0 \\ \&\& \\ |x_k - x_{k+1}| > V_t \end{gathered} \tag{3}$$

where V_t is a voltage threshold, which is selected according to the noise in the signal. This algorithm increments ZC only if it falls outside the dead-zone as it tries to eliminate the effect of noise on the zero crossings.

B. Motion Decoding Model

Random Forests (RF)-based regression models were selected for the motion decoding tasks due to their robustness even with limited data availability. Further details regarding RF-based models can be found in [7].

IV. Language Model-based Feature Extraction and Motion Decoding

This study leveraged OpenAI's ChatGPT to generate feature extraction code for EMG signals, focusing on replicating the manually implemented feature set used in the baseline method. The goal was to evaluate ChatGPT's capabilities to generate and run functionally correct code for extracting time-domain features. For this work, OpenAI's GPT-4o model was used. This analysis using ChatGPT can be conducted in two primary ways: through OpenAI's servers on ChatGPT.com or locally via API interaction available at OpenAI's Playground. The former offers ease of use by offloading the entire processing pipeline to OpenAI's servers and downloading either the processed data or machine learning models trained using the processed data. This is especially useful in cases where computing resources are not available locally. However, this approach has limitations, including restrictions on the size of datasets that can be uploaded, as larger datasets may lead to processing timeouts or loss of context. Additionally, the online server is constrained by the availability of programming environments, with limited support for certain frameworks, such as PyTorch.

The latter approach allows users to integrate language models into their workflows while ensuring sensitive data remains local on their machines. This approach is particularly useful for handling large datasets that require high computational resources. Using the API, a custom module can extract Python code from ChatGPT's responses and execute it automatically on the local machine. A safety verification step can be introduced to mitigate security risks in which the user inspects and approves the generated code before execution. Once the module is run, this setup provides the user with the same conversational experience as ChatGPT.com but with greater control over data privacy and execution security. It also provides essential computational resources and eliminates timeouts from heavy computations. The implemented module can be accessed at: GPT-EMG-Analyser[1].

A. Feature Extraction

To ensure comparability with the baseline, ChatGPT was instructed to extract RMS, WL, and ZC from multi-channel EMG data. The prompts were designed to guide the language model in performing feature extraction similar to the baseline method. Instructions for feature extraction were:

"*Extract RMS, Waveform Length, and Zero Crossing with a window size of 200 ms and a window slide of 10 ms for the data with a sampling rate of 1200 Hz. The first column contains the labels, and the following 16 columns contain the EMG data points.*"

B. Model Training

To evaluate the performance of models generated by the language model without explicit programming, the experiments from Set 1 of the baseline study [7] were replicated. In this set, object-specific models were developed for each subject, and performance was assessed by measuring Pearson's correlation and Normalized Mean Squared Error (NMSE) between the predicted and ground truth motion. To do this, 10-fold cross-validation was performed, and the instructions given to the language model for this were:

"I have data in 3 batches in the folder Data/Batch1, Data/Batch2, and Data/Batch3. The data files have EMG data, where the user does in-hand manipulation motions. In this dataset, there are 3 different motions. One motion in each batch. Each file is named so that it has a number identifying the repetition number of that motion. Column 1 in each file is the target labels, and columns 2 through 17 are EMG activations from different muscles. Extract 3 time domain features from the EMG data, RMS, Waveform Length, and Zero Crossings with a window length of 200 ms and window slide of 10 ms. The sampling rate of the data acquisition was 1200 Hz. Train a Random Forest regression model with the feature extracted dataset. Perform a 10-fold cross-validation. Fold 1 means using data file 1 from each motion to test and remaining files to train. Fold 2 means to use data file 2 from each motion for test and remaining for training. Give me the code to process all the 10 folds and finally give mean Pearson's correlation and NMSE value across the 10 folds."

[1]https://github.com/adwi592/GPT-EMG-Analyser/

TABLE I
COMPARISON OF FEATURES EXTRACTED BY THE LANGUAGE MODEL VS. MANUALLY CODED FEATURE EXTRACTION

Metric	RMS		WL		ZC	
	Mean	SD	Mean	SD	Mean	SD
Correlation	1.00	1.37E-16	1.00	1.74E-16	0.99	3.68E-04
MSE	0.00	4.91E-28	0.00	2.33E-23	0.02	4.10E-02

V. RESULTS

Fig. 4 shows the interactive workflow of the ChatGPT-assisted system for EMG analysis, demonstrating how user-provided inputs are processed by the language model to produce functional code snippets, which are executed in to compute metrics such as NMSE and Pearson's correlation. Building on this workflow, this section compares the features extracted by the language model and manual coding, and presents the decoding accuracy results for subject-specific, object-specific, but motion-generic models (Set 1).

A. Feature Comparison

Table I presents the Pearson's correlation and Mean Squared Error (MSE) for features extracted using the language model and manual coding. The values represent the mean across all 11 subjects, each performing three different motions with three objects. The minor deviations in RMS and WL values are likely due to precision differences, while the small errors for ZC can be attributed to variations in the dead-zone value used for its calculation (see Eq. 3).

B. Model Training

The results in Table II show that the decoding models trained by the language model achieve comparable performance to the baseline feature extraction method across all objects, with minor deviations. For the Rubik's Cube, the decoding models trained by language model produced a Pearson's correlation of 81.94% and accuracy of 65.50%, closely matching the baseline (83.30% Corr and 67.32% Acc). Similar trends were observed for the Chips Can and Off-center Mass Cube, where correlations were 77.23% and 79.32% (compared to 77.58% and 79.71% for the baseline), and accuracies were slightly higher, at 57.07% vs 56.52%, for the chips can while slightly lower, at 61.97% vs 62.09% for the off-center mass cube. Despite these small discrepancies, the decoding models trained by the language model effectively replicate the motion decoding performance of the baseline method, demonstrating its potential for automating the development of motion decoding models for EMG-based with minimal manual intervention. Fig. 5 shows an example of the predicted motion with the actual motion.

To evaluate whether there was an overall difference in performance between the baseline and decoding models trained by the language model across all tasks and metrics, we conducted a one-way repeated measures ANOVA. The ANOVA compared the aggregated performance scores—which were computed as the mean of the six conditions (Rubik's Cube

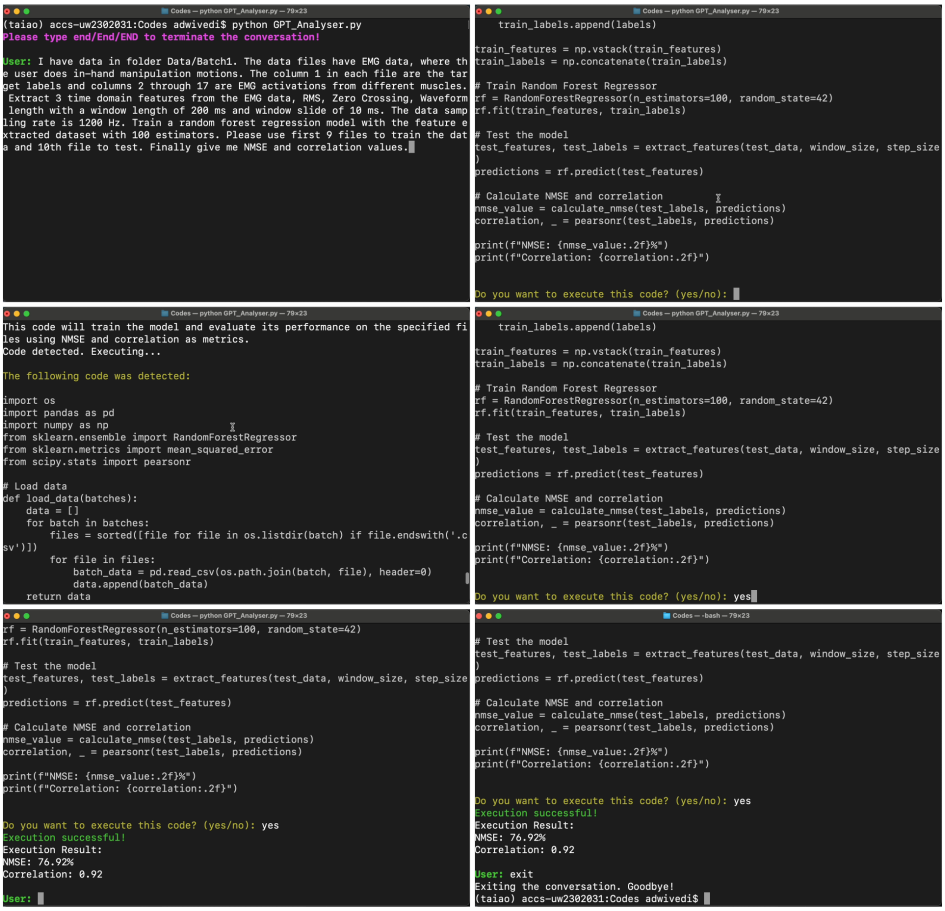

Fig. 4. The image demonstrates the ChatGPT-driven EMG feature extraction and model training process. The user provides input queries, such as data location and processing requirements, while the system generates and executes Python code snippets for EMG data analysis. Results, including NMSE and correlation values, are computed and displayed interactively, ensuring a seamless and efficient analysis workflow.

TABLE II

COMPARISON OF THE MOTION DECODING PERFORMANCE OF THE BASELINE MODELS WITH THE MODELS TRAINED BY THE LANGUAGE MODEL. THE EVALUATION METRICS FOR THE COMPARISON ARE PEARSON'S CORRELATION AND NMSE ACCURACY VALUE.

Subject	Rubik's Cube				Chips Can				Off-center Mass Cube			
	Baseline		GPT-4o-ChatGPT		Baseline		GPT-4o-ChatGPT		Baseline		GPT-4o-ChatGPT	
	Corr	Acc	Corr	Acc	Corr	Acc	Corr	Acc	Corr	Acc	Corr	Acc
1	92.60	83.61	89.61	78.13	76.00	52.40	74.92	51.21	75.67	59.60	74.51	57.68
2	83.62	67.91	79.87	68.38	76.72	55.68	75.04	53.51	75.55	55.02	73.03	53.34
3	87.06	74.26	85.35	73.89	79.77	60.41	78.41	69.83	86.60	72.15	86.91	73.12
4	61.64	32.49	60.22	31.65	74.24	49.32	74.21	49.55	53.45	27.01	51.06	26.91
5	87.86	75.01	87.65	75.56	77.37	54.95	76.45	53.21	77.84	56.72	74.36	53.08
6	79.64	57.93	78.92	59.71	76.15	53.64	77.15	54.45	83.86	66.65	85.42	67.15
7	89.61	77.91	87.08	75.41	78.60	59.72	79.34	60.23	88.49	75.09	89.54	76.34
8	85.78	72.18	84.25	70.29	83.13	67.35	82.31	67.67	86.98	73.11	84.85	71.18
9	87.27	73.64	88.32	64.36	81.44	63.64	85.42	69.04	85.15	69.38	83.81	68.49
10	78.36	57.88	78.27	57.51	70.97	45.99	69.56	43.21	79.30	60.14	85.75	65.78
11	82.89	67.71	81.82	65.61	78.98	58.61	76.77	55.89	83.94	68.09	83.33	68.65
Mean	**83.30**	**67.32**	**81.94**	**65.50**	**77.58**	**56.52**	**77.23**	**57.07**	**79.71**	**62.09**	**79.32**	**61.97**

Correlation and Accuracy, Chips Can Correlation and Accuracy, and Off-center Mass Cube Correlation and Accuracy) for each model. Results indicated that there was no statistically significant difference between model types in performance ($p = 0.16$). These findings indicate that, when considering the overall performance across the various conditions, the baseline and decoding models trained by language models perform similarly.

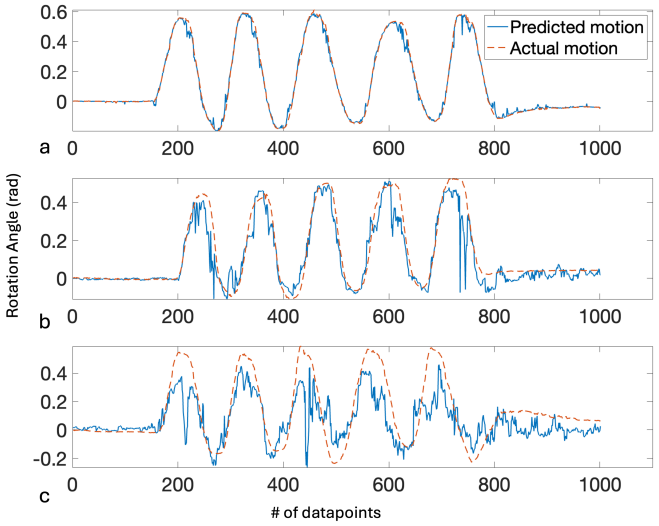

Fig. 5. Plots of actual vs estimated motion from the myoelectric activations using the motion decoding models trained by language models for subject 7. Subfigure A depicts the pitch motion, subfigure B depicts the roll motion and subfigure C depicts the yaw motion. The dashed lines represent the actual motion while the solid lines represent the predicted motion.

VI. Conclusion

This study demonstrates the potential of leveraging LLMs (e.g., OpenAI's ChatGPT or others) to automate feature extraction and develop motion decoding models for EMG-based interfaces. By replicating the baseline method [7] using LLM-generated code, we achieved comparable performance in feature extraction and motion decoding accuracy, with minor deviations attributed to precision differences and parameter variations. The results showed that the language model could effectively generate functionally correct code for EMG data processing with minimal human intervention, highlighting its utility for automating repetitive programming tasks. This approach's key advantage lies in its ability to streamline the development process, significantly reducing the effort required for novice programmers to perform manual coding. Moreover, the ability to execute the generated code locally via APIs ensures data privacy and facilitates handling large datasets that demand high computational resources. While the LLM performed well overall, feature precision and error handling areas warrant further refinement.

References

[1] A. Dwivedi, Y. Kwon, and M. Liarokapis, "EMG-Based Decoding of Manipulation Motions in Virtual Reality: Towards Immersive Interfaces," in *International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 3296–3303.

[2] F. Arena, M. Collotta, G. Pau, and F. Termine, "An overview of augmented reality," *Computers*, vol. 11, no. 2, p. 28, 2022.

[3] P. K. Artemiadis and K. J. Kyriakopoulos, "Emg-based position and force control of a robot arm: Application to teleoperation and orthosis," in *2007 IEEE/ASME international Conference on advanced intelligent mechatronics*. IEEE, 2007, pp. 1–6.

[4] M. Jafarzadeh, D. C. Hussey, and Y. Tadesse, "Deep learning approach to control of prosthetic hands with electromyography signals," in *2019 IEEE International Symposium on Measurement and Control in Robotics (ISMCR)*. IEEE, 2019, pp. A1–4.

[5] C. Loconsole, G. D. Cascarano, A. Brunetti, G. F. Trotta, G. Losavio, V. Bevilacqua, and E. Di Sciascio, "A model-free technique based on computer vision and semg for classification in parkinson's disease by using computer-assisted handwriting analysis," *Pattern Recognition Letters*, vol. 121, pp. 28–36, 2019.

[6] C. Castellini, A. E. Fiorilla, and G. Sandini, "Multi-subject/daily-life activity emg-based control of mechanical hands," *Journal of neuroengineering and rehabilitation*, vol. 6, pp. 1–11, 2009.

[7] A. Dwivedi, Y. Kwon, A. J. McDaid, and M. Liarokapis, "A learning scheme for emg based decoding of dexterous, in-hand manipulation motions," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 10, pp. 2205–2215, 2019.

[8] N. Jiang, S. Dosen, K.-R. Muller, and D. Farina, "Myoelectric control of artificial limbs—is there a need to change focus?[in the spotlight]," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 152–150, 2012.

[9] M. A. Oskoei and H. Hu, "Myoelectric control systems: A survey," *Biomedical Signal Processing and Control, Elsevier*, vol. 2, no. 4, pp. 275–294, 2007.

[10] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE Transactions on Biomedical Engineering, IEEE*, vol. 40, no. 1, pp. 82–94, 1993.

[11] C. Kendell, E. D. Lemaire, Y. Losier, A. Wilson, A. Chan, and B. Hudgins, "A novel approach to surface electromyography: an exploratory study of electrode-pair selection based on signal characteristics," *Journal of Neuro Engineering and Rehabilitation, Elsevier*, vol. 9, no. 1, p. 24, 4 2012.

[12] A. Turner, D. Shieff, A. Dwivedi, and M. Liarokapis, "Comparing machine learning methods and feature extraction techniques for the emg based decoding of human intention," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2021, pp. 4738–4743.

[13] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multifunction myoelectric control," *IEEE transactions on biomedical engineering*, vol. 50, no. 7, pp. 848–854, 2003.

[14] A. Phinyomark, F. Quaine, S. Charbonnier, C. Serviere, F. Tarpin-Bernard, and Y. Laurillau, "EMG feature evaluation for improving myoelectric pattern recognition robustness," *Expert Systems with applications*, vol. 40, no. 12, pp. 4832–4840, 2013.

[15] N. Parajuli, N. Sreenivasan, P. Bifulco, M. Cesarelli, S. Savino, V. Niola, D. Esposito, T. J. Hamilton, G. R. Naik, U. Gunawardana *et al.*, "Real-time emg based pattern recognition control for hand prostheses: a review on existing methods, challenges and future implementation," *Sensors*, vol. 19, no. 20, p. 4596, 2019.

[16] J. Shenouda and W. U. Bajwa, "A guide to computational reproducibility in signal processing and machine learning [tips & tricks]," *IEEE Signal Processing Magazine*, vol. 40, no. 2, pp. 141–151, 2023.

[17] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[18] GitHub, "Github copilot: Your ai pair programmer," 2025, accessed: 2025-02-07. [Online]. Available: https://github.com/features/copilot

[19] L. B. Heitz, J. Chamas, and C. Scherb, "Evaluation of the programming skills of large language models," *arXiv preprint arXiv:2405.14388*, 2024.

[20] A. Ni, P. Yin, Y. Zhao, M. Riddell, T. Feng, R. Shen, S. Yin, Y. Liu, S. Yavuz, C. Xiong *et al.*, "L2ceval: Evaluating language-to-code generation capabilities of large language models," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 1311–1329, 2024.

[21] Y. Yu, G. Rong, H. Shen, H. Zhang, D. Shao, M. Wang, Z. Wei, Y. Xu, and J. Wang, "Fine-tuning large language models to improve accuracy and comprehensibility of automated code review," *ACM transactions on software engineering and methodology*, vol. 34, no. 1, pp. 1–26, 2024.

[22] N. Chan, F. Parker, W. Bennett, T. Wu, M. Y. Jia, J. Fackler, and K. Ghobadi, "Medtsllm: Leveraging llms for multimodal medical time series analysis," *arXiv preprint arXiv:2408.07773*, 2024.

[23] D. Tkach, H. Huang, and T. A. Kuiken, "Study of stability of time-domain features for electromyographic pattern recognition," *Journal of neuroengineering and rehabilitation, BioMed Central*, vol. 7, no. 1, p. 21, 2010.

[24] G. Hägg, "Electromyographic fatigue analysis based on the number of zero crossings," *European Journal of Physiology, Springer*, vol. 391, no. 1, pp. 78–80, Jul 1981.