Tiny Machine Learning for Real-time Postural Stability Analysis

Veysi Adın
Department of Computer and Electrical Engineering
Mid Sweden University
Sundsvall, Sweden
veysi.adin@miun.se

Yuxuan Zhang
Department of Computer and Electrical Engineering
Mid Sweden University
Sundsvall, Sweden
yuxuan.zhang@miun.se

Bruno Andò
Department of Electric, Electronic and Information Engineering (DIEEI)
University of Catania
Catania, Italy
bruno.ando@unict.it

Bengt Oelmann
Department of Computer and Electrical Engineering
Mid Sweden University
Sundsvall, Sweden
bengt.oelmann@miun.se

Sebastian Bader
Department of Computer and Electrical Engineering
Mid Sweden University
Sundsvall, Sweden
sebastian.bader@miun.se

***Abstract*—Postural sway is a critical measure for evaluating postural control, and its analysis plays a vital role in preventing falls among the elderly. Typically, physiotherapists assess an individual's postural control using tests such as the Berg Balance Scale, Tinetti Test, and time up-and-go test. Sensor-based analysis is available based on devices such as force plates or inertial measurement units. Recently, machine learning methods have demonstrated promising results in the sensor-based analysis of postural control. However, these models are often complex, slow, and energy-intensive. To address these limitations, this study explores the design space of lightweight machine learning models deployable to microcontrollers to assess postural stability. We developed an artificial neural network (ANN) model and compare its performance to that of random forests, gaussian naive bayes, and extra tree classifiers. The models are trained using a sway dataset with varying input sizes and signal-to-noise ratios. The dataset comprises two feature vectors extracted from raw accelerometer data. The developed models are deployed to an ARM Cortex M4-based microcontroller, and their performance is evaluated and compared. We show that the ANN model has 99.03% accuracy, higher noise immunity, and the model performs better with a window size of one second with 590.96 us inference time.**

***Index Terms*—TinyML, postural sway, fall prevention, real-time postural assessment, embedded systems, machine learning**

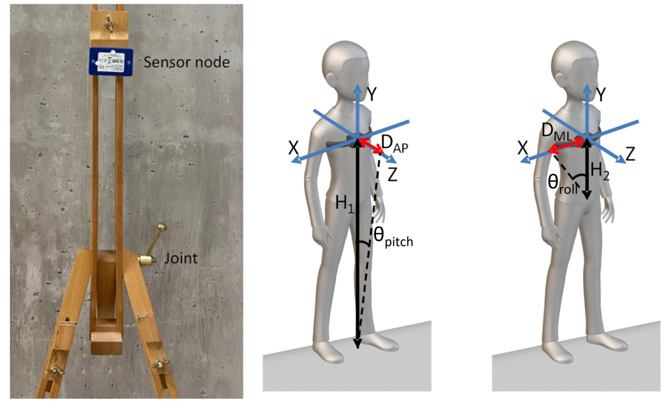
I. Introduction

Postural stability, or the ability to maintain the position of the body within specific boundaries of space known as stability limits, is essential for maintaining balance and controlling individual mobility [1]. Postural sway, the movement of the center of mass (COM) in a standing position, is minimized through the maintenance of postural control [2]. Individuals who have difficulty with balance and coordination tend to have increased postural sway [3]. Falls, often caused by balance impairments, are the leading cause of accidental death in older adults, with 28.7% of US adults over 65 years experiencing a fall each year, resulting in over 300,000 hip fractures [4] and $50 billion in medical costs [5]. Falls can lead to loss of mobility, anxiety, and reduced quality of life. Annual fall risk assessments become a standard part of care for older adults having diseases such as Alzheimer, Parkinson's disease, autism spectrum disorder, or multiple sclerosis. By analyzing the postural sway involved in postural control, it may be possible to develop more effective interventions to prevent falls and improve quality of life [6].

Traditional methods for assessing postural stability include physiotherapist- or neurologist-supervised tests such as the Berg balance test [7], Tinetti test [8], and time up-and-go test [9]. These tests are observation-based analyses and require expertise in the field. Another approach is to use force plates [10] to analyze the time variation of the body's center of pressure (COP). These assessments are usually conducted periodically in a structured environment, such as a medical clinic or hospital, under supervision and are therefore costly. To enable real-time, continuous monitoring of postural instability, a different approach is necessary. In [11], a vision-based approach using a Kinect depth camera is used to assess postural sway by localizing the center of mass and constructing its trajectories. Wearable devices based on inertial measurement units (IMU) have become increasingly common and can be used to detect user movement and extract information about postural sway [12]–[14]. For example, [13] presents a pendant-mounted wearable sensor for the assessment of postural sway, and the ability of the pendant sensor to discriminate between different balance conditions is evaluated. A comprehensive review for IMU usage in the assessment of standing balance can be found in [15]. The IMU-based postural assessment methods include statistical analysis of extracted time-based features such as the range of accelerometer signals in antero-posterior (AP) and medio-lateral (ML) direction and the root mean square (RMS) values of the range [16] or frequency-based features such as frequency dispersion and centroidal frequencies of the signals [17]. Although statistical analysis of postural assessment is fast, easy to implement, and explainable, the accuracy of this method is low and has generalization issues [18]. More recently, machine learning emerged as a solution to overcome the limitations of conventional statistical analysis methods. For example, in [19] the support vector machine (SVM) algorithm is used to automatically evaluate balance outside of clinical settings. In [20], the authors developed a deep learning model based on a long short-term memory (LSTM) network for the fall risk assessment using raw accelerometer data. In another work [21], authors proposed a novel neuro-fuzzy algorithm for the assessment of postural sway, and in [22] authors compared different methods for detecting unstable behaviors in postural sway. A comprehensive review of the use of machine learning for postural assessment can be found in [18]. Even though high accuracies are achieved with these machine learning-based solutions, they have not yet been deployed to wearable embedded systems. In a recent study [23], authors implemented their machine learning algorithm on a Raspberry Pi for detecting postural behavior. However, this solution has a high power consumption.

To conduct a comprehensive evaluation of postural behavior in elderly individuals, it is essential to monitor them within the comfort of their own homes. In order to address this issue, the development of cost-effective, dependable, and real-time monitoring solutions is imperative to accurately monitor postural behaviors in unstructured settings. Additionally, it is crucial to ensure that such solutions incorporate energy-efficient algorithms to minimize power consumption on resource-constrained microcontrollers, which leads to long-term, unattended operation. Therefore, in this paper, we focus on the development, implementation, and performance comparison of lightweight machine learning models that are deployable to a wide range of microcontrollers for the real-time assessment of postural behaviors. The implemented models are deployed to a widely available ARM Cortex M4-based microcontroller. We evaluated the performance of multi-layer perceptron (MLP), random forest (RF), Gaussian naive bayes (GNB), and extra tree (ET) classifiers under different input sizes, and different signal-to-noise ratios (SNR). To explore the trade-off between different machine learning models and the evaluation metrics, the implemented models are evaluated and compared in terms of test accuracy, inference time, and noise immunity. Thus, enabling machine learning based energy-efficient, real-time postural stability analysis running on microcontrollers.

The remainder of this paper is structured as follows. Section II describes the postural sway dataset used for the training, validation and test, and details the development, implementation, and deployment of the machine learning models. Section III presents the results of the model evaluation and the comparison between deployed models. Finally, the conclusion and future works are given in Section IV.



(a) (b)

Fig. 1. (a) The mechanism imitating the behaviors of standing postural sway (b) The corresponding location of the node and the depiction of key values that aid in the reconstruction of the AP and ML dynamics. [22]

II. Method

In this study, we utilized a postural sway dataset [22] to develop machine learning models. Specifically, we constructed an artificial neural network (ANN) model and employed AutoML to identify the most suitable traditional machine learning models. From the pool of candidate models, we selected the three best-performing traditional machine learning models, namely RF, GNB, and ET classifiers, for comparison with the ANN. By comparing the performance of the ANN with these three traditional machine learning methods, we aim to explore the trade-off between the ANN and the three selected traditional machine learning models in terms of test accuracy, inference time, which is a crucial metric for energy consumption, and noise immunity, which is a critical metric for robustness.

A. Dataset

The dataset used in this paper was initially described in [22]. As shown in Fig. 1 the data collection setup consists of a dedicated structure for emulating postural behaviors, as well as a sensor node containing an Arm Cortex M4-based STM32 microcontroller and a LIS2DW12 MEMS accelerometer. The accelerometer has a 16-bit resolution with configurable sampling frequencies ranging from 1.6 Hz to 1600 Hz. For the data collection, a sampling rate of 100 Hz was used. The structure was designed to simulate Stable (ST), Antero-Posterior (AP), Medio-Lateral (ML), and overall unstable (INST) behaviors. The node was positioned at a height corresponding to standard chest positions to acquire a consistent dataset of various standing postural behaviors without inconveniencing actual patients during development. Manual manipulation of the structure allowed for the emulation of different postural behaviors. Similar to the corresponding human movements shown in Fig. 2, ML displacement was achieved through tilting movements around the belt joint, while AP dynamics were reproduced through tilting movements around the bottom joint. Displacements exceeding a certain threshold in one direction

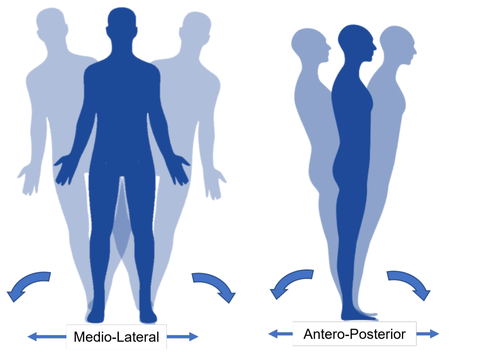


Fig. 2. Corresponding postural behaviors emulated by the mechanism

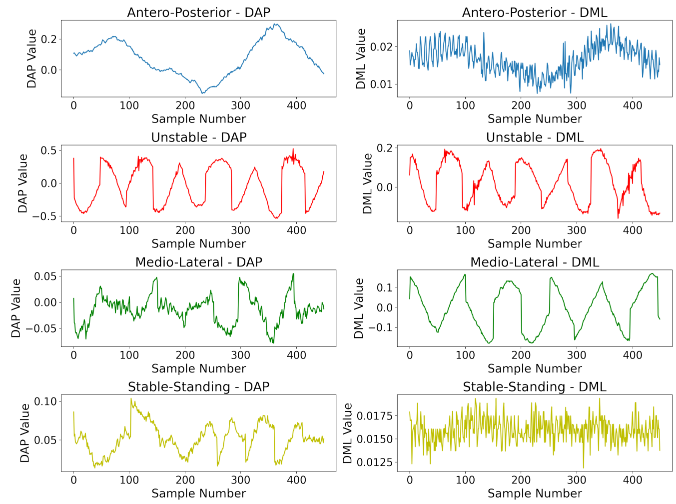


Fig. 3. Examples of DAP and DML signals for each postural behavior type in the dataset with 100 Hz sample rate.

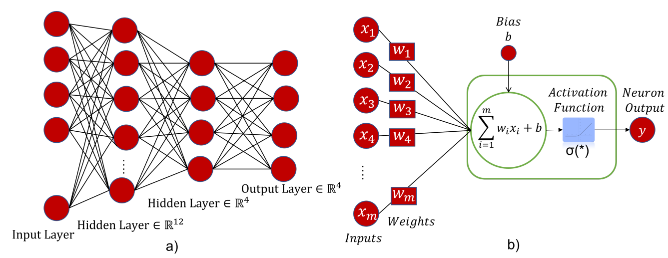


Fig. 4. a) Artificial neural network model structure b) Perceptron block

or AP-ML combinations were labeled as unstable behaviors while minor displacements were considered stable dynamics. More information about the setup can be found in [22].

From the collected accelerometer data, displacements in AP (DAP) and ML (DML) are calculated and used to populate the dataset. The trajectories of the sensor node's COM can be reconstructed using these displacements. Equations (1) and (2) describe the calculation of DAP and DML parameters, where $A_{x,y,z}$ are the acceleration in each direction, H_1 and H_2 are the distances described in Fig. 1.

$$DAP = H_1 \frac{A_z}{\sqrt{A_y^2 + A_x^2}} \tag{1}$$

$$DML = H_2 \frac{A_x}{\sqrt{A_y^2 + A_z^2}} \tag{2}$$

Each file in the dataset contains DAP and DML values with varying numbers of samples. The dataset contains 1000 emulated postural behaviors, ranging between $4.55\,\mathrm{s}$ - $8.69\,\mathrm{s}$, with around 250 for each behavior namely, ST, ML, AP, and INST. The dataset contains five different cases of H_1 and H_2 distances, representing different individuals. Examples of DAP and DML signals in the dataset for each postural behavior type are shown in Fig. 3.

The dataset is split into training, validation, and testing datasets with a ratio of 70%, 15%, and 15%, resulting in different numbers of samples for different input sizes.

B. Artificial Neural Network Model Design

Artificial neural networks (ANNs) are a form of machine learning that takes inspiration from the way biological neurons work. These networks are made up of layers of interconnected neurons, with each neuron serving as a node in the network. ANNs are typically composed of an input layer, one or more hidden layers, and an output layer. Each layer, except for the input layer, takes inputs from the previous layer, calculates the weighted sum of the inputs, adds a bias, and applies an activation function such as ReLU, sigmoid, or ELU to produce the output. Fig. 4(b) illustrates the calculation performed by each node, and its transfer function is given as

$$f(x) = \sigma\left(\sum_{i=1}^{m} w_i x_i + b\right) \tag{3}$$

where x_i are the inputs of the node, w_i are the weights, b is the bias, m is the number of inputs and finally σ is the nonlinear activation function.

The proposed ANN model is depicted in Fig. 5. It consists of an input layer with different input sizes, two hidden layers consisting of 12 and 4 neurons with ReLU activation functions respectively, and an output layer consisting of 4 neurons for the four different postural sway states with a respective softmax activation function for each output node. When selecting the ANN, a preference is given to the network with the fewest number of layers and neurons that still achieves the highest level of accuracy. This approach aims to ensure that the network remains as simple as possible, thereby allowing for a faster inference time and lower energy consumption.

C. AutoML: Building a classification model

Automated Machine Learning (AutoML) streamlines the application of machine learning to real-world problems by automating the entire process. AutoML enables users to rapidly identify the optimal neural network architecture for their data

TABLE I
AN EXAMPLE OUTPUT FROM PYCARET AUTOML LIBRARY WITH 1X900 INPUT SIZE

	Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (sec)
NB	Naive Bayes	**0.9971**	0.9995	**0.9971**	**0.9973**	**0.9971**	0.9962	0.9962	**0.0200**
RF	Random Forest Classifier	0.9957	**1.000**	0.9957	0.9959	0.9957	0.9943	**0.9964**	0.0550
ET	Extra Tree Classifier	0.9957	**1.000**	0.9957	0.9959	0.9957	0.9943	0.9944	0.0350

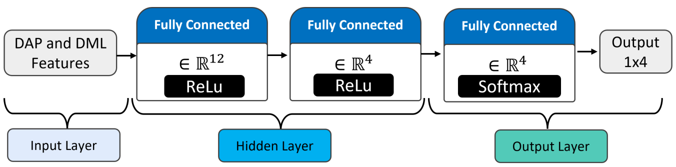


Fig. 5. Developed ANN model structure

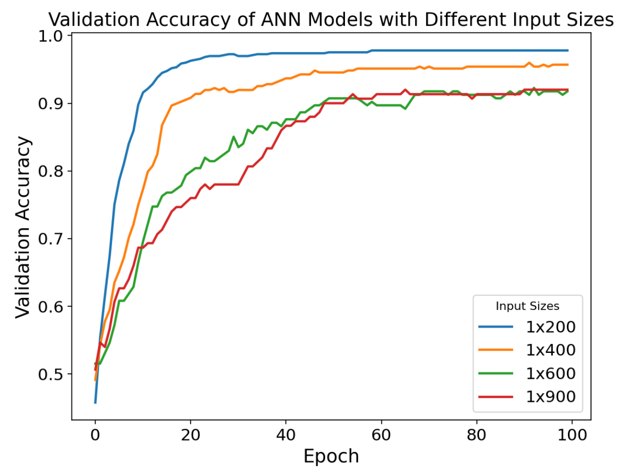


Fig. 6. Validation accuracy of the ANN models with different input sizes developed and implemented in Tensorflow

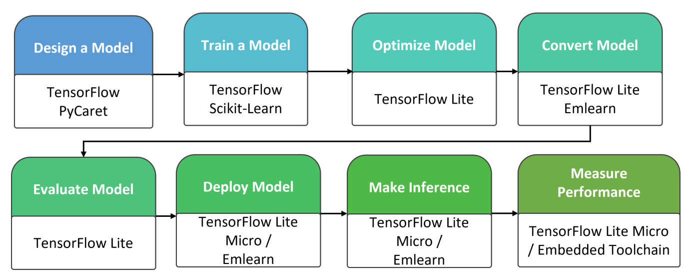


Fig. 7. TinyML model deployment workflow

and task, reducing completion time from hours to minutes. In this study, we used the PyCaret AutoML library [24] to search for the best classifiers for the given dataset automatically.

PyCaret is a low-code, open-source machine learning library written in Python that simplifies the automation of machine learning workflows [24]. PyCaret is a wrapper in Python that integrates several machine learning libraries and frameworks such as scikit-learn, XGBoost, LightGBM, CatBoost, spaCy, Optuna, Hyperopt, Ray, and more. It supports models that are available within these libraries. Example models include RF, ET, support vector machines (SVM), GNB, K-nearest neighbors classifier (KNN), etc. The three best-performing models from the PyCaret library applied to the postural sway dataset are shown in Table I. The three best-performing models are RF, ET, and GNB models and their performances will be compared to the developed ANN model. RF is a widely used machine learning algorithm that aggregates the predictions of multiple decision trees to produce a single outcome. It is capable of handling both classification and regression tasks. ET is an ensemble learning method that combines the results of multiple de-correlated decision trees in a "forest" to generate its classification prediction. Gaussian Naive Bayes is a classification algorithm that applies Bayes' theorem with strong independence assumptions. It is particularly useful for continuous data and assumes that the features follow a Gaussian distribution.

D. Model Training

The TensorFlow library is used to train the ANN model because it can be conveniently deployed to different microcontrollers using the TensorFlow Lite Micro (TFLM) framework [25]. The model is trained using sparse categorical cross-entropy loss as its loss function and the Adam optimizer for 100 epochs. The best learning rate and batch size are found to be 0.0005 and 32 respectively using the Weights & Biases [26] hyperparameter sweep interface. A fixed random seed is used for reproducibility so that the network can be retrained with the same results. The same model was trained with different input sizes using the same hyperparameters to find the optimum input window for the ANN. The used input sizes are 1x200, 1x400, 1x600 and 1x900 corresponding to 1 s, 2 s, 3 s and 4.5 s of concatenated DAP and DML signals sampled at 100 Hz respectively.

Similarly, different input sizes are used for training the three best performing models identified by the PyCaret AutoML library. These models, namely RF, ET, and GNB models, and their hyper-parameters are extracted using the PyCaret library. The models are retrained with the scikit-learn library and converted to microcontroller deployable format using the emlearn [27] library.

The validation results for the ANN models with different input sizes are shown in Fig. 6. The model achieves 99.58% validation accuracy with the 1x200 input size and 90.66% validation accuracy with an input of 1x900.

E. Model Deployment

The open-source TensorFlow Lite Micro (TFLM) [25] inference framework is utilized to deploy the ANN model to the

TABLE II
USED MCU SPECIFICATIONS

MCU	Core CPU	Frequency	SRAM	Flash	FPU	Current Consumption per MHz	Voltage	Compiler
nRF52840	ARM Cortex M4	64 MHz	256 KB	1 MB	✓	52 µA*	3.3V	arm-none-eabi-g++

*Bluetooth and Wi-Fi are disabled

TABLE III
MODEL TEST ACCURACIES AND INPUT SIZES

Input Size	**ANN**	**RF**	**ET**	**GNB**	**AutoML Max Accuracy**	**Info**	**Dataset Size**
1x200	99.03	99.29	**99.44**	98.76	99.44	1s DAP&DML	4832 cases
1x400	96.84	**99.69**	99.51	99.38	99.69	2s DAP&DML	2321 cases
1x600	95.89	**99.67**	**99.67**	99.56	99.67	3s DAP&DML	1296 cases
1x900	92.66	99.57	99.57	**99.71**	99.71	4.5s DAP&DML	1000 cases

target microcontroller.

For the deployment of RF, ET, and GNB models the emlearn [27] library is used. Emlearn [27] is a Python library that facilitates machine learning on microcontrollers and embedded systems. With emlearn, users can train models using Python and then run inference on any device equipped with a C99 compiler. Some of its notable features include being designed for embedded systems, having portable C99 code, not requiring libc or dynamic allocations, and supporting integer/fixed-point math (for some methods).

In our implementation, the developed ANN, RF, ET, and GNB models are deployed to an ARM Cortex M4-based NRF52840 microcontroller. The microcontroller's key properties are presented in Table II. The deployed models are evaluated based on test accuracy, average inference time, and noise immunity. The noise immunity is tested by adding random uniform noise to the test data. Fig. 7 illustrates the workflow from the design of the models to the performance measurements of the deployed models.

III. RESULTS AND DISCUSSION

The developed models are tested on the test dataset, and the result of the test accuracies with different input sizes are presented in Table III. RF, ET, and GNB models have accuracies of more than 98.5% on the test dataset, regardless of the input size, while the ANN model test accuracy is dropping with the increasing input size. This is likely due to the dataset getting smaller with increasing input size. Additionally, the models are tested with the DAP and DML inputs having different SNR values. Fig. 8 shows the model test accuracies as a function of the noise ratio. It can be seen that RF, ET, and GNB model test accuracies are dropping to less than 60% with the 1.53 SNR value, while the ANN model performance is more stable having 87.03% accuracy with the same noise ratio.

A. Deployment Results

The models are finally deployed to the ARM Cortex M4-based NRF52840 microcontroller, and average inference times are acquired for 1000 samples per model. The result of average inference time values is shown in Table IV. Inference time is an important metric for a machine learning model as it is

TABLE IV
MODEL INFERENCE TIMES AND INPUT SIZE

Input Size	**ANN** (µs)	**RF** (µs)	**ET** (µs)	**GNB** (µs)
1x200	590.96	**48.05**	90.08	2320.98
1x400	1017.07	**41.83**	74.45	4751.16
1x600	1447.04	**37.63**	71.18	7151.84
1x900	2093.30	**36.19**	61.11	10696.72

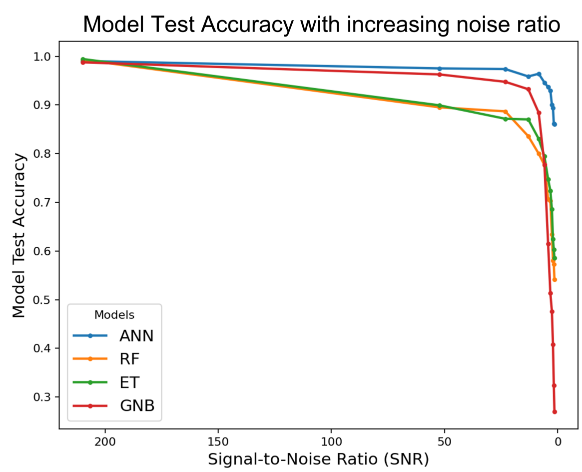


Fig. 8. Model test accuracies with increasing signal-to-noise ratio

linearly related to energy consumption [28]. According to the table, the ANN model inference time is getting slower with increasing input size due to the fact that the models' number of parameters is increasing. The ANN model has an average inference time of 590.96 µs with 1x200 input size and has an average inference time of 2093.30 µs with 1x900 input size. RF and ET models are getting faster with increasing input size and the RF model has the fastest inference time among the four models with 36.19 µs average inference time using 1x900 input size. The GNB model has the slowest average inference time with 2320.98 µs with 1x200 input size and has 10 696.72 µs average inference time with 1x900 input size. The RF model is faster compared to other models because it treats each input as a feature and creates a tree of if and else statements related to each feature and based on the result of if-else statements the output is acquired.

IV. Conclusions and Future Work

In this work, we compared different machine learning models using different input sizes for the real-time postural sway assessment on embedded systems. We developed tiny ANN, RF, ET, and GNB models for postural stability analysis using a postural sway dataset. The models are implemented, tested, and deployed to a microcontroller, and the performance of each model is evaluated. The four models are compared with each other in terms of accuracy, noise immunity, and inference time.

We showed that among the four machine learning models, RF has the fastest average inference time regardless of the input size, while the ANN model has the highest noise immunity and has an increasing average inference time with increasing input size. Additionally, we showed that the GNB model has the slowest inference time and lowest noise immunity among the four models deployed. Our findings indicate that in environments with high levels of noise, the implementation of artificial neural networks (ANN) proves advantageous due to the inherent robustness of the model, while the utilization of the RF model is deemed more suitable for applications in low-noise environments with limited battery capacity.

These models enable energy-efficient algorithms deployable on a wide range of embedded microcontrollers for real-time postural stability analysis applications.

In future work, the real-time performance of the implemented models will be tested on the data collection setup and on real subjects.

References

[1] M. Woollacott and A. Shumway-Cook, "Attention and the control of posture and gait: a review of an emerging area of research," *Gait & Posture*, vol. 16, no. 1, pp. 1–14, 2002.

[2] F. Horak, *Postural Control*. Springer Berlin Heidelberg, 01 2009, pp. 3212–3219.

[3] F. Reynard, D. Christe, and P. Terrier, "Postural control in healthy adults: Determinants of trunk sway assessed with a chest-worn accelerometer in 12 quiet standing tasks," *PLOS ONE*, vol. 14, p. e0211051, 01 2019.

[4] C. M. Casey, E. M. Parker, G. Winkler, X. Liu, G. H. Lambert, and E. Eckstrom, "Lessons learned from implementing cdc's steadi falls prevention algorithm in primary care," *Gerontologist*, vol. 57, pp. 787–796, 8 2017.

[5] C. S. Florence, G. Bergen, A. Atherly, E. Burns, J. Stevens, and C. Drake, "Medical costs of fatal and nonfatal falls in older adults," *Journal of the American Geriatrics Society*, vol. 66, no. 4, pp. 693–698, 2018.

[6] J. Johansson, E. Jarocka, G. Westling, A. Nordström, and P. Nordström, "Predicting incident falls: Relationship between postural sway and limits of stability in older adults," *Human Movement Science*, vol. 66, pp. 117–123, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167945718306973

[7] K. Berg, S. Wood-Dauphinee, J. Williams, and B. Maki, "Measuring balance in the elderly: validation of an instrument," *Canadian journal of public health*, vol. 83 Suppl 2, p. S7—11, 1992.

[8] S. Köpke and G. Meyer, "The tinetti test," *Zeitschrift für Gerontologie und Geriatrie*, vol. 39, no. 4, p. 288–291, 2006.

[9] B. M. Kear, T. P. Guck, and A. L. McGaha, "Timed up and go (tug) test: Normative reference values for ages 20 to 59 years and relationships with physical and mental health risk factors," *Journal of Primary Care & Community Health*, vol. 8, no. 1, pp. 9–13, 2017, pMID: 27450179. [Online]. Available: https://doi.org/10.1177/2150131916659282

[10] C.-H. Lee and T.-L. Sun, "Evaluation of postural stability based on a force plate and inertial sensor during static balance measurements," *Journal of Physiological Anthropology*, vol. 37, no. 1, 2018.

[11] S. Maudsley-Barton, M. Hoon Yap, A. Bukowski, R. Mills, and J. McPhee, "A new process to measure postural sway using a kinect depth camera during a sensory organisation test," *PLOS ONE*, vol. 15, no. 2, pp. 1–15, 02 2020. [Online]. Available: https://doi.org/10.1371/journal.pone.0227485

[12] M. L. Pollind and R. Soangra, "Mini-logger- a wearable inertial measurement unit (imu) for postural sway analysis," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2020, pp. 4600–4603.

[13] S. Lyu, A. Freivalds, D. S. Downs, and S. J. Piazza, "Assessment of postural sway with a pendant-mounted wearable sensor," *Gait & Posture*, vol. 92, pp. 199–205, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0966636221006123

[14] B. Andò, V. Marletta, S. Baglio, R. Crispino, G. Mostile, V. Dibilio, A. Nicoletti, and M. Zappia, "A measurement system to monitor postural behavior: Strategy assessment and classification rating," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 10, pp. 8020–8031, 2020.

[15] M. Ghislieri, L. Gastaldi, S. Pastorelli, S. Tadano, and V. Agostini, "Wearable inertial sensors to assess standing balance: A systematic review," *Sensors*, vol. 19, no. 19, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/19/4075

[16] J. J. Craig, A. P. Bruetsch, S. G. Lynch, F. B. Horak, and J. M. Huisinga, "Instrumented balance and walking assessments in persons with multiple sclerosis show strong test-retest reliability," *Journal of NeuroEngineering and Rehabilitation*, vol. 14, 5 2017.

[17] J.-H. Park, M. Mancini, P. Carlson-Kuhta, J. G. Nutt, and F. B. Horak, "Quantifying effects of age on balance and gait with inertial sensors in community-dwelling healthy adults," *Experimental Gerontology*, vol. 85, pp. 48–58, 2016.

[18] I. Bargiotas, D. Wang, J. Mantilla, F. Quijoux, A. Moreau, C. Vidal, R. Barrois, A. Nicolai, J. Audiffren, C. Labourdette, F. Bertin-Hugaul, L. Oudre, S. Buffat, A. Yelnik, D. Ricard, N. Vayatis, and P. P. Vidal, "Preventing falls: the use of machine learning for the prediction of future falls in individuals without history of fall," *Journal of Neurology*, 2 2022.

[19] T. Bao, B. N. Klatt, S. L. Whitney, K. H. Sienko, and J. Wiens, "Automatically evaluating balance: A machine learning approach," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 2, pp. 179–186, 2019.

[20] C. Tunca, G. Salur, and C. Ersoy, "Deep learning for fall risk assessment with inertial sensors: Utilizing domain knowledge in spatio-temporal gait parameters," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 7, pp. 1994–2005, 2020.

[21] B. Andò, S. Baglio, V. Di Bilio, V. Marletta, M. Marella, G. Mostile, S. Rajan, and M. Zappia, "A neuro-fuzzy approach to assess postural sway," in *2022 IEEE Sensors Applications Symposium (SAS)*, 2022, pp. 1–6.

[22] B. Andò, S. Baglio, S. Graziani, V. Marletta, V. Dibilio, G. Mostile, and M. Zappia, "A comparison among different strategies to detect potential unstable behaviors in postural sway," *Sensors*, vol. 22, no. 19, 2022.

[23] Q. Mascret, G. Gagnon-Turcotte, M. Bielmann, C. L. Fall, L. J. Bouyer, and B. Gosselin, "A wearable sensor network with embedded machine learning for real-time motion analysis and complex posture detection," *IEEE Sensors Journal*, vol. 22, no. 8, pp. 7868–7876, 2022.

[24] M. Ali, *PyCaret: An open source, low-code machine learning library in Python*, April 2020, pyCaret version 2.3.10. [Online]. Available: https://www.pycaret.org

[25] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang, P. Warden, and R. Rhodes, "Tensorflow lite micro: Embedded machine learning for tinyml systems," in *Proceedings of Machine Learning and Systems*, A. Smola, A. Dimakis, and I. Stoica, Eds., vol. 3, 2021, pp. 800–811.

[26] L. Biewald, "Experiment tracking with weights and biases," 2020, software available from wandb.com. [Online]. Available: https://www.wandb.com/

[27] J. Nordby, "emlearn: Machine Learning inference engine for Microcontrollers and Embedded Devices," Mar. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2589394

[28] L. Heim, A. Biri, Z. Qu, and L. Thiele, "Measuring what really matters: Optimizing neural networks for tinyml," *CoRR*, vol. abs/2104.10645, 2021.