EEG Wavelet Classification for Fall Detection with Genetic Programming

Jordan J. Bird
jordan.bird@ntu.ac.uk
Computational Intelligence and Applications Research Group (CIA), Nottingham Trent University
Nottingham, Nottinghamshire, United Kingdom

ABSTRACT

The ability to autonomously detect a physical fall is one of the many enabling technologies towards better independent living. This work explores how genetic programming can be leveraged to develop machine learning pipelines for the classification of falls via EEG brainwave activity. Eleven physical activities (5 types of falls and 6 non-fall activities) are clustered into a binary classification problem of whether a fall has occurred or not. Wavelet features are extracted from the brainwaves before machine learning models are explored and tuned for better k-fold classification accuracy, precision, recall, and F1 score. Results show that solutions discovered through genetic programming can detect falls with a mean accuracy of 89.34%, precision of 0.883, recall of 0.908, and an F1-Score of 0.895 from EEG brainwave data alone. All three genetic programming solutions chose a further step of Principal Component Analysis for additional feature extraction from the computed wavelet features, each with iterated powers of 6, 3, and 7, and all with a randomised Singular Value Decomposition approach. The best model is finally analysed via the Receiver Operating Characteristic and Precision-Recall curves. Python code for each of the genetic programming pipelines are provided.

CCS CONCEPTS

• **Computing methodologies → Machine learning**; **Machine learning approaches**; • **Human-centered computing**;

KEYWORDS

Fall Detection, EEG, Signal Processing, Signal Classification

1 INTRODUCTION

The ability to autonomously detect a physical fall is one of many enabling technologies towards better independent living. Many State-of-the-Art fall detection techniques are based on the detection of physical movements such as through accelerometers and gyroscopes, whereas many consider other traits such as bioelectrical activity from the muscles and brain. Applied machine learning is never perfect, and thus provision of multiple methods of fall detection reduces the potential error in the real world, since there are several observational models to consider rather than reliance on just one or a few. In the United Kingdom during 2021, there were more deaths registered than births [18] in part due to the world facing an ever-ageing population. The demographics of those who provide care and those who are service users are changing in size and pace at considerable rates throughout the world [5], and thus changes are required for healthcare systems throughout the world to continue to operate effectively and provide a suitable level of care to those who require it. A number of state-of-the-art solutions to these issues are presented in the form of applied artificial intelligence for independent assisted living [21, 24]. This work proposes the utilisation of a single electroencephalography electrode to detect the event of a fall autonomously through a process of data collection, feature extraction, processing, and machine learning. To detect a fall by this method would provide a further facet to independent assisted living and allow for further independence within the home. The main scientific contributions of this work are as follows:

(1) Exploration of brainwave features via Kullback-Leibler Divergence shows that the absolute mean of the 8^{th} wavelet and variance of the 3^{rd} wavelet hold the most information for fall classification.
(2) Balancing and normalisation provide an alleviation to the data scarcity of brainwave activity recorded during a fall event.
(3) Manual tuning of machine learning models presents a Gaussian Process as a candidate for fall detection.
(4) Genetic Programming to develop pipelines for better classification are successful, and the three solutions found outperform all other approaches explored within this work.

The remainder of this article is as follows; Section 2 explores the background and state of the art within the fields of study related to this work. Section 3 then describes the method of the experiments prior to the results being presented in Section 4. Finally, Section 5 concludes this study and suggests future work based on the findings.

2 RELATED WORK

Falls in older adults are caused in part by loss of balance due to ageing [8]. The risk of preventable injury by a fall grows with age, with around 33% of older adults experiencing a fall once or more per year, and around half of people over the age of 80 experience falls annually [29]. According to the NHS, falls do not often result in serious physical injuries in older adults, but can cause the person to lose confidence, withdraw socially, and feel like they have lost their independence [17]. It was noted in [2], that 0.1% of all healthcare expenditures in the United States and 1.5% in Europe are directly related to fall-related injuries. The review notes risk factors including impaired balance and gait, polypharmacy, history of previous falls, advancing age, sex, visual impairments, cognitive decline, and environmental factors. In the United States, there were an estimated 10,300 fatal and 2.6 million non-fatal fall-related injuries in the year 2000 alone [26]. The main goal of fall detection is the employment of technology to detect a fall event (abnormal behaviour recognition), leading to a quicker response

Table 1: Class labels applied to group the 11 individual activities found within the dataset [14].

Activity	Duration (s)	Class Label
Falling forward using hands	10	Falling
Falling forward using knees	10	Falling
Falling backwards	10	Falling
Falling sideward	10	Falling
Falling sitting in empty chair	10	Falling
Walking	60	Not Falling
Standing	60	Not Falling
Sitting	60	Not Falling
Picking up an object	10	Not Falling
Jumping	30	Not Falling
Laying	60	Not Falling

from carers, and alleviates issues in situations where the sufferer of the fall cannot locate or reach an emergency call button or cord [15].

Falls can be detected through a number of proposed methods including the analysis of wireless networks [31], computer vision [22], thermal image processing [16], acoustic classification [11], and activities recorded via wearable sensors [6]. Adkin et al. [1] note that compensatory balance reactions are recognisable within recorded EEG data. In [7], authors proposed a random forest ensemble for the classification of fall events and drowsiness with electrodes embedded within a helmet. The model achieved around 98% accuracy, but the authors note the exhaustiveness of the electrode array approach in terms of its computational complexity and thus inference time of the model, and the authors propose that future work may find more efficiency in an array of fewer electrodes. Annese et al. [3] proposed a multimodal approach to learning from EEG and EMG signals. In particular, seven electrodes are placed around the motor cortex and the occipital lobe. The results on the dataset were almost perfect, but similarly to the previous work, the authors note the computational expense of the approach. Given the level of consumer hardware available which could be provided by healthcare systems, a slow classification of an event would not solve the goal of quick response times that fall detection requires during real-world use. The NeuroSky EEG headset has a single electrode placed on the Fp1 position within the 10-20 EEG electrode placement system. Although many of the commercial applications of the device are based on the classification of concentration [33], the NeuroSky has proposed applications in fatigue detection [9], blink detection [23], and fall detection [14].

3 METHOD

The initial raw signals are collected from the *Preliminar Fall-UP Dataset* presented in [14]. The dataset is comprised of 11 activities performed by 4 subjects (three trials each). This work focuses only on the data recorded by the Neurosky MindWave EEG device, and all other features are disregarded. Table 1 details the binary classification problem that is formed from the dataset due to the consideration of whether a fall is occurring during the recording. Feature extraction in EEG is the process of deriving mathematical descriptions of sections of the wave for classification [4, 10, 12], and wavelet characteristics have been noted as informative descriptors of brainwave activity [27, 30]; EEG signals are divided into half-second windows, and seven sets of features are extracted, which leads to a dataset of 39 numerical features. The spectral entropies of the signals are computed via Fourier transform. The spectral entropy is given as $H = -\sum P(m) log_2 P(m)$ where P is the power spectrum and probability distribution of the input signal. Shannon entropy $H(X) = -\sum_n^i = 1P(x_i)logP(x_i)$ is also calculated. In terms of each wavelet scale, the following features are extracted via the continuous wavelet transform: absolute mean value, energy, entropy, standard deviation, and variance. After extraction, all numerical features are normalised to the range 0-1.

Prior to machine learning, the dataset is explored to discern how effective each attribute is for classification prediction. The information gain $IG(T, a) = E(T) - E(T|a)$ of each attribute is considered via observed changes in entropy $E(s) = -\sum_j p_j log(p_j)$. Hyperparameters for the KNN and Random Forest models are explored through a linear search $k = \{10, 20, 30, ..., 90, 100\}$ to discern whether hyperparameter tuning has a noticeable effect on mean classification metrics. Various machine learning algorithms are selected with a range of different statistical methods to provide a general overview of the classification ability using multiple methods (see Section 4.5 for more details). Following this, further tuning is performed via Adaptive Boosting [25] on all the selected models that are compatible with the algorithm (Random Forest, Logistic Regression, Naive Bayes, Stochastic Gradient Descent). Finally, a Genetic Programming approach is explored through a tree-based algorithm detailed in [19]; the algorithm is executed three times with random seeds equal to the iteration (1, 2, 3) and the source code is provided. All models are trained by 10-fold cross-validation with a seed set to 1 for randomisation and are therefore directly comparable. The algorithms were trained on an overclocked Intel Core i7-8700K CPU (4.3GHz) with scikit-learn [20] and TPOT [19].

4 RESULTS

In this section, the results of all planned experiments are presented. First, the information gain of the best features are noted prior to a a machine learning argument for class balancing and numerical normalisation are presented. Hyperparameter optimisation of select models is explored, and boosting is performed where possible. This section also details the results of genetic programming before giving a final comparison of all experiments performed in this work.

4.1 Data Preprocessing

The information gain (Kullback-Leibler divergence) of the top 5 features within the dataset by 10-fold cross-validation can be observed in Table 2. Prior to performing the experiments, Table 3 shows further details on the reasoning behind class balancing. When the dataset is unbalanced, there is a much higher frequency of EEG signals linked to activities under the category of *not falling*. Due to this, misleading results can be achieved; for example, even though the class balanced approach seemingly has a lower classification accuracy (83.3% vs. 92.21%), the ability to recognise the *falling* behaviour is improved from 885 correctly classified instances to 980. The baseline (prediction based on the most common label) for the balanced

Table 2: Top 5 features in the dataset by their Kullback-Leibler divergence after feature extraction and normalisation.

Attribute	KLD	Rank
Wavelet absolute mean_8	0.288 ± 0.004	2 ± 1.18
Wavelet variance_3	0.288 ± 0.006	2.4 ± 1.28
Wavelet variance_4	0.286 ± 0.005	2.5 ± 0.92
Wavelet standard deviation_5	0.284 ± 0.005	5.8 ± 4.66
Wavelet absolute mean_2	0.28 ± 0.006	12.1 ± 5.84

Table 3: Confusion matrices of balanced and unbalanced datasets following the training of a simple random decision tree. Due to the higher frequency of "Not Falling", classification without class balancing produces misleadingly high results.

Balanced (Acc 83.3%)			Unbalanced (Acc 92.21%)		
No Fall	***Fall***		***No Fall***	***Fall***	
856	246	***No Fall***	4771	261	***No Fall***
122	980	***Fall***	217	885	***Fall***

Table 4: Classification metrics on normalised and non-normalised numeric attributes with a simple random decision tree.

Normalised			Non-Normalised		
Precision	***Recall***	***F-Score***	***Precision***	***Recall***	***F-Score***
0.839	0.834	0.833	0.837	0.833	0.833

dataset is 50% whereas the baseline for the unbalanced dataset is 82.03% - thus, balancing in this preliminary example provides a 33.3% advantage over the baseline, whereas leaving the dataset unbalanced provides only a 10.18% advantage over the baseline. In Table 4, the classification metrics are compared when normalising the data within the range of 0-1. As can be observed for this preliminary decision tree classifier, the metrics increase slightly when normalisation is performed.

It is due to these examples and discussion that the normalised and equally balanced dataset is chosen for the remainder of the experiments presented in this work.

4.2 Hyperparameter Tuning

Figures 1 and 2 show the affects of the number of estimators in the Random Forest model. The best overall model was a random forest containing 80 decision trees, which had a mean accuracy of 84.94, a precision of 0.81, a recall of 0.915, and an F-Score of 0.856. These were the highest observed metrics within the linear search except for mean precision, where a Random Forest of 50 trees scored 0.81.

A similar linear search for the value of k within the K-Nearest Neighbour model can be observed in Figures 3 and 4. The most effective model was $k = 40$, which had a mean accuracy of 73.37, a precision of 0.793, a recall of 0.634 and an F score of 0.704.

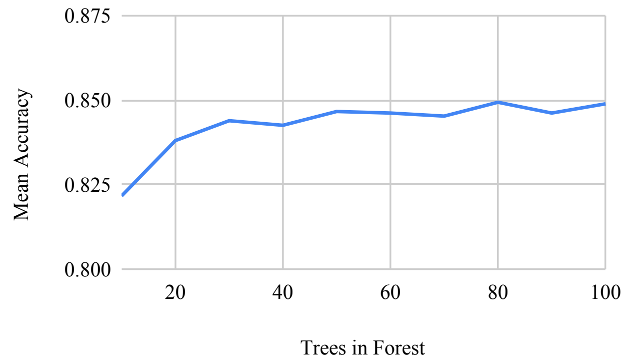

Figure 1: Effect of the number of estimators on the mean kfold accuracy of the Random Forest model.

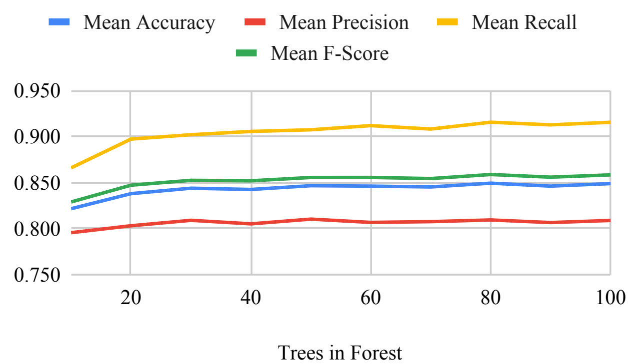

Figure 2: Effect of the number of estimators on the mean kfold accuracy, precision, recall, and F-Score of the Random Forest model.

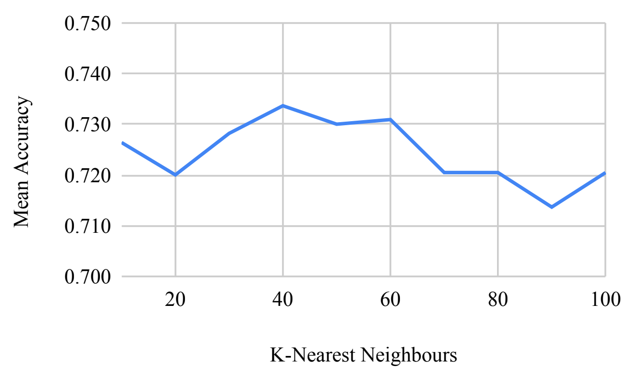

Figure 3: Effect of the number of K-Nearest Neighbours on the mean 10-fold accuracy of the K-Nearest Neighbours model.

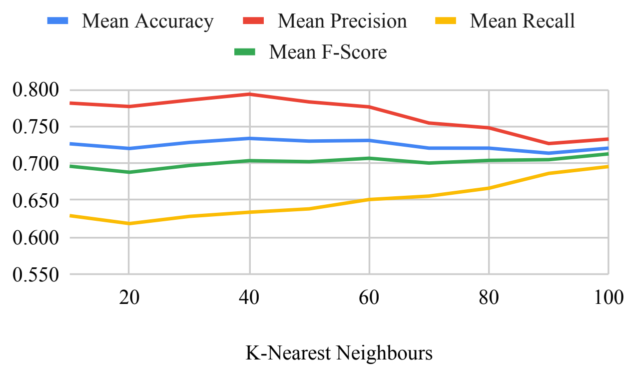

Figure 4: Effect of the number of K-Nearest Neighbours on the mean 10-fold accuracy, precision, recall, and F-Score of the K-Nearest Neighbours model.

Table 5: Results for the Adaptive Boosted models (Log. - Logistic Regression).

Model	Acc.	Prec.	Recall	F1
RF	84.71 (2.51)	0.81 (0.03)	0.908 (0.027)	0.856 (0.021)
Log.	61.57 (2.28)	0.575 (0.024)	0.881 (0.022)	0.696 (0.021)
SGD	59.84 (2.48)	0.564 (0.023)	0.846 (0.132)	0.672 (0.063)
NB	48.69 (5.49)	0.536 (0.257)	0.459 (0.425)	0.359 (0.283)

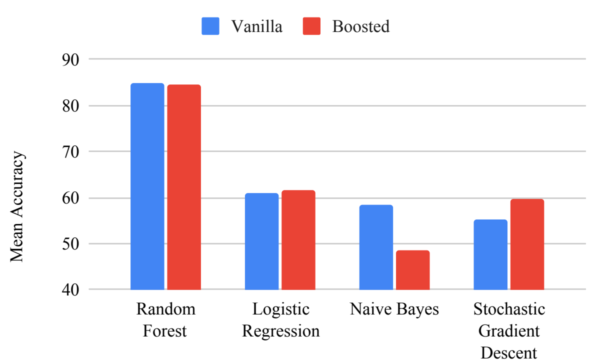

Figure 5: Comparison of models before and after being Adaptive Boosted.

4.3 Adaptive Boosting

The models which supported adaptive boosting due to their ability to predict probabilities are presented in Table 5. Figure 5 shows a comparison between the original model and the effect of adaptive boosting. It can be observed that adaptive boosting Random Forest and Naive Bayes models for this problem leads to a lower mean classification accuracy, whereas Logistic Regression and Stochastic Gradient Descent classification is improved. It must be noted here that although improvements were made in some cases, these were not competitive with the other results explored. Additionally,

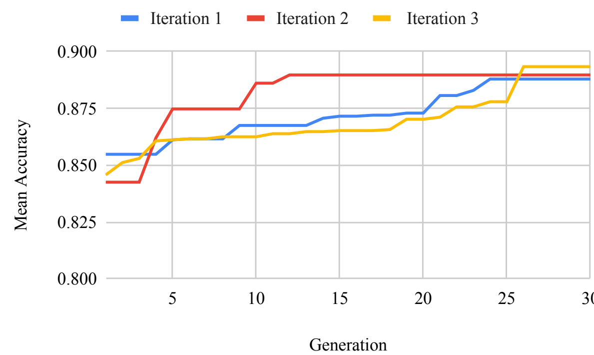

Figure 6: Best fitness (mean accuracy) observed during each generation for three genetic programming experiments.

Table 6: Classification metrics of the best solutions discovered after three individual runs for the genetic programming experiments.

GP	Accuracy	Precision	Recall	F1
1	88.79 (1.88)	0.892 (0.024)	0.889 (0.042)	0.888 (0.016)
2	88.97 (2.14)	0.882 (0.013)	0.901 (0.04)	0.891 (0.021)
3	89.34 (2.19)	0.883 (0.021)	0.908 (0.037)	0.895 (0.02)

Adaptive Boosting is computationally expensive compared to many of the approaches explored in this work.

4.4 Genetic Programming

As previously described, the genetic programming approach explored 30 generations with 20 solutions as a population size. The learning process for three iterations of the GP algorithm can be observed in Figure 6, and the best final solutions are detailed further in Table 6. Although starting at the highest fitness, iteration 1 had the lowest final score of 88.79%, with iteration 2 (which started at the lowest fitness) scoring slightly more by the end of the simulation at 88.79%. The best solution found was that by iteration 3, which scored 89.34%. Due to their complexity, the solutions are presented by their iteration ID in this work - the source code for all three machine learning pipelines can be found in Appendix A. Although features are extracted manually, it is interesting to note that all simulations decided upon further engineering through Principal Component Analysis (PCA); a number of related works have also proposed PCA as a dimensionality reduction technique to improve EEG classification [13, 28, 32].

4.5 Comparison of all Models

A final comparison of all models is provided in Table 7. As can be observed, the best models were all those that were explored through genetic programming. Though, it is worth noting that these models are relatively complex, whereas the Gaussian Process and Random Forest models are less computationally expensive but compete at -2.86% -4.4%, respectively. Interestingly, the adaptive

Table 7: Overall comparison of all fall detection models explored within this work.

Model	Accuracy	Precision	Recall	F1
Genetic Programming (3)	89.34 (2.19)	0.883 (0.021)	0.908 (0.037)	0.895 (0.02)
Genetic Programming (2)	88.97 (2.14)	0.882 (0.013)	0.901 (0.04)	0.891 (0.021)
Genetic Programming (1)	88.79 (1.88)	0.892 (0.024)	0.889 (0.042)	0.888 (0.016)
Gaussian Process	86.48 (2.65)	0.842 (0.044)	0.902 (0.033)	0.87 (0.024)
Random Forest	84.94 (2.39)	0.81 (0.03)	0.915 (0.027)	0.856 (0.021)
AB(Random Forest)	84.71 (2.51)	0.81 (0.03)	0.908 (0.027)	0.856 (0.021)
Extreme Gradient Boost	76.95 (3.16)	0.791 (0.038)	0.733 (0.036)	0.761 (0.033)
Adaptive Boosting	73.59 (4.12)	0.777 (0.058)	0.665 (0.051)	0.715 (0.045)
K-Nearest Neighbours	73.37 (3.29)	0.793 (0.049)	0.634 (0.046)	0.704 (0.038)
Linear Discriminant Analysis	64.93 (3.29)	0.611 (0.037)	0.824 (0.056)	0.7 (0.034)
AB(Logistic Regression)	61.57 (2.28)	0.575 (0.024)	0.881 (0.022)	0.696 (0.021)
Linear SVM	61.16 (1.92)	0.572 (0.022)	0.88 (0.018)	0.694 (0.019)
Logistic Regression	60.84 (1.92)	0.57 (0.022)	0.882 (0.02)	0.692 (0.019)
Radial Basis SVM	59.98 (2.15)	0.562 (0.024)	0.898 (0.021)	0.691 (0.021)
AB(Stochastic Gradient Descent)	59.84 (2.48)	0.564 (0.023)	0.846 (0.132)	0.672 (0.063)
Quadratic Discriminant Analysis	59.39 (2.08)	0.557 (0.025)	0.914 (0.015)	0.692 (0.021)
Naive Bayes	58.44 (1.78)	0.549 (0.021)	0.939 (0.016)	0.693 (0.019)
Stochastic Gradient Descent	55.17 (3.68)	0.381 (0.251)	0.625 (0.433)	0.468 (0.312)
AB(Naive Bayes)	48.69 (5.49)	0.536 (0.257)	0.459 (0.425)	0.359 (0.283)

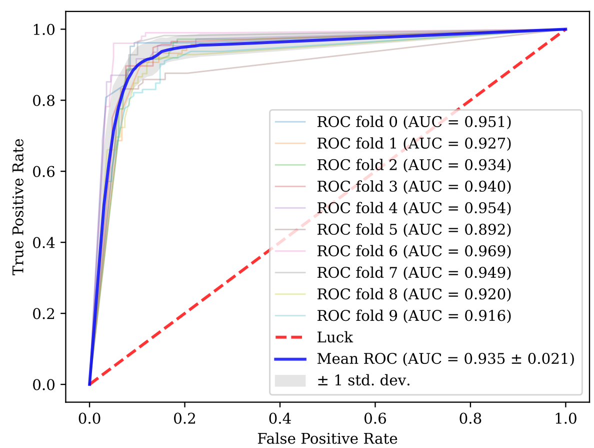

Figure 7: Receiver Operating Characteristic (ROC) curve for the best genetic programming-based solution.

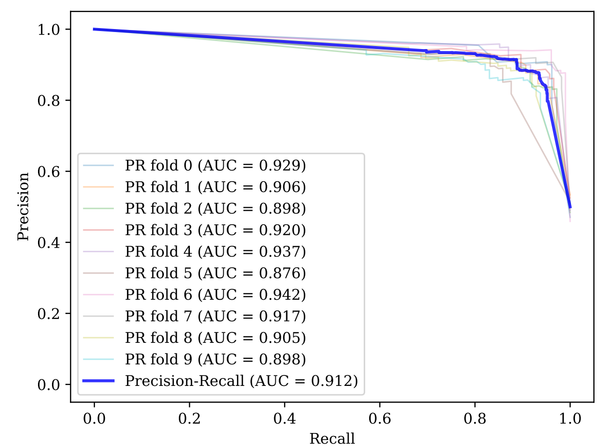

Figure 8: Precision-Recall curve for the best genetic programming-based solution.

boost of the Naive Bayes model was worse than random guessing, and this was the only instance of such an occurrence. The Receiver Operating Characteristic (ROC) and Precision-Recall curves for the best solution can be observed within Figures 7 and 8, respectively.

5 CONCLUSION AND FUTURE WORK

To finally conclude, this work has explored how machine learning and genetic programming can be leveraged to autonomously detect physical falls via a single electrode reading brain activity. Although the problem was difficult, due in part to activities such as *laying down* being present in the category of *not falling*, genetic programming developed a machine learning pipeline that could detect falls with an average accuracy of 89.34%.

The results presented in this work provide a good basis for future experiments, given that some approaches were particularly worse than the more impressive set of results. In the future, larger datasets could be leveraged to attempt a generalisation of the population. In particular, a larger dataset collected from a larger number of subjects would also enable leave-one-subject-out cross validation to test this. Additional ensemble methods could also be explored, as the genetic programming results seem to point towards a statistical ensemble being a particularly powerful method for EEG-based fall detection. In addition to the models explored, future work could involve the multimodal classification of falls by including information collected by other sensors e.g. those which are wearable and ambient sensors around the home environment. Finally, deep learning and data augmentation could be explored towards methods that can be tuned in the future as more data becomes available.

REFERENCES

[1] Allan L Adkin, Sylvia Quant, Brian E Maki, and William E McIlroy. 2006. Cortical responses associated with predictable and unpredictable compensatory balance reactions. *Experimental brain research* 172, 1 (2006), 85–93.

[2] Anne Felicia Ambrose, Geet Paul, and Jeffrey M Hausdorff. 2013. Risk factors for falls among older adults: a review of the literature. *Maturitas* 75, 1 (2013), 51–61.

[3] Valerio F Annese, Marco Crepaldi, Danilo Demarchi, and Daniela De Venuto. 2016. A digital processor architecture for combined EEG/EMG falling risk prediction. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 714–719.

[4] Jordan J Bird, Diego R Faria, Luis J Manso, Anikó Ekárt, and Christopher D Buckingham. 2019. A deep evolutionary approach to bioinspired classifier optimisation for brain-machine interaction. *Complexity* 2019 (2019).

[5] Jeffrey Braithwaite, Russell Mannion, Yukihiro Matsuyama, Paul G Shekelle, Stuart Whittaker, Samir Al-Adawi, Kristiana Ludlow, Wendy James, Hsuen P Ting, Jessica Herkes, et al. 2018. The future of health systems to 2030: a roadmap for global progress and sustainability. *International Journal for Quality in Health Care* 30, 10 (2018), 823–831.

[6] Jay Chen, Karric Kwong, Dennis Chang, Jerry Luk, and Ruzena Bajcsy. 2006. Wearable sensors for reliable fall detection. In *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*. IEEE, 3551–3554.

[7] Sameer Raju Dhole, Amith Kashyap, Animesh Narayan Dangwal, and Rajasekar Mohan. 2019. A novel helmet design and implementation for drowsiness and fall detection of workers on-site using EEG and Random-Forest Classifier. *Procedia Computer Science* 151 (2019), 947–952.

[8] David Eibling. 2018. Balance disorders in older adults. *Clinics in geriatric medicine* 34, 2 (2018), 175–181.

[9] Dhavalkumar H Joshi, UK Jaliya, and Darshak Thakore. 2015. Raw EEG-based Fatigue and Drowsiness Detection: A Review. *International Institute for Technological Research and Development* 1, 1 (2015).

[10] Sridhar Krishnan and Yashodhan Athavale. 2018. Trends in biomedical signal feature extraction. *Biomedical Signal Processing and Control* 43 (2018), 41–63.

[11] Yun Li, KC Ho, and Mihail Popescu. 2012. A microphone array system for automatic fall detection. *IEEE Transactions on Biomedical Engineering* 59, 5 (2012), 1291–1301.

[12] Fabien Lotte and Marco Congedo. 2016. EEG feature extraction. *Brain–Computer Interfaces 1: Foundations and Methods* (2016), 127–143.

[13] Kavita Mahajan, MR Vargantwar, and Sangita M Rajput. 2011. Classification of EEG using PCA, ICA and Neural Network. *International Journal of Engineering and Advanced Technology* 1, 1 (2011), 80–83.

[14] Lourdes Martínez-Villaseñor, Hiram Ponce, Jorge Brieva, Ernesto Moya-Albor, José Núñez-Martínez, and Carlos Peñafort-Asturiano. 2019. UP-fall detection dataset: A multimodal approach. *Sensors* 19, 9 (2019).

[15] Muhammad Mubashir, Ling Shao, and Luke Seed. 2013. A survey on fall detection: Principles and approaches. *Neurocomputing* 100 (2013), 144–152.

[16] Abdallah Naser, Ahmad Lotfi, and Junpei Zhong. 2022. Multiple thermal sensor array fusion towards enabling privacy-preserving human monitoring applications. *IEEE Internet of Things Journal* (2022).

[17] National Health Service. 2021. Falls. https://www.nhs.uk/conditions/falls/

[18] Office for National Statistics. 2021. Dataset: Vital statistics in the UK: births, deaths and marriages. https://www.ons.gov.uk/peoplepopulationandcommunity/populationandmigration/populationestimates/datasets/vitalstatisticspopulationandhealthreferencetables

[19] Randal S Olson, Nathan Bartley, Ryan J Urbanowicz, and Jason H Moore. 2016. Evaluation of a tree-based pipeline optimization tool for automating data science. In *Proceedings of the genetic and evolutionary computation conference 2016*. 485–492.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[21] Parisa Rashidi and Alex Mihailidis. 2012. A survey on ambient-assisted living tools for older adults. *IEEE journal of biomedical and health informatics* 17, 3 (2012), 579–590.

[22] Caroline Rougier, Jean Meunier, Alain St-Arnaud, and Jacqueline Rousseau. 2007. Fall detection from human shape and motion history using video surveillance. In *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*, Vol. 2. IEEE, 875–880.

[23] Mridu Sahu, Praveen Shukla, Aditya Chandel, Saloni Jain, and Shrish Verma. 2021. Eye Blinking Classification Through NeuroSky MindWave Headset Using EegID Tool. In *International Conference on Innovative Computing and Communications*. Springer, 789–799.

[24] A Hasan Sapci and H Aylin Sapci. 2019. Innovative assisted living tools, remote monitoring technologies, artificial intelligence-driven solutions, and robotic systems for aging societies: systematic review. *JMIR aging* 2, 2 (2019), e15429.

[25] Robert E Schapire. 2013. Explaining adaboost. In *Empirical inference*. Springer, 37–52.

[26] Judy A Stevens, Phaedra S Corso, Eric A Finkelstein, and Ted R Miller. 2006. The costs of fatal and non-fatal falls among older adults. *Injury prevention* 12, 5 (2006), 290–295.

[27] Abdulhamit Subasi. 2007. EEG signal classification using wavelet feature extraction and a mixture of expert model. *Expert Systems with Applications* 32, 4 (2007), 1084–1093.

[28] Abdulhamit Subasi and M Ismail Gursoy. 2010. EEG signal classification using PCA, ICA, LDA and support vector machines. *Expert systems with applications* 37, 12 (2010), 8659–8666.

[29] Mary E Tinetti, Mark Speechley, and Sandra F Ginter. 1988. Risk factors for falls among elderly persons living in the community. *New England journal of medicine* 319, 26 (1988), 1701–1707.

[30] Elif Derya Übeyli. 2009. Combined neural network model employing wavelet coefficients for EEG signals classification. *Digital Signal Processing* 19, 2 (2009), 297–308.

[31] Yuxi Wang, Kaishun Wu, and Lionel M Ni. 2016. Wifall: Device-free fall detection by wireless networks. *IEEE Transactions on Mobile Computing* 16, 2 (2016), 581–594.

[32] Xinyang Yu, Pharino Chum, and Kwee-Bo Sim. 2014. Analysis the effect of PCA for feature reduction in non-stationary EEG based motor imagery of BCI system. *Optik* 125, 3 (2014), 1498–1502.

[33] Biao Zhang, Jianjun Wang, and Thomas Fuhlbrigge. 2010. A review of the commercial brain-computer interface technology from perspective of industrial robotics. In *2010 IEEE international conference on automation and logistics*. IEEE, 379–384.

A PYTHON SOURCE CODE FOR THE GENETIC PROGRAMMING SOLUTIONS

This appendix provides the source code for the final solutions found by the three iterations of genetic programming. Python 3.x code is presented and is compatible with the scikit-learn library.

A.1 Iteration 1

```
iter1 = make_pipeline(
    make_union(
        StackingEstimator(estimator = make_pipeline(
            PCA(iterated_power = 6, svd_solver =
            ↪  "randomized"),
            ExtraTreesClassifier(bootstrap = False,
            ↪  criterion = "entropy", max_features =
            ↪  0.6500000000000001, min_samples_leaf
            ↪  = 5, min_samples_split = 11,
            ↪  n_estimators = 100)
        )),
        FunctionTransformer(copy)
    ),
    ZeroCount(),
    KNeighborsClassifier(n_neighbors = 64, p = 2,
    ↪  weights = "distance")
)
```

A.2 Iteration 2

```
iter2 = make_pipeline(
    PCA(iterated_power = 3, svd_solver =
    ↪  "randomized"),
    StackingEstimator(estimator = LogisticRegression(C
    ↪  = 10.0, dual = False, penalty = "l2")),
    GradientBoostingClassifier(learning_rate = 0.5,
    ↪  max_depth = 7, max_features = 0.5,
    ↪  min_samples_leaf = 2, min_samples_split = 8,
    ↪  n_estimators = 100, subsample = 1.0)
)
```

A.3 Iteration 3

```
iter3 = make_pipeline(
    PCA(iterated_power = 7, svd_solver =
    ↪  "randomized"),
    StackingEstimator(estimator =
    ↪  GradientBoostingClassifier(learning_rate =
    ↪  0.5, max_depth = 8, max_features = 0.5,
    ↪  min_samples_leaf = 12, min_samples_split = 7,
    ↪  n_estimators = 100, subsample = 1.0)),
    KNeighborsClassifier(n_neighbors = 63, p = 1,
    ↪  weights = "distance")
)
```