Optimising Machine Learning Models for Gesture Classification in Music

Samina Bibi[1], Chris Rhodes[2], Richard Allmendinger[3*]

[1] Alliance Manchester Business School (AMBS), University of Manchester, UK
saminabibi847@gmail.com

[2] NOVARS Research Centre, University of Manchester, UK
chris.rhodes@manchester.ac.uk

[3] Alliance Manchester Business School (AMBS), University of Manchester, UK
richard.allmendinger@manchester.ac.uk

* Address correspondence:
NOVARS Research Centre, University of Manchester, Bridgeford Street, Manchester, M13 9PL
chris.rhodes@manchester.ac.uk

Abstract

This paper aims to optimise parameters of several machine learning (ML) algorithms, available in Wekinator (an open-source ML software), to improve interactive music practice. ML algorithms can be used to predict performance gestures (i.e. movements used by a musician to create music) from complex biometric data, such as electromyographic (EMG) data, taken from a wearable sensor – termed a gestural interface (GI). EMG data is useful because it can be gathered in a non-intrusive manner and provides detailed information about performance gestures. However, different ML models (in Wekinator) have varying behaviours, which can have an impact on model accuracy when predicting gestures. Optimising ML models is important because modifying model parameters can improve model accuracy and behaviour, when used to classify performance gestures. Therefore, it is important to evaluate and compare ML models to improve their efficacy, when using GIs to recognise performance gestures in music. Current literature investigating ML practice in music focuses on using qualitative methods when evaluating model performance. Therefore, there is a lack of research into which ML algorithm is most efficient with EMG data analysis. In this paper, we use the Myo armband GI to extract significant features from EMG data, based on one performance gesture for piano, train ML models and then optimise models via modifying model parameters in Wekinator. ML models are then evaluated via a quantitative analysis. We find that optimising ML models clearly impacts model prediction accuracy. Therefore, model accuracy and behaviour will affect model choice in Wekinator for music composition practice.

1. Introduction

Interactive music practice is the use of software and computers to interpret live performances and "affect music generated or modified by a computer" (Winkler, 2001, p.4). Gestural interfaces (GIs) - wearable sensors used to track human movement and return biometric data - can be used within

live music performances to create a performer-computer relationship, forming a series of Human-Computer Interaction (HCI) events to be used for a musical output.

GIs allow music performers to interact with computers using gestures, juxtaposed to conventional HCI methods (i.e. keyboard and mouse). For digital systems to detect these gestures, different forms of biometric data have to be gathered from GIs, including EMG data. EMG data measures electrical activity from the muscles and is useful because such information can accurately inform musicians of how they interact with musical instruments (via performance gestures). EMG data is useful for musicians because it can more accurately represent their performance gestures, compared to other HCI peripherals used in music practice, such as cameras (e.g. the Leap Motion). EMG data is more accurate because it can directly access electrical signals from skeletal muscles and use them to represent user gestures, without environmental factors (e.g. light), affecting the information received. Information regarding a performer's gestures can then be used to stimulate interactive musical composition, as such gestures allow a performer to directly apply Digital Signal Processing (DSP) to EMG signals from a GI; therefore, allowing a performer to create sound through fine movement of the arms and create a process of musical embodiment. In recent history, EMG data has been used by musicians during musical performances. Atau Tanaka, for example, has used the 'BioMuse' to create musical works; an EMG based GI, using biometric signals to stimulate music (Bongers, 1998).

Alternative biometric data used in musical composition is electroencephalography (EEG) - measuring electrical activity of the brain. To gather EEG data, the user is required to wear an interface which attaches electrodes to the scalp, rendering the user immobile, albeit still resulting in interesting musical compositions (Hamadicharef, Xu & Aditya, 2010). The immobility of EEG interfaces makes it less ideal for use in musical composition and, as a result, makes it difficult for users to meaningfully embody the music they are creating. EMG data would be a better alternative to use, in comparison to EEG data, because GIs used to gather EMG data allow the user better mobility, leading to better creative expression within interactive music composition practice.

However, EMG data can be very complex. To reduce the complexity of EMG data, ML models have to be used to analyse EMG data and derive classifications based off which performance gestures are being performed. ML models are used for EMG data analysis in particular because manual EMG data analysis is far less efficient. Therefore, ML models can be used to automate the prediction process of music gestures via training ML models; here, via labelled inputs and outputs (i.e. supervised learning).

Current literature in the field, observing the use of GIs and ML, does not look at the efficacy of using multiple ML models when used with GIs and EMG data analysis (Caramiaux& Tanaka, 2013). Instead, such research investigates the improvement of gesture classification when a single ML model is provided with EMG data (Ruan, Wang, Liu, Fan & Liu, 2018; Zadeh, Calitz & Greyling, 2018). Whilst such research may constructively optimise a single ML algorithm, through changing the size of the training set (Ruan et al., 2018), there is a lack of research into the comparison of optimised ML models and observed accuracies when used to analyse EMG signals. This is a research problem because, when using a popular interactive ML software like Wekinator (Fiebrink & Cook, 2010), users are given multiple models to choose from. Therefore, particular ML models may be more accurate than others, when making a prediction, because of their data processing behaviours; resulting in disparate outputs. Furthermore, it then becomes

important to compare these models to understand the impact they could have on musical composition.

Moreover, this piece of research attempts to solve the problem that several ML algorithms are available for use within interactive music practice, when used to recognise performance gestures (via GIs and EMG data). We do this by quantitively evaluating all ML models available in Wekinator and applying parameter optimisation in order to achieve optimum results, as other research conducted in 2019 (Rhodes et. al.) has seemingly begun the discussion of empirically evaluating ML models for music practice (via Wekinator).

Our results show that changing the parameters of ML models can have a significant impact on model accuracy, however, some show no perceived change. This means that ML model choice is important, as well as the parameter values set for each model. Therefore, this research paper looks at how ML algorithms can be optimised through changing ML model parameters to analyse and classify EMG data gathered from GIs, for use in a musical context. The next section will discuss current literature in this field, then the methodology used in the project, followed by the results gathered, a discussion of the results and the conclusions derived following this.

2. Literature review

Investigating a topic concerning interactive music composition, ML and optimisation relies on a working knowledge of ML algorithmic behaviour, mapping, model efficacy and music aesthetics. Current literature in the field looks at how ML and GIs can be used in gesture recognition, for more intuitive experiences with HCI and music.

A review by Caramiaux & Tanaka (2013) provides an overview of ML techniques that have been used within interactive music practice and the New Interfaces for Musical Expression (NIME) community. The NIME community explores how advances in HCI can be merged with traditional music practice (Winkler, 2001). The ML techniques discussed in this study are classification and regression, which can be used for different tasks, as mentioned in Caramiaux & Tanaka's (2013) review. Classification problems require the model to categorise a piece of input data. However, regression problems require the model to find a relationship between an independent and dependent variable (Caramiaux & Tanaka, 2013). Caramiaux & Tanaka mention that "[their] review deliberately did not deal with other common ML tasks such as segmentation and clustering" because there is "less work in the NIME literature" (2013, p.7) that use these techniques. Caramiaux & Tanaka's paper is useful in providing an overview of the literature that explores the use of ML in interactive music and shows what is yet to be done in this field. The lack of specificity regarding the models used (i.e. several models can be used for classification or regression, as seen by the range of options available in Wekinator) suggests that there is a need to find an optimal ML model for analysing and processing EMG data, when identifying gestures for musical expression; this is elucidated when the same authors state "there is a need to create evaluation models to assess their suitability for real world music performance situations" (2013, p.6). This is indeed relevant to our study, as we evaluate how ML models perform when they are optimised, compared to when they are not, as well as their suitability for different musical contexts - as identified in Caramiaux & Tanaka's (2013) paper.

Feature extraction is an important aspect of reducing the complexity of data captured by GIs, such as EMG data. This is important, as ML models work better with less complex training datasets and therefore allow a model to make more accurate predictions, as patterns are better

identifiable by the model when there is less data variance. However, there are several feature extraction methods that can be applied to EMG data.

Arief, Sulistijono & Ardiansyah's (2015) paper on feature extraction methods (with EMG data) observes the suitability of numerous feature extraction models to determine which is the optimal feature extraction method. Arief et al.'s findings show that the optimal feature extraction is the mean absolute value (MAV). This is an important finding in the field and amongst HCI scholars working with EMG data. Due to its significance, we use this finding to post-process the dataset used in our investigation when applying the MAV feature extraction method (see Section 3.4). Due to the significance of this finding, this feature extraction method is used in several other works in the field, when working with EMG data. A paper by Zadeh et al (2018), on gesture recognition whilst using the Myo armband, uses the MAV feature extraction method under "the recommendation of Arief et al." due to its evaluated success (p.5). Similarly, authors Donato et al. also reference Arief et al.'s work when using the MAV feature extraction method with EMG data and music practice (Donato, Dooley, Hockman & Hall, 2017).

GIs, and ML, can be used to enable better sophisticated HCI within interactive music composition. Currently, there is a range of literature which observes the use of GIs and EMG data, either in a musical or unspecific context.

Zadeh et al.'s (2018) paper aims to evaluate a GIs capability (i.e. Myo armband) at identifying gestures when supplied with EMG signal data, using a single ML model. This is relevant because a single ML method is used (a feed-forward neural network - FNN) to analyse EMG data signals (Zadeh et al., 2018). The FNN was found in their study to have "an average task completion success rate of 95%" (p.9) when it was used to classify the gestures used. This shows there was an empirical evaluation of the ML model. However, the authors focused on user experience with the GI, in the context of HCI. This there shows that there is a lack of research into the empirical evaluation of several ML models used with GIs to analyse EMG data or optimise models. This is further supported by a table in their study regarding previous research projects evaluating GIs and their ability to recognise different gestures (2018). This table included the classifier ML model used, within the respective previous studies, and the accuracy of said model. This shows that there is no evidence of ML models undergoing an optimisation process, within the studies outlined in the table, as there is no attempt at modifying model parameters and observing an outcome. There is also no evidence of using several ML models to improve model accuracy regarding gesture classification. This illustrates the need for research into the effects of optimising ML models on accuracy when used to classify gestures from EMG data.

Furthermore, Donato et al.'s paper also evaluates a GI and presents the MyoSpat, an "interactive system that enables performers to control sound and light projections through hand-gestures" (2017, p.1). Their paper evaluates the HCI aspect of the system they created and focuses on user experience over model performance. The model used in their paper is a neural network (NN) available in Wekinator, specifically, a multi-layer perceptron (MLP). Whilst there are some empirical values of the model's accuracy when identifying gestures performed by participants, there is no explanation for model behaviour. Instead, the ML model is evaluated in terms of user experience (i.e. qualitatively) and to determine the usability of the MyoSpat in a musical context. This work also shows that there is a lack of empirical evaluation of optimised ML models, as there is no attempt at optimising the MLP used to improve model accuracy when classifying gestures. Instead, there is a focus on subjective user feedback of the model to evaluate

the GI and ML model's ability to classify gestures. This means that there is a lack of research that empirically evaluate different ML models when optimised and their suitability for different musical contexts when used to analyse EMG data and classify gestures. Studying this will be useful because it would allow musicians to have a better understanding of ML models, allowing them to take advantage of model behaviour for use in musical composition.

Ruan et al.'s (2018) study on finger movement classification aimed to improve the accuracy of finger movement classification through the analysis of EMG signals, where GIs were also evaluated when classifying gestures. Experiments conducted by the authors evaluated the performance of their chosen GI (the Myo armband). The overall accuracy of using an "EMG based finger movement detection" interaction method was 82.6% (p.1). Similar to Donato et al.'s paper, Ruan et al. focus on the HCI aspect of GIs to classify gestures. A user study was also carried out to evaluate the performance of the model used (an MLP). They used this MLP to classify different gestures, which represent the plucking actions of a ukulele. To improve the accuracy of the MLP, they adjusted the size of the training set from 60 to 240 (p.5). However, optimisation of the MLP, via the modification of model parameters, was not conducted in order to improve model accuracy. This shows that there is a need to investigate how optimising ML model parameters can affect model accuracy when used in EMG data analysis. The authors also did not optimise other ML models used in the study (i.e. adjusting the training dataset) as they only used one model to do this. This is important to note as other ML models could be more efficient than the one used. Therefore, there is evidently a gap for the evaluation of multiple ML models (when optimised) to find the most suitable ML model when classifying gestures in music.

In conclusion, it can be observed that the literature studied - investigating the use of GIs and ML to classify gestures – tend to focus on HCI and user experience when evaluating a GI and ML model. This shows that there is a gap in the literature for the empirical evaluation of multiple ML models (especially when optimised) and their accuracies when classifying gestures. Therefore, conducting research in this area would better inform music performers that are wanting to utilise GIs and ML within interactive musical composition, as it would provide a greater understanding of ML model behaviour (covered in Section 5).

3. Methodology

The methodology behind evaluating the efficiency of ML models in Wekinator when they are optimised (via the modification of model parameters), are as follows.

3.1 Gestural interface: Myo armband

We will be using the Myo armband GI (see Figure 1) to access raw biometric data (EMG) needed to investigate how optimising ML models can affect model accuracy, when predicting performance gestures.

Figure 1. The Myo armband

The Myo armband can gather two types of biometric data: inertial measurement units (IMUs) and EMG data. However, in this project, we will be using the Myo to access solely raw EMG data. EMG data is gathered from eight EMG electrodes placed around the Myo, which can be seen in Figure 1.

The Myo needs to be calibrated for efficient use because of physiological differences, not doing so would lead to unreliable data being gathered (Rhodes, Allmendinger & Climent, 2019). Calibrating the Myo will help lessen variance between EMG data gathered from different users and therefore will improve the efficacy of data processing.

The Myo collects IMU data at a sampling rate of 50Hz and 200Hz for EMG data (Nyomen, Haugen & Jensenius, 2015). The Myo can access 8 channels of 8-bit EMG data (Nyomen et al., 2015). When communicating biometric data from the Myo to the computer, Bluetooth is used, which is operated at a frequency between 2.402GHz and 2.480 GHz (North, n.d.). However, for optimal performance, a distance less than 15m must be kept (North, n.d.).

3.2 Data acquisition

This study uses a single performance gesture EMG dataset collected by Rhodes et al. (2019), via the Myo GI, in their study on comparing the efficacy of several ML models in music. Datasets for two performance gestures were gathered during Rhodes et al.'s investigation (see Figure 2 for the data acquisition process used in their research). However, in this investigation, we will be using the dataset of the second performance gesture used in their study; this gesture was performed by extending the fingers on the right-hand and returning the arm to a limp position (see Figure 3).

3.3 Data post-processing

In Rhodes et al.'s (2019) study, the dataset of the single performance gesture shown in Figure 3 had gone through a post-processing stage. This involved "taking an average of each EMG parameter" (p.6) and applying an absolute value (ABS) function to the dataset, ensuring the polarity of every data value was positive. However, in this study we will be applying our own post-processing stage to the raw dataset acquired in Rhodes et al.'s investigation. This is because the post-processing stage, used in Rhodes et al.'s study, did not involve the use of a MAV function (finding the mean of absolute values). As a result, model accuracies would not be optimal, as the MAV has shown to be the optimal feature extraction method when analysing EMG signals; supported by Arief et al.'s (2015) research into feature extraction with EMG data. Therefore, we will be extracting the MAV from the raw dataset in this project, which will help improve the accuracy of the ML models.

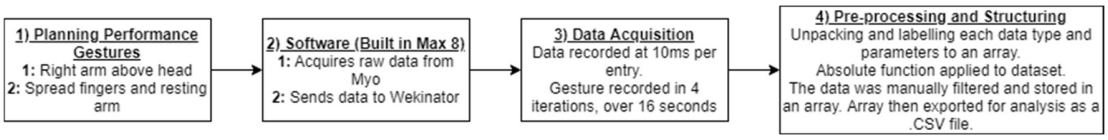

Figure 2. Flowchart summary of Rhodes et al.'s (2019) methodology and data acquisition process.

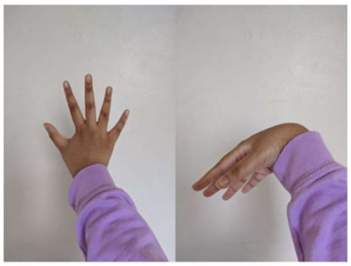

Figure 3. Pictures illustrating the EMG performance gesture used in Rhodes et al.'s 2019 study. a) shows the arm with fingers spread outwards. b) shows the arm in a limp, resting position.

In this investigation, we post-process Rhodes et al.'s dataset (for the performance gesture shown in Figure 3) by extracting the MAV feature from the dataset. The MAV is the mean of all absolute values in the dataset. The reason for extracting the MAV is that it is the optimal feature extraction method for reducing the complexity and variance of EMG data (Arief et al., 2015), allowing for better model accuracy; as more accurate predictions can be created from a dataset with lower variance. However, reducing data variance too much (via the MAV) could also cause overfitting. This is when the model is too biased towards a certain output (Webb, 2010). If the model is too biased, it can result in the model creating incorrect predictions, reducing the accuracy of the model.

We used a manual process to extract the MAV; this involved opening a csv file containing the pre-processed (raw) EMG performance gesture dataset (taken from Rhodes et al.'s 2019 study) and applying the MAV function to the dataset via Microsoft Excel. To calculate the MAV, a moving average was implemented at intervals of 20 samples. An interval of 20 samples was used because calculating a moving average with a lower interval, such as 1, would not reduce the variance within the dataset; therefore, such a moving average would not decrease the resolution of the dataset enough to see a meaningful representation within the data. However, having too high an interval, such as 100, will reduce data variance too much, resulting in biased outputs when the model is faced with unknown data inputs.

A sample interval of 20 was used for the moving average because the sampling rate of the Myo is 200Hz (i.e. 20ms per sample), and the pre-processed dataset, acquired during Rhodes et al.'s (2019) study, recorded the data at 100Hz (i.e. a sampling rate 10ms per sample). Therefore, an interval of 20 samples represents a time length of 200ms, which is equivalent to the Myo's sampling rate of 200Hz.

When applying the moving average function in Microsoft Excel, the function did not find numerical values for the first 20 samples, leaving them null. This is because the first 20 samples would be used to calculate the first value for the MAV, which was our determined interval.

Therefore, the moving average function would not produce any outputs until the first interval of 20 samples have passed, meaning the first 20 values for the MAV were void. To solve this, the first numerical value calculated from the moving average function were replicated to the first 20 samples in order to ensure there is a numerical value for every sample. This is a suitable solution, as the first numerical value after finding the moving average is the average of the first 20 samples, which means it can be used to replace the null values in the dataset.

In this study, we have used both a pre-processed performance gesture dataset (acquired from Rhodes et al.'s 2019 study), and a post-processed version of this dataset, so that we can clearly see any differences in the accuracies of the models before – and after – the MAV was applied to the original (pre-processed) dataset.

3.4 ML models in Wekinator

Wekinator software (Fiebrink & Cook, 2010) will be used to train and run ML models in this study. This piece of software is built upon the WEKA framework. The WEKA framework is used as a research tool for data science and provides a set of different ML algorithms to "aid in the application of machine learning techniques to a variety of real world problems" (Homes, Donkin & Witten, 1994). Wekinator builds upon the WEKA framework to provide a software which musicians can use to intuitively train and run ML models via a GUI (see Figure 4 for an example of the interface). The models that will be used in this project are supervised learning models. There are two types of supervised models to use in Wekinator: classifiers and continuous. All such available models in Wekinator are as follows.

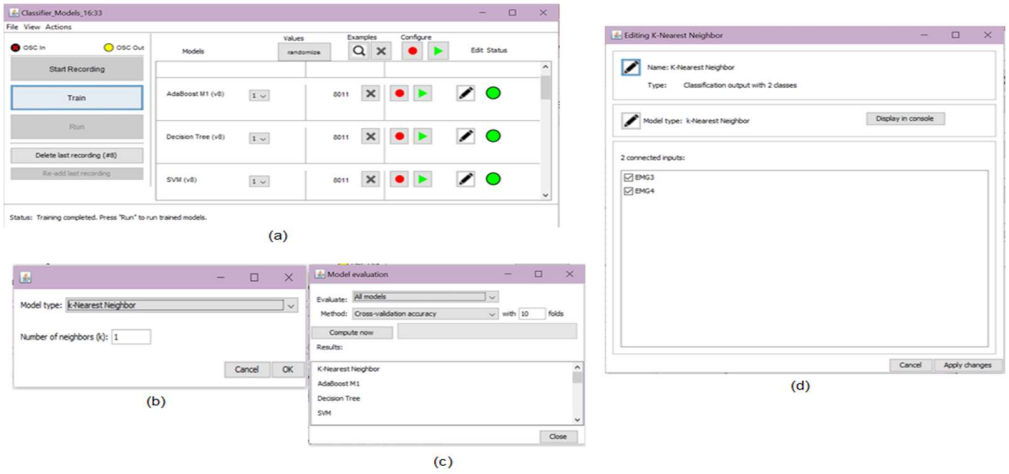

Figure 4. (a) This shows all the trained models, and several buttons which can be used by the user to interact with the ML model, via new windows that open, (b) this shows the window that is used to change the parameters of the model, (c) this shows the model evaluation interface, (d) this shows the interface for changing different aspects of the model, including name, model type, etc.

- **Classifier models**:
    - Support vector machine (SVM)
    - K-nearest neighbour (k-NN)
    - AdaBoost.M1
    - Decision tree
    - Decision stump
    - Naive Bayes
- **Continuous models**:
    - Polynomial regression (PR)
    - Linear regression (LR)
    - Neural Network (NN)

Table 1. Table of classifier models available and their available parameters in Wekinator.

Model	Available Parameters	Data Type of Parameter	Default Model Parameter Values
K-Nearest Neighbour	No. of neighbours	Integer	1
AdaBoost.M1	Training Rounds	Integer	100
	Base Classifier	Decision Tree/Decision Stump	Decision Tree
SVM (Linear Kernel)	Complexity constant	Integer	1
SVM (Polynomial Kernel)	Complexity constant	Integer	1
	Exponent	Float	2
	Lower-order terms	Boolean	False
SVM (RBF Kernel)	Complexity constant	Integer	1
	Gamma	Float	0.01

Table 2. Table of continuous models and their available parameters in Wekinator.

Model	Available Parameters	Data Type of Parameter	Default Model Parameter Values
Linear Regression	Feature selection	None/M5/Greedy	None
	Remove collinear inputs	Boolean	True
Polynomial Regression	Polynomial Exponent	Integer	2
	Feature Selection	None/M5/Greedy	None
	Remove collinear inputs	Boolean	False
Neural Network	No. of hidden layers	Integer (1 - 3)	1
	Nodes per layer	Integer	1

4. Results

The results gathered both before and after optimising ML models (via model parameters) are as follows. Section 4.1 and 4.2 show the default accuracy values of continuous and classifier models, when in their default states, in Wekinator. Section 4.3 shows the results when optimising such models.

4.1 Continuous models

Results taken when applying a MAV function to the EMG dataset used to train continuous models (without optimisation) can be seen in Table 3.

Table 3. Table showing the cross-validation accuracy (10 folds) of the default continuous models available in Wekinator, when they are trained with pre-processed and post-processed performance gesture datasets, where percentage increase between both datasets is also provided. Cross validation is measured where 0 is optimal.

Model	Raw dataset	Post-processed dataset	Percentage increase
Neural Network (NN)	**0.12**	**0.04**	**67**
Linear Regression (LR)	0.32	0.23	28
Polynomial Regression (PR)	0.32	0.23	28

4.2 Classifier models

Results taken when applying a MAV function to the pre-processed performance gesture dataset of all classifier models (without optimisation) can be seen in Table 4. These are the accuracy values of the classifier models in their default state in Wekinator.

Table 4. Table showing the cross-validation accuracy (10 folds) of the default classifier models available in Wekinator, when they are trained with pre-processed and post-processed performance gesture datasets, where percentage increase between both datasets is also provided. Cross validation is measured where 100 is optimal.

Model	Raw Dataset	Post-processed dataset	Percentage increase
k-Nearest Neighbour (k-NN)	**99.36**	99.56	0.20
Decision Tree	99.36	99.75	0.39
Decision Stump	94.78	99.54	**5.0**
AdaBoost.M1	99.36	99.64	0.28
Naive Bayes	98.86	**99.75**	0.90
SVM	97.60	99.31	1.8

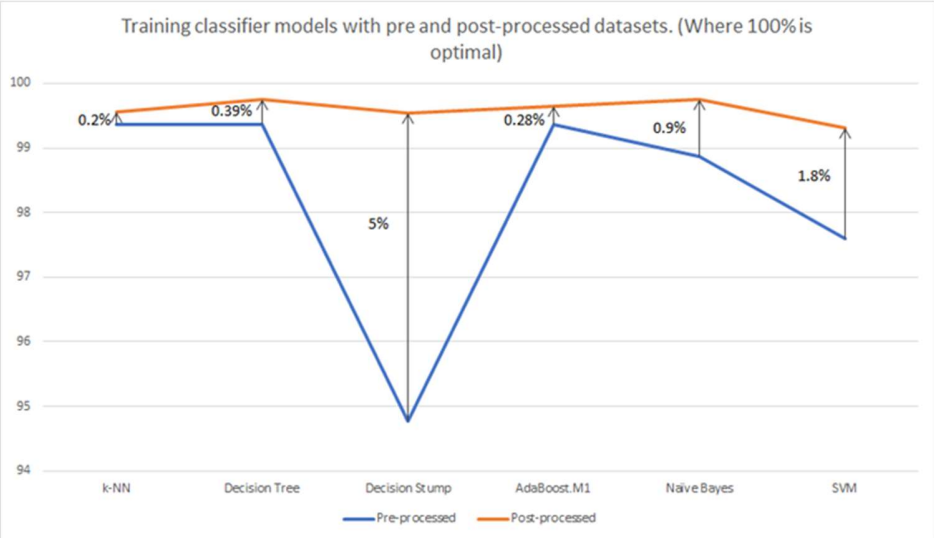

Figure 5. Graph showing the cross-validation accuracy of classifier models when trained with both performance gesture datasets, where percentage increases are also shown.

4.3 Optimising models

The results taken when optimising continuous and classifier models (trained with the post-processed performance gesture dataset) are as follows.

4.3.1 Optimising continuous models

Optimising continuous models showed that modifying parameters of the LR and PR models had no effect on model accuracy (see Table 3 for model accuracies when trained with the pre-processed and post-processed datasets), when used to classify the musical performance gesture studied in this project (see Figure 3).

NN Model. The results from optimising the NN model (when trained with the pre-processed and post-processed version of the gesture dataset) can be seen in Tables 5 and 6.

Table 5. A table showing the cross-validation accuracy (10 folds) of the NN, and percentage increase when increasing the no. of layers and increasing the no. of nodes per layer, when it is trained with the pre-processed gesture dataset. Where 0 is the optimal figure.

No. of nodes per layer	No. of layers = 1	No. of layers = 2	No. of layers = 3	Percentage increase (layers 1 to 3)	Percentage increase (from no. nodes = 1)		
					1 layer	2 layers	3 layers
1	**0.12**	0.13	0.15	25	0	0	0
5	0.14	**0.11**	**0.10**	**40**	-17	15	33
10	0.14	0.13	**0.10**	**40**	-17	0	33
100	0.36	0.46	0.54	-78	**-200**	**-250**	**-260**

Table 6. A table showing the cross-validation accuracy (10 folds) of the NN, and the percentage increase when increasing the no. of layers and increasing the no. of nodes per layer, when it is trained with the post-processed gesture dataset. Where 0 is the optimal figure.

No. of nodes per layer	No. of layers = 1	No. of layers = 2	No. of layers = 3	Percentage increase (layers 1 to 3)	Percentage increase (from no. of nodes = 1)		
					1 layer	2 layers	3 layers
1	0.04	0.03	0.03	**25**	0	0	0
5	0.03	0.03	0.03	0.0	25	0	0
10	0.03	0.03	0.03	0.0	25	0	0
100	0.43	0.33	0.42	2.3	**-980**	**-1000**	**-1300**

4.3.2 Optimising classifier models

The results from optimising all studied classifier models (when trained with the post-processed performance gesture dataset) is as follows.

Support Vector Machines (SVM). The results from optimising the linear kernel of the SVM model (when trained with the post-processed gesture dataset) can be seen in Table 7.

k-Nearest Neighbour (k-NN). The results from optimising the k-NN model (when trained with the post-processed gesture dataset) can be seen in Table 8.

Table 7. Table showing accuracies (cross-validation accuracy and training accuracy), and percentage increase of the accuracies when increasing complexity constant, of the linear SVM kernel when trained with the post-processed gesture dataset. Where 100% is optimal.

Complexity Constant	Cross-validation accuracy (10 folds)	Training Accuracy	Cross-validation accuracy increase (%) (from complexity constant = 1)	Training accuracy increase (%) (from complexity constant = 1)
1	99.31	99.31	0	0
5	99.48	99.49	0.17	0.18
50	99.64	99.65	0.33	0.34
100	99.65	99.66	0.34	0.35
1000	**99.68**	**99.68**	**0.37**	**0.37**

k-Nearest Neighbour (k-NN). The results from optimising the k-NN model (when trained with the post-processed gesture dataset) can be seen in Table 8.

Table 8. Table showing the accuracies (cross-validation accuracy and training accuracy) of the k-NN model, and the percentage increase of the accuracies when increasing k, when trained with the post-processed gesture dataset. Where 100% is optimal.

No. of neighbours (k)	Cross-validation accuracy (10 folds)	Training accuracy	Cross-validation accuracy increase (%) (from k = 1)	Training accuracy increase (%) (from k = 1)
1	99.56	**100.00**	0	0
10	**99.76**	99.78	**0.2**	-0.2
100	99.59	99.63	0.03	-0.4
1000	98.36	98.50	-1.2	-1.5

AdaBoost.M1. The results from optimising the AdaBoost.M1 model (when trained with the post-processed gesture dataset) can be seen in Tables 9 and 10. Table 9 shows the results reported when optimising the AdaBoost.M1 model with the decision stump as the base classifier. Table 10 shows the results from optimising the AdaBoost.M1 model with the decision tree as the base classifier.

Table 9. Table showing the cross-validation accuracy, training accuracy and the percentage increase of the accuracies when increasing the number of training rounds of the AdaBoost.M1 decision stump model when trained with the post-processed gesture dataset. Where 100% is optimal.

No. of training rounds	Cross-validation accuracy (10 folds)	Training accuracy	Cross-validation accuracy increase (%) from training rounds = 100	Training accuracy increase (%) from no. training rounds = 100
100	**99.61**	**99.66**	0.00	0.00
10	99.51	99.56	-0.10	-0.10
1	99.49	99.55	**-0.12**	**-0.11**

Table 10. Table showing the cross-validation accuracy, training accuracy and the percentage increase when increasing the number of training rounds of the AdaBoost.M1 decision tree model when it is trained with the post-processed gesture dataset. Where 100% is optimal.

No. of training rounds	Cross-validation accuracy (10 folds)	Training accuracy	Cross-validation accuracy increase (%) from training rounds = 100	Training accuracy increase (%) from no. training rounds = 100
100	99.64	**100**	0.00	0.00
10	99.66	**100**	0.02	0.00
1	**99.73**	99.78	**0.09**	**-0.22**

5. Discussion

The discussion regarding the results gathered from both before and after optimising ML models (via model parameters) are as follows (when trained with the pre-processed and post-processed performance gesture datasets).

5.1 Continuous models

When continuous models were trained with the pre-processed and post-processed version of the gesture dataset (see Figure 3 for the gesture studied), we can see that, in both cases, the NN reports the best cross validation and training accuracies of all three regression models (i.e. NN, LR and PR – see Table 3). This is because NNs can be used for both classification and regression problems. This means that a NN will be more accurate when it is used to classify the performance gesture.

The LR and PR models may be less accurate than the NN because they are designed to be used in regression problems. Therefore, LR and PR models are less accurate at classification, as they are not designed for this purpose; unlike NNs, which can be used to solve both types of problems (i.e. classification and regression) (Rocha, Cortez & Neves, 2007).

Continuous ML models can also be used to map an input signal. This can be seen in Rhodes et al.'s paper (2019), which includes a mapping of the output signal of several continuous models available in Wekinator (i.e. with a NN, PR and LR). They can be used to do this because they output continuous data, rather than categorical data.

5.2 Classifier models

When classifier models were trained with the pre-processed and post-processed version of the studied gesture dataset, all models demonstrated noticeable improvements in accuracy when trained with the post-processed dataset (see Table 4 and Figure 5). This is because the MAV function reduces the variance of the raw dataset, meaning that the models can better make predictions from the data to classify gestures from EMG signal data. Notably, the decision stump model showed a percentage increase of 5% in cross-validation accuracy (see Table 4).

The decision stump model demonstrated the most significant improvement (see Figure 5) because the model is less susceptible to the effects of overfitting (Sammut & Webb, 2010). This means that reducing the variance of the training dataset (by applying the MAV function) would allow the decision stump model to make better accurate predictions, without being too biased during classification. An improvement in model accuracy was expected as the MAV is cited as the optimal feature extraction method with EMG data (Arief et al., 2015).

5.3 Optimising models

The discussion regarding the results of optimising ML model parameters (when trained with the post-processed performance gesture dataset) is as follows.

5.3.1 Optimising continuous models

LR and PR models. Optimising the LR and PR models (by changing parameters such as feature selection and exponent (only in PR) - see Table 2) had no effect on model accuracy (see Table 3 for their default accuracies).

This is most likely to be due to overfitting. Changing the feature selection of the LR and PR models (i.e. none, greedy and M5) affects model behaviour, resulting in the model overfitting the data in order to produce a regression output from a classification task as "some algorithms cannot, or cannot easily be used for both problem types" (Brownlee, 2017).

NN model. The NN model, on the other hand, benefits from optimisation even though it is a regressive model (also like LR/PR models) being used for classification. This is because it is flexible, so small changes can be made to the model, to make it suitable for both "classification prediction problems" and "regression prediction problems" (Brownlee, 2018). Optimising model parameters of the NN did show to influence model accuracy. Marginally increasing the number of nodes per layer, as well as number of hidden layers, within the model demonstrated a positive impact on model accuracy, for example, having no. of nodes = 5 and no. of layers = 3 leads to cross-validation accuracy increasing by 40% (see Tables 5 and 6). However, when the number of nodes per layer within the model is significantly increased, an adverse effect on model accuracy occurs. We see this clearly when the percentage increase of cross-validation accuracy from 1 node to 100 nodes, with 1 hidden layer, is -980% (see Table 6).

The reason for this decrease in performance is overfitting. Having too high a number of nodes per layer means that the NN is unable to "properly handle new patterns that were not used in the training process" (Wilamowski, 2009, p.2). Therefore, the accuracy of the model would diminish when the number of nodes become too high because the model would significantly overfit the data, leading to a higher bias towards an output.

5.3.2 Optimising classifier models

SVM Model. When the complexity constant of the SVM is increased, model accuracy improves. We can see this when the complexity constant parameter has a value of 1000, which leads to the cross-validation accuracy increasing by 0.37% (see Table 7). This is because, as the complexity constant increases for the SVM model, the algorithm starts to overfit the data. This means that the model can better 'fit' the patterns of the data, leading to more complex hyperplanes and better classification (Yadav, 2018), resulting in an improved accuracy. However, having too high a value of the complexity constant can lead to a lower accuracy due to overfitting (Webb, 2010). In this sense, if the dataset has too low of a variance, the model may become biased towards a certain output, resulting in a lower accuracy. However, applying the MAV function to the dataset doesn't seem to have reduced the variance too drastically in order to impact the model's accuracy; instead, the accuracy increases when the MAV is applied, having a noticeable percentage increase of 1.8% (see Table 4).

When the model is trained with the raw dataset (i.e. without MAV function applied), the accuracy of the model increases when the complexity constant increases. This is because the SVM model can overfit the data and make better predictions when classifying the performance gesture used in this investigation (see Section 3.2 for this discussed gesture).

k-NN Model. When the number of neighbours (k) is increased within the k-NN model, accuracy is adversely affected, for example, when the k parameter has a value of 1000, cross-

validation accuracy had a percentage increase of –1.2%. (see Table 8). The reason for this pattern is that the model classifies input data by comparing it with 'neighbours' (i.e. the closest classified data points). As k increases, so do the number of neighbours that are compared with the unclassified data point (i.e input data). However, the distances between the unclassified EMG signals - and the classified EMG signals - are no longer prioritised; instead the number of each classified data point is considered (unless weights are assigned to neighbors based on distance (Cunningham & Delany, 2020)). This results in a reduced model accuracy as the EMG signal will be classified as the most commonly occurring classification, rather than the data classification that is closest to the unclassified EMG data sample (Cunningham & Delany, 2020).

The optimised form of the k-NN model is when k = 10, where the cross-validation accuracy is reported at 99.76% (see Table 8). This is when the unclassified EMG signal is compared to ten neighbours, meaning it will be classified more accurately, as similar data points are usually closer to each other. Therefore, it would be needless to compare the unclassified data point with a larger range of points (Cunningham & Delany, 2020).

AdaBoost.M1 Model. There are two base classifiers for the AdaBoost.M1 model: decision stump and decision tree. Regarding the decision stump, when the number of training rounds decreases, model accuracy is negatively impacted, for example, when the no. of training rounds parameter has a value of 1, cross-validation accuracy has a percentage increase of –0.12% (see Table 9). This is because a decision stump base classifier can be used as a 'weak learner' in boosting algorithms, such as AdaBoost.M1 (Sammut & Webb, 2011). When the decision stump base classifier is trained with the AdaBoost.M1 model, a new model type is trained every round. After each training round, the boosting algorithm identifies the training samples that have not been learned well and adjusts "the sampling distribution of the training data" (Cunningham, Cord & Delany, 2008, p.45). This is done in order to focus on the errors made by the model in previous rounds. This means, here, that the decision stump model will become more accurate when it is trained with this boosting algorithm, and with a higher number of training rounds.

Regarding the decision tree base classifier, it is evident that, as the number of training rounds decreases, the cross-validation accuracy score for the AdaBoost.M1 model improves. This is due to model overfitting. When the number of training rounds increase, the AdaBoost.M1 model will adjust the training data used to account for errors made by the decision tree in previous rounds (Cunningham et al., 2008). However, this could lead to the model overfitting unseen data (i.e. when the no. of training rounds is high), leading to a lower cross-validation accuracy; using unseen data to test the accuracy of the model. However, the opposite effect is true for the training accuracy (see Table 10). This is because the decision tree base classifier was used to train the AdaBoost.M1 model; a boosting algorithm which can be used to increase the accuracy of a model by "reduc[ing] the bias component of error as well as the variance component" (Cunningham et al., 2008, p.45). Furthermore, we can see that a reduction in variance improves the accuracy of ML models (see Tables 3 and 4), as the ML models were trained with a dataset that had the MAV function applied. As the number of training rounds are increased, we can see that the training accuracy of the decision tree base classifier improves, as the boosting algorithm (AdaBoost.M1 model) will have adjusted the distribution of the training samples to improve the errors made in prior training rounds (Cunningham et al., 2008).

However, when the decision tree base classifier was trained with the pre-processed performance gesture dataset, a different pattern is shown. In this case, both the cross-validation

and training accuracies of the model remained at a value of 99.36%, regardless of any changes to the number of training rounds (see Table 4). This is because of the high variance in the dataset, since the MAV function was not applied. Thus, the high variance of the training dataset could have influenced the decision tree's ability to classify, as it would attempt to classify the data through using a binary classification method (Liu & Özsu, 2009). This would explain the lack of variability in the decision tree's accuracy when the number of training rounds is decreased. Therefore, the high variability within the dataset would mean that the model cannot improve.

6. ML model optimisation and interactive music practice

ML model efficacy affects interactive music composition practice because we observe, in this study, that model accuracy is affected by parameter optimisation and dataset choice. This is evident when we see that the post-processed performance gesture dataset (i.e. using the MAV as a feature extractor), significantly impacted model accuracy (see Sections 5.1 and 5.2 for a breakdown on the affect). Applying parameter optimisation to ML models, as well as training them with the post-processed performance gesture dataset, leads to a greater impact on model accuracy. Therefore, model accuracy will consequently affect the efficacy of performance gesture recognition. In this sense, music performers using ML software (i.e. Wekinator) and GIs, to embody the composition process, can improve the efficacy of their interactive music practice - and thus, their art. For example, studies looking at applying ML to music practice, such as one conducted by Dalmazzo & Ramirez (2019), investigated how using a specific ML model (Hierarchical Hidden Markov Model) when recognising musical gestures (working with a professional violinist to record seven violin bowing techniques) demonstrated that ML can be applied successfully to music practice and performance, outside of the context of research.

Furthermore, using the output from this study, we can see that optimised classifier ML models can be used to categorise musical performance gestures due to the nature of the model output, which is a discrete, arbitrary integer; representing the prediction through classification. Optimised classifier model outputs can be used to trigger a set of predefined sounds for the musician to use. Even though the musical material must be predefined, this type of model output is useful because musicians can creatively enhance their performance through the accuracy of the gesture recognition itself, when applied. Therefore, optimising classifier models will improve classifier ML model accuracies and enhance the artform. Furthermore, optimising classifier models will also allow for a better fluid music performance, through the use of GIs, as performance gestures are less likely to be misclassified; in this sense, a performer can be expressive without being hindered by technical issues. Although, however useful, it is important to note that classifier models do not allow for the same depth of musical embodiment and flexibility as continuous ML models do.

Continuous ML models output predictions for EMG data across time. In this sense, they are suitable for the real-time DSP of audio signals, rather than the classification of performance gestures. In this study, we find that that parameter optimisation done on continuous models in this study does not have an affect on the model accuracy when used to classify the studied performance gesture – (see Figure 3 for an illustration of the gesture). However, the NN was shown to improve model accuracy, through optimisation, because it can also operate as a classifier model, as well as a continuous/regression model – see Section 5.1 & 5.3.2). This means that NNs would allow musicians the flexibility to use the model in different ways, as classifier

models can enhance performances by providing musicians with the ability to trigger predefined actions and continuous models can enhance performances by providing musicians the ability to apply real time DSP. Therefore, because of the increased mobility when handling data, continuous models allow for the performer to better embody music, than if a classifier model were used. This is because the performer is not restricted by pre-determined (trained before a performance) gestures. Instead, live EMG data gathered from a GI can undergo a process of live sonification (i.e. representing data in an auditory manner (Barrass & Kramer, 1999)) if a continuous model is used. However, it should be noted that type of continuous model used affects model behaviour and, therefore, the mapped signal output produced from the continuous models. We can see in Rhodes et al.'s (2019) paper that the NN, PR, LR continuous models have different mappings of an input signal. However, the effects of optimising continuous models on the mapped signal output has not been investigated in this project. This could mean that model parameter optimisation, which has shown to have an effect on model accuracy (and therefore, model output), would also affect the mapping of the output produced from a continuous model. Therefore, this means that the optimisation process will affect the sonification and signal mapping process. This would give the musician greater creative freedom, as they are able to optimise continuous models and therefore produce disparate sonic behaviours.

Furthermore, it is clear that model choice has an impact of music practice and, as a result, the working accuracy of each model type will affect music practice accordingly. Therefore, this study has shown that optimising each model type can improve the quality of interactive music performance, which frequently uses GIs, ML and biometric data to stimulate it. Classifier models, because of the data type of their model outputs (integer based), can be used to retain the aesthetic integrity and most stable elements of a musical composition (such as rhythm). This is because the model output is non-continuous and, as a result, less mobile. However, continuous models are most appropriate when sonifying data continuously, in real time (via DSP), because of the nature of the ML model output; therefore, applying live DSP to an audio signal would result in a lack of metricity. Therefore, it is evident that both model outputs can be used to each disparate musical aesthetics.

7. Conclusion

In conclusion, we have observed that applying parameter optimisation to ML models affects model accuracy when used to classify performance gestures within the context of interactive music composition practice (see Sections 5 & 6). ML model choice (when being used to classify performance gestures from EMG datasets) depends on the musical context it will be used in (see Section 6). Parameter optimisation can then be used to improve model accuracy, making the model more suitable for use in live musical compositions, where misclassifications of performance gestures would occur less often as model accuracy is optimal.

Additionally, we have observed that using different ML training datasets affects the impact of parameter optimisation and model prediction accuracy. This can be evidenced in Tables 4 & 10, where the AdaBoost.M1 decision tree (base classifier) model only improved after model optimisation when trained with the post-processed performance gesture dataset. Therefore, using a post-processed dataset (i.e. applying MAV feature extraction method) not only improves default model accuracy but also improves the efficacy of parameter optimisation for some models (see Sections 4 & 5).

Further work in this topic could investigate the effects optimising continuous models will have on a continuous model's mapping output, as different mappings of the data will result in a different audio representation of the mapping when it is sonified (see Section 6). In order to optimise model parameters within Wekinator, we used a manual process of typing in/selecting model parameters, resulting in an inefficient process in finding the optimal state of an ML model, when used to classify performance gestures. Therefore, future research could find an automatic process method when optimising parameters, in turn making the process of parameter optimisation of ML models efficient for musicians.

8. Further applications of ML model optimisation

Whilst this research project looks at using GIs to observe how ML algorithms can be optimised within music, the findings made in this report could have an impact on HCI applications (using ML models with GIs) and accommodate for wider commercial opportunities.

In particular, current VR applications using biometric data, GIs and ML (Kulshreshth, Anand & Lakanpal, 2019) can benefit from the use of optimised models to better classify gestures. VR applications using GIs (i.e. the Myo) and EMG data are unaffected by variables which some commercial peripherals are, when using cameras and ML, regarding gesture recognition (Ma, Varley, Shark & Richards, 2010). For example, commercial VR hardware, such as the Oculus Quest, uses cameras within a VR headset to track user gestures via ML (Oculus, 2019). However, environmental factors such as light will impact the accuracy of using a camera to identify user gestures. Therefore, GIs measuring EMG behaviour (i.e. the Myo) can be considered a solution when improving the efficacy of gesture recognition with VR peripherals and ML. For this reason, using GIs and EMG data in VR (Smys, 2019) is a novel field of research and application. However, this has not been adopted in mainstream VR applications yet.

Furthermore, using GIs (such as the Myo armband) to gather EMG data, and processing it via ML, can also be used within medical settings. For example, GIs like the Myo have been used by medical doctors and patients in physiotherapy, where EMG signals are analysed by doctors to "detect medical abnormalities and hand movements" (Sathiyanarayanan & Rajan, 2016, p.1). This is useful because it shows the use of GIs and EMG data can be used to better assist doctors in early diagnosis in physiotherapy, however, there was no use of ML in this paper. Our research could then be applied to this use of GIs and EMG, as ML can be used to make EMG data analysis efficient for diagnosis in physiotherapy. Therefore, this research project can be used to find an efficient ML model, via optimisation, that can be used to analyse EMG data signals in a medical context (Sathiyanarayanan & Rajan, 2016). Finding a highly accurate ML model, through optimisation, would also strengthen the benefits of using ML for diagnosis, as patients and doctors may feel wary of using these methods. (Longoni, Bonezzi & Morewedge, 2019).

Acknowledgements:

Thank you to the Nuffield Foundation for providing the opportunity to work on this research placement. I would also like to thank Chris Rhodes and Richard Allmendinger for their support throughout this project and for providing the raw EMG dataset of the performance gesture studied in this paper, gathered in their 2019 study.

Author's Contribution:

S.B. wrote the core body of the paper and analysed all data. C.R. created the project idea and scope of the topic, provided technical support, and regular manuscript edits. R.A. provided project insights.

References

- Amores, J., Richer, R., Zhao, N., Maes, P., & Eskofier, B. M. (2018, March). Promoting relaxation using virtual reality, olfactory interfaces and wearable EEG. In 2018 IEEE 15th international conference on wearable and implantable body sensor networks (BSN) (pp. 98-101). IEEE.
- rief, Z., Sulistijono, I. A., & Ardiansyah, R. A. (2015, September). Comparison of five time series EMG features extractions using Myo Armband. In 2015 International Electronics Symposium (IES) (pp. 11-14). IEEE.
- Barrass, S., & Kramer, G. (1999). Using sonification. Multimedia systems, 7(1), 23-31.
- Bongers, B. (1998). An interview with Sensorband. Computer Music Journal, 22(1), 13-24.
- Brownlee, J. (23 July 2018). When to Use MLP, CNN, and RNN Neural Networks. Retrieved from https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/
- Brownlee, J. (11 December 2017). Difference Between Classification and Regression in Machine Learning. Retrieved from https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/
- Caramiaux, B., & Tanaka, A. (2013). Machine Learning of Musical Gestures: Principles and Review. In Proceedings of the International Conference on New Interfaces for Musical Expression (NIME) (pp. 513-518). Graduate School of Culture Technology, KAIST.
- Cunningham, P., Cord, M., & Delany, S. J. (2008). Supervised learning. In Machine learning techniques for multimedia (pp. 21-49). Springer, Berlin, Heidelberg.
- Cunningham, P., & Delany, S. J. (2020). k-Nearest Neighbour Classifiers--. arXiv preprint arXiv:2004.04523.
- Dalmazzo, D., & Ramírez, R. (2019). Bowing gestures classification in violin performance: a machine learning approach. Frontiers in Psychology, 10, 344.
- Di Donato, B., Dooley, J., Hockman, J., & Hall, S. (2017). Myospat: A hand-gesture controlled system for sound and light projections manipulation.
- Winkler, T. (2001). Composing interactive music: techniques and ideas using Max. MIT press.
- Fiebrink, R., & Cook, P. R. (2010, August). The Wekinator: a system for real-time, interactive machine learning in music. In Proceedings of The Eleventh International Society for Music Information Retrieval Conference (ISMIR 2010) (Utrecht).
- Hamadicharef, B., Xu, M. and Aditya, S., "Brain-Computer Interface (BCI) Based Musical Composition," *2010 International Conference on Cyberworlds*, Singapore, 2010, pp. 282-286, doi: 10.1109/CW.2010.32.
- Holmes, G., Donkin, A., & Witten, I. H. (1994, November). Weka: A machine learning workbench. In Proceedings of ANZIIS'94-Australian New Zealnd Intelligent Information Systems Conference (pp. 357-361). IEEE.
- Kulshreshth, A., Anand, A., & Lakanpal, A. (2019, October). Neuralink-An Elon Musk Start-up Achieve symbiosis with Artificial Intelligence. In *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)* (pp. 105-109). IEEE.
- Liu, L., & Özsu, M. T. (Eds.). (2009). Encyclopedia of database systems (Vol. 6). New York, NY, USA: Springer.
- Longoni, C., Bonezzi, A., & Morewedge, C. K. (2019). Resistance to medical artificial intelligence. Journal of Consumer Research, 46(4), 629-650.
- Ma, S., Varley, M., Shark, L. K., & Richards, J. (2010, July). EMG biofeedback based VR system for hand rotation and grasping rehabilitation. In *2010 14th International Conference Information Visualisation* (pp. 479-484). IEEE.
- Marquez-Borbon, A., & Stapleton, P. (2015, May). Fourteen years of nime: the value and meaning of 'community' in interactive music research. In NIME (pp. 307-312).

- North. (Date unknown) What is the Wireless Range of the Myo? https://support.getmyo.com/hc/en-us/articles/202668603-What-is-the-wireless-range-of-Myo-
- Nymoen, K., Haugen, M. R., & Jensenius, A. R. (2015, May). MuMYO-Evaluating and Exploring the MYO Armband for Musical Interaction. In Proceedings of the international conference on New Interfaces for Musical Expression (pp. 215-218).
- Oculus. (25 September 2019) Oculus Quest @ OC6: Introducing Hand Tracking, Oculus Link, Passthrough+ on Quest, and More. Retrieved from https://www.oculus.com/blog/oculus-quest-at-oc6-introducing-hand-tracking-oculus-link-passthrough-on-quest-and-more/?locale=en_GB
- Rocha, M., Cortez, P., & Neves, J. (2007). Evolution of neural networks for classification and regression. Neurocomputing, 70(16-18), 2809-2816.
- Rhodes, C., Allmendinger, R., & Climent, R. (2019, November). New Interfaces for Classifying Performance Gestures in Music. In International Conference on Intelligent Data Engineering and Automated Learning (pp. 31-42). Springer, Cham.
- Ruan, X., Wang, Q., Liu, R., Fan, J., & Liu, J. (2018, April). Air-Ukulele: Finger Movement Detection Based on sEMG for Ukulele playing. In Proceedings of the Sixth International Symposium of Chinese CHI (pp. 132-135)
- Sammut, C., & Webb, G. I. (Eds.). (2011). Encyclopedia of machine learning. Springer Science & Business Media.
- Sathiyanarayanan, M., & Rajan, S. (2016, January). MYO Armband for physiotherapy healthcare: A case study using gesture recognition application. In 2016 8th International Conference on Communication Systems and Networks (COMSNETS) (pp. 1-6). IEEE.
- Smys, S. (2019). Virtual reality gaming technology for mental stimulation and therapy. Journal of Information Technology, 1(01), 19-26.
- Webb, G. I. (2010). Overfitting.
- Wilamowski, B. M. (2009). Neural network architectures and learning algorithms. IEEE Industrial Electronics Magazine, 3(4), 56-63.
- Winkler, T. (2001). Composing interactive music: techniques and ideas using Max. MIT press.
- Yadav, A. Support Vector Machines (SVM). (2018) https://towardsdatascience.com/support-vector-machines-svm-c9ef22815589
- Zadeh, A. S., Calitz, A. P., & Greyling, J. H. (2018, September). Evaluating a biosensor-based interface to recognize hand-finger gestures using a Myo armband. In Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists (pp. 229-238).