A Machine Learning Toolbox For Musician Computer Interaction

Nicholas Gillian
Sonic Arts Research Centre
Queen's University Belfast
United Kingdom
ngillian01@qub.ac.uk

R. Benjamin Knapp
Sonic Arts Research Centre
Queen's University Belfast
United Kingdom
b.knapp@qub.ac.uk

Sile O'Modhrain
Sonic Arts Research Centre
Queen's University Belfast
United Kingdom
sile@qub.ac.uk

ABSTRACT

This paper presents the SARC EyesWeb Catalog, (**SEC**), a machine learning toolbox that has been specifically developed for musician-computer interaction. The SEC features a large number of machine learning algorithms that can be used in real-time to recognise static postures, perform regression and classify multivariate temporal gestures. The algorithms within the toolbox have been designed to work with any N-dimensional signal and can be quickly trained with a small number of training examples. We also provide the motivation for the algorithms used for the recognition of musical gestures to achieve a low intra-personal generalisation error, as opposed to the inter-personal generalisation error that is more common in other areas of human-computer interaction.

Keywords

Machine learning, gesture recognition, musician-computer interaction, SEC

1. INTRODUCTION

It has long been the goal of many composers, performers and researchers alike to be able to use their own body movements to trigger, control and manipulate electronic sounds in real-time, live on stage. This goal is slowly being made possible by the ever decreasing cost of sensor devices, such as the Wii[1] or SHAKE[2], combined with the increasing number of machine learning algorithms in programs like Max/MSP[3], Pure Data[4], Chuck[5], EyesWeb[6] and the Wekinator[12]. As authors such as Fiebrink et. al.[12] have shown, it is now possible for performers to train a machine learning algorithm in real-time, live on stage and have the performer's movements (being sensed from anything such as a common gamepad to body worn accelerometers or EMG) be mapped directly to, for example, the synthesis parameters of a FM synthesiser. The machine learning algorithms featured in the programs listed above are generally excellent for solving two common problems; namely the discrete classification of a static posture as one of K possible postures and the mapping of an input signal to one or more continuous output variables, also known as regression. However, many of the existing machine learning toolboxes are still unable to classify patterns that occur in a multidimensional space and change over a variable time period, otherwise known in the machine learning literature as multivariate temporal signals. Further, it was found that the few toolboxes that do include multivariate temporal recognition algorithms either only work offline, take an extensive amount of time to train or are limited to accept a specific form of sensor input, such as the 2D data from a mouse or 3D data from an expensive motion capture device.

This has therefore provided the motivation for the design and development of a novel machine learning toolbox that can recognise static postures, perform regression and classify multivariate temporal gestures. The toolbox, called the SARC EyesWeb Catalog (**SEC**), has been specifically designed to work with any-type of N-dimensional signal and operates as a middleware application; thus enabling a performer to easily integrate it into their own existing composition and performance environment. The SEC has been designed so it is suitable for both performers with basic technology skills and no knowledge of machine learning right through to domain experts who want to create their own custom-built recognition systems. It is also suitable for researchers who wish to integrate and test a new form of feature extraction algorithm with the existing machine learning algorithms in the SEC. One of the main benefits that the SEC offers a performer is that it enables them to use the raw data or features from any sensor to quickly train a machine learning algorithm with the gestures the performer wants to use. After training the machine learning algorithm the performer can then use it to recognise their gestures in real-time, even in a continuous stream of data that also contains non-gestural data. In this paper we present the SEC and describe how its machine learning algorithms have been specifically adapted for the recognition of musical gestures.

2. RELATED WORK

Machine learning algorithms have been successfully applied to a number of tasks throughout many areas of musician-computer interaction (**MCI**). Lee et al. [17] and Fels et al. [9] were some of the first to apply the broad history of machine learning research on Artificial Neural Networks (**ANN**) to the field of MCI. Lee used an ANN to map the input from a radio baton, sensor glove or a MIDI keyboard to audio output and Fels mapped the input from a Cyberglove, 3-D tracker and a footpedal to a speech synthesiser. Fels work was later extended by Pritchard [22] who used several ANN to allow the user to synthesise audio, speech

[1]http://uk.wii.com/
[2]http://www.dcs.gla.ac.uk/research/shake/
[3]http://cycling74.com/products/maxmspjitter/
[4]http://puredata.info/
[5]http://chuck.cs.princeton.edu/
[6]http://www.infomus.org/EywMain.html

and song in real-time. Modler [19] also applied an ANN to map the sensor data captured by a sensor glove to continuously control the parameters of a synthesis engine running in SuperCollider. Along with applying the ANN to continually map the glove data to synthesis parameters, Modler also used the ANN to recognise patterns in the glove data, such as the classification of certain hand postures like thumbs up or an extended index finger. The recognition of a specific symbolic hand gesture could then be used to trigger a sound, with the energy of the finger movement being mapped to control the damping factor of a plate model. Cont et al. [6] created a number of ANN blocks for the Graphical User Interface (**GUI**) program Pure Data that enabled a performer to quickly train and recognise dynamic temporal gestures sensed by two perpendicular accelerometers. The network was trained using six constant speed circle gestures and was able to satisfactorily recognise a large variety of circles performed at different speeds and sizes.

Merrill et al. [18] built the FlexiGesture, a two handed device that features a number of sensors including 3-degree-of-freedom (**DOF**) accelerometers, 3-DOF gyroscopes, 4-DOF squeezing, 2-DOF bending and 1-DOF twisting. The user could train the system to recognise a temporal gesture by pressing a 'trigger' button which starts the data recording process, releasing the button when they have completed the gesture. The system then asks the user to continually re-perform the gesture as it trains a template model for that gesture. Dynamic Time Warping was used as the recognition algorithm and tests showed that the system was able to classify novel gestures into one of ten classes with up to 98% accuracy.

Fiebrink et al. [12] created a real-time, on-the-fly machine learning-based system called the Wekinator that can be trained by the user in a number of seconds. The Wekinator affords the user the ability to quickly experiment with input/output mappings and even form judgements on the quality of the mapping by training and running it in real-time and observing the sonic results. The system was used for a live performance in which six performers started the training/mapping process from scratch, live on stage, and each performer gradually converged on the mapping setup they wanted as the piece progressed. Fiebrink et al. [11] extended this work by adding an additional 'play-along' paradigm to the Wekinator in which the user listened to a specific piece of music whilst mimicking the gesture they would have liked to have performed to make that sound. The system was then trained on this gesture-sound relationship and the user was able to create a sound or effect by performing the corresponding gesture.

Bevilacqua et al. [1] [3] have developed a real-time continuous gesture recognition system for Max/MSP in which a Hidden Markov Model can continuously output, not only the likelihood of the user performing a given gesture at the current time, but also, where in that gesture the user might be. One of the main benefits of this system is that it has been specifically designed to be trained with the minimum possible training examples (in some cases even one example can be sufficient). Bevilacqua et al. also [2] developed the MnM toolbox for Max/MSP which is dedicated to mapping between gesture and sound, applying algorithms such as Principal Component Analysis to reduce the dimensionality of the data, thus simplifying the mapping procedure.

A number of researchers have focused on capturing the natural gestures performed on acoustic instruments such as Overholt et. al. [21] who added a number of algorithms from the OpenCV library to their Multimodal Music Stand System (MMSS) to recognise the gestures of a flautist and use these to control a Max/MSP patch. Morales-Mazanares et al. [20] also tried to recognise the gestures of a flautist, using a probabilistic model to estimate what the attacks or angular displacement of the instrument could inferrer about the player's gestures. The accurate classification of violin bowing gestures has also received attention from [23] [26] [10]. Finally the recognition of a conductors gestures has received a large body or research [24] [16] [7].

These examples have illustrated how machine learning algorithms have been successfully applied to solve both classification and regression problems throughout many areas of MCI. A large majority of this work, however, has been designed for custom-built hardware devices [9] [18] or is constrained to recognising gestures from a specific sensor, such as the data from a 2D mouse [3], or is designed for a particular instrument, such as a flute [21]. This provided the motivation for us to develop a new machine learning toolbox that is specifically aimed for the real-time recognition of musical gestures. The SEC contributes to this existing work because it has not be constrained to work with just one sensor device or audio environment, can be used to classify both static postures, temporal gestures and perform regression and most importantly can be quickly trained by a musician with a small number of training examples.

3. THE SEC

The SEC[7] has been fully integrated as a third party library within a free program called EyesWeb. EyesWeb is an open software platform that was established to support the development of real-time multimodal distributed interactive applications and already features a large number of algorithms for processing both video and audio signals [5]. EyesWeb is a GUI orientated program that runs in Windows[8] which features a *patch window* onto which the user can drag a number of *blocks* that represent a specific algorithm or function. A block will commonly feature a number of input, output and parameter pins, with one block's output pin being connected to another block's input pin to create a signal flow between the two respective blocks. Using a small number of blocks in EyesWeb, for example, a performer could build a patch to capture real-time data from a sensor unit, filter the data and plot the results without having to write a single line of code. EyesWeb also enables any performer with more technical skills to develop their own blocks, which may be required to perform a specific type of feature extraction or to interface with a custom-built piece of hardware. All the blocks in EyesWeb are written in C++, giving the developer the ability to write fast, efficient code which is a necessity for real-time machine learning due to the large number of calculations required. EyesWeb therefore provides an excellent environment for both technical and non-technical users as complex signal processing operations can be easily constructed by connecting a number of blocks together or, alternatively, a custom block can be developed to perform one specific task.

3.1 The SEC Blocks

The SEC contains over 80 blocks (almost twice the number of blocks since its first public release [13]), all of which have been specifically designed for the real-time recognition of musical gestures. Along with featuring a number of rudimentary blocks for saving/loading data, converting from one data type to another etc., the SEC also contains blocks for signal processing, performing mathematical operations, and interfacing directly with hardware sensor units such as the Wii, the SHAKE and Infusion System's Wi-

[7]`http://www.somasa.qub.ac.uk/~ngillian/SEC.html`
[8]EyesWeb is currently being ported to Linux

Table 1: SEC Gesture Recognition Algorithms

Algorithm Name	Suitable Application	Learning Type
Adaptive Naïve Bayes Classifier	Classification	Supervised
Artificial Neural Networks	Regression	Supervised
Hidden Markov Models	Classification	Supervised
N-Dimensional Dynamic Time Warping	Classification	Supervised
Fuzzy C-Means Clustering	Classification	Unsupervised
K-Means Clustering	Classification	Unsupervised
K-Nearest Neighbor Classification	Classification	Supervised
Support Vector Machines	Classification	Supervised

microDig[9]. The SEC contains a large number of machine learning algorithms that can be used to classify static and temporal gestures as well as perform regression, a list of the main algorithms can be found in table 1. The SEC also contains a number of pre-processing or feature extraction algorithms that can be applied to reduce the computational load and complexity of a recognition problem along with a variety of post-processing algorithms that can be used to improve the overall system's classification performance. Each machine learning, feature extraction and post-processing algorithm in the SEC has been encapsulated as a single block, enabling the user to quickly create their own recognition system by dragging the algorithms they wish to use onto the EyesWeb patch window and connecting them together. This facilitates a user with even basic technical skills to apply a wide range of extremely powerful machine learning algorithms, such as Support Vector Machines (**SVM**) or Hidden Markov Models (**HMM**), to recognise their musical gestures without having to write a single line of code. One of the most important features of the algorithms within the SEC is that they have all been designed to work with any N-dimensional input signal. This means that the recognition algorithms are not constrained to just work with the two-dimensional data from a mouse for example, but can work with any N-dimensional continuous stream of data. This is a key advantage for performers, particularly those that create their own custom built sensors, interfaces or instruments, as the output from any sensor(s), or features derived from this sensor data, can easily be used as input to any of the SEC blocks.

3.2 Using the SEC for MCI

The machine learning algorithms within the SEC enable any performer to use one or more musical gestures to control and manipulate the performer's composition or improvisation software in real-time. For example, a musician could use a classification algorithm such as N-Dimensional Dynamic Time Warping (ND-DTW) [15] to classify a specific conducting gesture and use the recognition of this movement to trigger the computer to start manipulating the live audio recording of the musician the gesture was directed towards. At the same time, the performer could use a regression algorithm like an ANN to continuously map the velocity at which the performer made the conducting gesture to control the degree of the warping effect on the live audio recording.

Rather than targeting the SEC for just one specific piece of audio software, it has been designed to function as middleware enabling the user to pipe their sensor data into the SEC via a number of standard communication protocols, such as Open Sound Control (**OSC**) [25]. After recognition the classification results can be piped out of the recognition system to control any piece of audio or visualization software that use the same communication protocols. A middleware design architecture also enables the SEC to run on an independent machine from that which is running the audio software; which is beneficially for CPU intensive recognition algorithms. A performer can therefore write their own software to capture and parse the real-time data from whatever sensor(s) they might be using and pipe this data into EyesWeb via OSC. Alternatively, a performer could directly implement the sensor interface as an additional EyesWeb block.

3.2.1 Creating a Robust Recognition System

Machine learning algorithms rarely exist in a vacuum [8]. A robust recognition system commonly requires an appropriate pre-processing or feature extraction stage prior to any classification by a trained machine learning algorithm, with the predicted classification label being post-processed prior to being acted upon. A user may therefore want to experiment with various feature extraction algorithms or post-processing functions as well as testing which machine learning algorithm works best for the recognition of their gestures. It is for this reason that each feature extraction algorithm or machine learning algorithm has been encapsulated as a single EyesWeb block as this enables the user to connect the blocks together to create the recognition system the user thinks maybe most appropriate for solving their recognition problem. One of the major advantages of using a patch-based GUI program such as EyesWeb is that multiple recognition algorithms can be used in parallel, with the output of one classifier providing contextual information for another classification chain. For example, the predicted event of one classifier could be used to permit/deny the output of a second classifier from being acted upon.

3.2.2 Training a Machine Learning Algorithm

Prior to using any machine learning algorithm it must first be trained. This can be achieved by using a number of examples, called a *training set*, to tune the parameters of the algorithm's adaptive model or function. The training set could contain, for example, a number of recordings of each of the G gestures the performer wants the algorithm to recognise. Each training example may also be hand-labelled by the user in which case the problem is known as *supervised learning*. By adopting a machine learning approach, a musician can teach a computer to recognise their musical gestures by performing a number of repetitions of each gesture and use this data to train a machine learning algorithm. If the appropriate feature(s) are used to represent the gesture and a suitable algorithm is trained then the algorithm should be able to classify a new input vector as one of the G gestures it was trained with; even if the new input vector was not contained in the original training set. The ability to categorize correctly new examples that differ from those used for training is known as *generalisation*. In

[9]http://infusionsystems.com

practical applications, the variability of the input vectors will be such that the training data can comprise only a tiny fraction of all possible input vectors, and so generalisation is a central goal in pattern recognition [4].

The SEC features a number of useful tools to facilitate a user to efficiently create a training set and then quickly train a machine learning algorithm. Each algorithm, for example, will commonly have a dedicated block for recording training data, a second block for training the algorithm and a third block for the real-time classification of any new data using the trained model. This three block design enables the user to create a 'training patch' for recording and training the algorithm and a separate 'prediction patch' for real-time classification that may also contain other trained machine learning algorithms, post-processing algorithms and network connections to communicate with other audio/visual software.

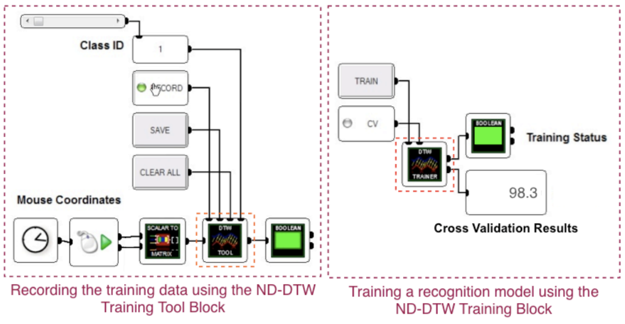

Figure 1: An example training patch for the ND-DTW algorithm

The training patch, illustrated in Figure 1, will have the identical sensor input and feature extraction methods as the prediction patch, see Figure 2, but can also contain a number of helpful features that assist the user in collecting and labeling the training data. This could consist of timer functions, for example, that enable the user to press a key to prepare the system to record a two-handed gesture. After a predetermined delay the user can then start to perform the gesture while the system records the training data, automatically labeling each training sample with the ID value of that gesture. After a further predetermined delay the system stops recording the gesture and the user can either record another example of the same gesture or move onto the next gesture in their vocabulary. When the user has created a number of training examples for each gesture they can save the training data to a file and then use this to train the machine learning algorithm. Each algorithm will then save its trained model to a file to enable it to be loaded by the real-time classification block.

The user can select if they wish to train the algorithm using an automatic validation method, such as K-fold cross-validation, to estimate the generalisation ability of the trained model or if they want to devote all of the available data to training the model and instead test the algorithm 'online' using the prediction patch. Either way, if a poor model has been created the user can quickly reload the original training data in the training patch and modify some of the parameters of the machine learning algorithm or even change the feature extraction method and quickly retrain a new model with the updated settings. Alternatively, the performer could use the one training set to train and validate several algorithms each with different settings all at the same time to determine the best features/algorithm/settings to use. The performer then simply needs to load the best model into the predication patch. These examples illustrate the advantages of using three separate blocks to create a training set, actually train a model and finally perform real-time prediction on new data using the trained model.

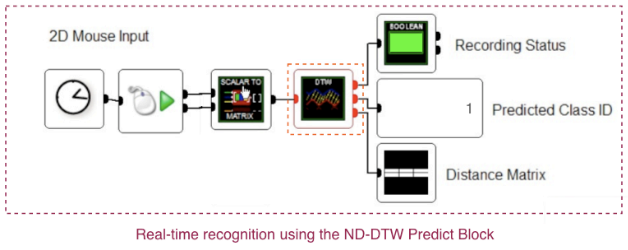

Figure 2: An example prediction patch for the ND-DTW algorithm

4. DESIGNING ALGORITHMS FOR MCI

The design, development and evaluation strategies applied to the algorithms within the SEC have required a fundamental paradigm shift from the common strategies employed throughout other areas of machine learning. In many areas of machine learning and gesture recognition the primary goal of any algorithm is to achieve a low inter-personal generalisation error, which is the algorithms ability to correctly classify the gestures of a new participant that was not included in the training set. This is the case, for example, with a computer game that recognises the gestures of a player and uses these to control a character in the game. In order for the system to robustly recognise thousands of different players across the world who may want to play the game, the recognition algorithm may need to be trained with a very large number of training examples collected from perhaps hundreds of different participants. The machine learning algorithm can then be trained with this extensive data set, perhaps overnight, after which it could be tested with another large test set that contains new data that was not used to train the algorithm to validate the classification abilities of the model.

4.1 The Intra-personal Generalisation Goal

Applying this approach to training and testing a machine learning algorithm may not be suitable, however, for the recognition of musical gestures - such as in a NIME interaction scenario. This is because each performer may want to define their own unique gestural vocabulary, i.e. the relationship between a gesture and its corresponding action. A performer may also want to capture the gestures using their own specific sensor device and use the recognition of a gesture to control a custom-made piece of audio software. It is therefore difficult to create the pre-trained recognition systems that are common throughout many areas of human-computer interaction (HCI). Musician-computer interaction alternatively requires a system that has a flexible input/output configuration and that can be trained by the performer using gestures from the their own vocabulary.

A user-configurable recognition system for MCI would not therefore require the inter-personal generalisation abilities found in other areas of HCI; instead it would simply need to provide a good intra-personal generalisation for the one performer that initially trained the system. If another performer wants to use their own input device or gestural vocabulary to control the same audio software, then they simply have to retrain the machine learning system with their own gestures. This intra-personal generalisation goal, which is quite a paradigm shift from many areas of machine learning and HCI, would not only offer the performer the advantage of being able to use their own hardware to capture gestures from their own gestural vocabulary and use these to control their own specific audio software, it would also result in the requirement for a lower number of training examples per gesture - leading to a reduced amount of time spent in data collection and computational-training time.

4.2 Rapid Training, Testing & Prototyping

Any machine learning algorithm that can be quickly trained with a small number of training examples is extremely beneficial to a performer. An efficient training phase enables a performer to quickly decide upon a possible gestural vocabulary to use, train the recognition system and then, importantly, test the real-time prediction abilities of the system by performing the gestures and checking if they are correctly classified. Testing the system in this manner not only validates if a robust intra-personal generalisation error has been achieved, it also tests the aesthetic and practical validity of the gestures themselves. If a performer is unhappy with either then they can either change the feature extraction method or parameters of the machine learning algorithm being used and retrain the model. Alternatively the performer could scrap one or more of the gestures and replace them with more suitable movements. A recognition system that can be quickly trained and tested allows a musician to rapidly prototype any action-sound relationship they think may be useful for a real-time performance scenario, test the validity of such gestures and then focus their time on the musical elements of the performance instead of spending hours training a system to recognise their gestures only to find that the gestures do not work aesthetically or practical.

4.3 Validating An Intra-Personal Classification Algorithm

An intra-personal generalisation error would call for a new method of evaluating the classification abilities of a machine learning algorithm for MCI. For example in most machine learning applications, a large amount of data is collected from perhaps hundreds of users and the data is split into a training set and a test set. The machine learning algorithm is then trained with the training set and evaluated with the test set. If the training data is difficult or expensive to acquire, then a hold-out validation method such as K-fold cross-validation is used instead. Both of these validation methods are suitable for estimating the generalisation abilities of an algorithm that will be used in an inter-personal recognition system. For MCI however, a more suitable generalisation metric would be to use the average cross-validation error (**ACVE**) calculated by independently computing the cross-validation error for each of the P participants and then averaging this result. The ACVE is suitable for MCI because it can accurately estimate the intra-personal generalisation abilities of a machine learning algorithm, while at the same time being validated by a large number of different users.

In addition to using a quantitative error function, such as the ACVE, qualitative subjective measures can also be particularly useful for validating an algorithms potential application for MCI. This is because, as highlighted in [10], cross-validation-based approaches may be problematic under certain circumstances, such as overestimating a models quality when there are errors in the training data. In addition, cross-validation does not capture user-specific and subjective notions of cost (e.g., whether the classifier makes a mistake on an input that is highly likely to occur in performance, or on an input that can be avoided). Therefore combining both quantitative and qualitative error measures provides an appropriate method for validating algorithms for MCI.

4.4 The SEC Design Goals

Creating a recognition system that has a flexible input/ouput configuration, can be easily trained with examples from the user's own gestural vocabulary and that achieves a low intra-personal generalisation goal have therefore been the key objectives in the design and development process of the SEC. These objectives not only informed the design of the SEC blocks but they also challenged us to adapt a number of existing machine learning algorithms and develop some novel recognition algorithms specifically for musician-computer interaction. The algorithms were adapted and developed because we wanted each algorithm to be able to:

- Classify any N-dimensional signal and not be constrained to just working with one type of sensor
- Be quickly trained with a small number of training examples for each gesture
- Be capable of recognising a gesture from within a continuous stream of real-time data that also contains non-gestural data without having to train a *null-class*, such as a noise or silence class that is used in speech recognition
- Classify both static and temporal musical gestures

Several existing machine learning algorithms were adapted to meet these criteria by developing specific feature extraction methods that enabled any N-dimensional signal to be quantized and used as input to the algorithm. The training time of the HMM algorithm, for example, was significantly improved by developing multi-threaded training routines so that a unique thread was created to train the model for each individual gesture. The HMM algorithm was also adapted so that a classification threshold was computed for each gesture in the model, thus enabling the algorithm to reject any null-gesture if the log-likelihood estimate for that gesture was below a given threshold.

The SEC also features a number of novel classification algorithms that have been developed specifically for MCI, such as the Adaptive Naïve Bayes Classifier (ANBC) [14] and N-Dimensional Dynamic Time Warping (ND-DTW) [15]. Both algorithms have been specifically designed to be quickly trained with a small number of training examples, with the average training times for both algorithms on a small sized vocabulary of 10 gestures of just a few seconds.

5. APPLICATIONS OF THE SEC

5.1 Real-Time Improvisation For Piano

The SEC is being used in a collaboration between the first author and the UK based pianist Sarah Nicolls[10] to facilitate an improvised piece that is built only from live sampled piano, controlled entirely by the pianist's gestures. During the piece, the recognition algorithms in the SEC enable the performer to use subtle hand gestures to 'save' an improvised theme to an area of space located at various points above the piano keys. After a theme has been 'saved' to a space, the performer can then revisit this space at anytime and perform a number of other fine-grain hand gestures to playback the theme, warping, stretching, looping and filtering the sample all via gestural control.

5.2 Gestural Diffusion

The SEC machine learning algorithms are currently being used in an electroacoustic composition by Robyn Farah for live gestural diffusion. For this piece, the SEC algorithms

[10]www.sarahnicolls.com

have been trained to recognise a number of gestures that enable the performer to 'throw' a sound into the sonic space, with the trajectory and intensity the of gesture being used to control a number of parameters of the behavior of the sound. The performer can also use a number of two handed 'sweeping gestures' to control a large body of sounds that have already been introduced to the sonic space, pushing and pulling them around the space or removing them all together.

5.3 RadioStreams

The SEC is being used as the recognition system for an interactive installation piece called *RadioStreams*. In RadioStreams, a user can navigate around a virtual globe using intuitive pointing gestures and listen to live radio streams from each country they navigate through. If the user likes a radio station they can 'grab' and 'throw' it onto one of eight speakers located around them. When all eight speakers have been populated with a radio station the user can then create a live improvisation by playing and controlling the live radio stations using gestures similar to that of a choral conductor.

6. CONCLUSIONS

This paper presented the SEC, a machine learning toolbox that has been specifically developed for musician-computer interaction. The SEC features a large number of machine learning algorithms that can be used in real-time to recognise static postures, perform regression and classify multivariate temporal gestures. We also provided the motivation for the algorithms used for the recognition of musical gestures to achieve a low intra-personal generalisation error, as opposed to the inter-personal generalisation error that is more common in other areas of human-computer interaction.

7. REFERENCES

[1] F. Bevilacqua, F. Guédy, N. Schnell, E. Fléty, and N. Leroy. Wireless sensor interface and gesture-follower for music pedagogy. In *NIME07*, pages 124–129, New York, NY, USA, 2007. ACM.

[2] F. Bevilacqua, R. Muller, and N. Schnell. Mnm: A max/msp mapping toolbox. In *NIME05, Vancouver, BC, Canada*, 2005.

[3] F. Bevilacqua, B. Zamborlin, A. Sypniewski, N. Schnell, F. Guédy, and N. Rasamimanana. Continuous realtime gesture following and recognition. *Lecture Notes in Cimputer Science (LNCS), Gesture Embodied Communication and Human-Computer Interaction*, 2009.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Science and Business Media, Springer, 2006.

[5] A. Camurri, P. Coletta, G. Varni, and S. Ghisio. Developing multimodal interactive systems with eyesweb xmi. In *NIME07*, pages 305–308. ACM, 2007.

[6] A. Cont, T. Coduys, and C. Henry. Real-time gesture mapping in pd environment using neural networks. In *NIME04, Hamamatsu, Japan*, 2004.

[7] R. Dillon, G. Wong, and R. Ang. Virtual orchestra: An immersive computer game for fun and education. In *Proceedings of the 2006 international conference on Game research and development*, CyberGames '06, pages 215–218. Murdoch University, 2006.

[8] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Citeseer, 2001.

[9] S. Fels. Glove-talkii: A neural network interface which maps gestures to parallel formant speech synthesizer controls. *CHI'95*, 1995.

[10] R. Fiebrink. *Real-time Human Interaction with Supervised Learning Algorithms for Music Composition and Performance*. PhD thesis, School of Computer Science, Princeton University, 2011.

[11] R. Fiebrink, P. R. Cook, and D. Trueman. Play-along mapping of musical controllers. *The International Computer Music Conference (ICMC)*, 2009.

[12] R. Fiebrink, D. Trueman, and P. R. Cook. A meta-instrument for interactive, on -the-fly machine learning. *NIME09*, 2009.

[13] N. Gillian, R. B. Knapp, and S. O'Modhrain. A pattern recognition toolbox for musician computer interaction. *NIME09*, 2009.

[14] N. Gillian, R. B. Knapp, and S. O'Modhrain. An adaptive classification algorithm for semiotic musical gestures. In *the 8th Sound and Music Computing Conference*, 2011.

[15] N. Gillian, R. B. Knapp, and S. O'Modhrain. Recognition of multivariate temporal musical gestures using n-dimensional dynamic time warping. *NIME11*, 2011.

[16] A. Hofer, A. Hadjakos, and M. Muhlhauser. Gyroscope-based conducting gesture recognition. *NIME09*, 2009.

[17] M. Lee, A. Freed, and D. Wessel. Neural networks for simultaneous classification and parameter estimation in musical instrument control. *Adaptive Learning Systems*, 1706:244–255, 1992.

[18] D. J. Merrill and J. A. Paradiso. Personalization, expressivity, and learnability of an implicit mapping strategy for physical interfaces. *Proceedings of CHI 2005 Conference on Human Factors in Computing Systems*, 2005.

[19] P. Modler, T. Myatt, and M. Saup. An experimental set of hand gestures for expressive control of musical parameters in realtime. In *NIME03*, 2003.

[20] R. Morales-Mazanares, E. F. Morales, and D. Wessel. Combining audio and gesture for a real-time improviser. *in International Computer Music Conference, (Barcelona, 2005), ICMA.*, 2005.

[21] D. Overholt, J. Thompson, L. Putnam, B. Bell, J. Kleban, B. Sturm, and J. Kuchera-Morin. A multimodal system for gesture recognition in interactive music performance. *Computer Music Journal*, 33(4):69–82, 2009.

[22] B. Pritchard and S. Fels. Grassp: Gesturally-realized audio, speech and song performance. *NIME06*, pages 272–271, 2006.

[23] N. Rasamimanana, E. Fléty, and F. Bevilacqua. Gesture analysis of violin bow strokes. *Gesture in Human-Computer Interaction and Simulation*, pages 145–155, 2006.

[24] A. Wilson and A. Bobick. Realtime online adaptive gesture recognition. In *Proceedings of the 15th International Conference on Pattern Recognition*, volume 1, pages 270 –275 vol.1, 2000.

[25] M. Wright and A. Freed. Open sound control: A new protocol for communicating with sound synthesizers. In *International Computer Music Conference*, pages 101–104, Thessaloniki, Hellas, 1997. International Computer Music Association.

[26] D. Young. Classification of common violin bowing techniques using gesture data from a playable measurement system. *NIME08*, pages 44–48, 2008.