Guitar Percussive Classification for Control of an External System

Oliver Whorwood
School of Electronic Engineering and Computer Science
Queen Mary University of London
London, UK
ec21904@qmul.ac.uk

***Abstract*—This paper outlines the design and development of a guitar percussive classification system for parameter control of effects. The need for such a system stems from a proposed requirement in conjunction with HyVibe who manufacture a guitar amplification system which uses actuators placed onto the inside of a guitar body. Currently, the way to control these parameters is either the use of physical buttons or through a separate wireless guitar pedal. The system proposed uses a CNN based approach to achieve around 75% accuracy for a realistic scenario involving triggering a control change whilst simultaneously playing a series and chords and performing some percussive guitar. A full, end-to-end system is built which successfully predicts the incoming signal in real-time and uses this to change an effect on the HyVibe system.**

I. Introduction

The direction of solving this problem will be through the use of some form of spectral processing and will not require interaction with any hardware apart from the guitar itself.

A piezoelectric pickup underneath the saddle of a guitar produces a small electrical signal which is amplified to line level by the HyVibe system, this will then be used to determine any interaction occurring on the guitar.

Several onset detection techniques will be researched, specifically relating to percussive onsets as this most closely relates to the task at hand.

A real-time classification system will need to be built on Python and used to recognise control taps and send a MIDI message to the controller to change certain parameters.

The system will be evaluated using objective measures such as F-measure for the offline system, overall latency for the real-time system and subjective ease of use of the whole system.

The aim is to have a system that can detect 2 types of control taps to send out 2 different control messages when the user is playing normal plucked or strummed guitar simultaneously.

II. Literature review

In a review of current onset detection techniques in the scope of Automatic Drum Transcription (ADT), it is noted that this task is more challenging than with pitched instruments due to the wide frequency response produced and noisy onsets (Wu et al., 2018). It groups current techniques into several methods "Segmentation-Based", "Classification-Based", "Language-Based" and "Activation-Based". "Segmentation-Based" methods tend to use simple Onset Detection functions based around the envelope of the signal. This has the advantage of being very simple to implement however this method tends to perform poorly when other types of instrument, particularly pitched, are added to the mix. "Classification-Based" methods are based around a more complex input representation combining things such as spectral centroid, RMS energy and Mel-Frequency Cepstral Coefficients. The input representation can be finely tuned to increase the accuracy of classification. This still suffers from interference of melodic instruments if these have not been introduced in the training phase."Language-Based" methods use language models such as Hidden Markov Models (HMMs) to detect patterns in the features of the audio. It is not commonly used in ADT scenarios as the structure of music does not follow patterns as closely as natural language. Modern "Activation-Based" methods use either Non-negative Matrix Factorisation (NMF) or Recurrent Neural Network (RNN) and produce easy to interpret outputs and can also be used in source separation tasks. They can suffer from degradation due to sources overlapping. This section also outlines the use of CNNs in the most recent papers. The paper concludes that the RNN based approaches tend to be the most accurate. This paper does not evaluate the CRNN and CNN methods.

Earlier methods related specifically to guitar onset detect use spectral methods (Mounir, Karsmakers and van Waterschoot, 2016). Specifically this paper compares spectral flux to a novel method of looking at the sparsity of the magnitude spectrogram. This shows an improved F-measure for the new method up to 1.000 for normal guitar notes. There is mention of filtering of the audio to remove, but it is not specified as to the specific type or amount of filtering applied.

A paper for Automatic Drum Transaction found three different methods worked better in different contexts (Southall, Stables and Hockman, 2017). These are a Soft Attention Bidirectional RNN (SA), BRNN with Peripheral Connections (PC) and a CNN. The SA involves using an RNN with weighted connections allowing weights to be brought to the output from previous time steps through hidden layers. The peripheral connection method follows a similar setup but has direct connections from previous time steps to the output. To increase the input feature size without requiring extensive computation time, a CNN system is also proposed. The evaluation uses a dataset which includes acoustically separated

drum sounds, combined drum sounds and those with other instruments included. The accuracy is determined via an F-measure. It was found that the SA method performed better for cases where the context was known i.e. just percussion or a mix of percussion and polyphonic. The CNN performed better in the scenarios where the context was unknown. The context of this paper's proposed system is unknown as the *control taps* could take place alongside other pitched sounds from the strings therefore a CNN based approach will be considered.

A CNN neural network aims to detect spacially invariant patterns (O'Shea and Nash, 2015). A filter consisting of weights is shifted along the input one element at a time (with a stride of 1) and with each iteration the result is summed . The translation invariance means that the pattern can be shifted to different positions in the input and still be detected. Ultimately, the dense layer will determine how different locations of matches are weighted. Supervised learning will take place which involves using inputs that are labelled beforehand and the model learns the patterns which relate to each label.

Classification of other types of percussive sound is explored by (Rohit, Bhattacharjee and Rao, 2021). It aims to classify hits to the Northern Indian originating tabla, consisting of two hand-struck drums. This will likely be more similar to the type of harmonic content produced from percussive guitar. One of their approaches uses a CNN model which consists of two convolution and max pooling layers followed by two dense layers with dropout, this 1-way model is tuned for each of the 4 stroke types and outperforms the more complex 3-way model.

An efficient CNN based approach for onset detection has been proposed (Schluter and Bock, 2014) which uses two convolutional filters followed by a fully connected layer. This method is used to detect downbeat positions in a range of genres of music and outperforms other competing methods such as RNN and SuperFlux. The F-measure of the CNN system is 0.885 compared to 0.873 for RNN. The reproducibility of this proposed architecture could be improved by providing open source code, however it is described is much detail. This is shown to be improved upon further by adding dropout, something which needs to be considered here, especially with a small dataset which can suffer from overfitting (Srivastava et al., 2014).

There are many sections of a guitar which can be struck to produce a sound, these can be split into harmonic and inharmonic components, the strings produce mainly harmonic content when struck whereas the body of the guitar produces mainly inharmonic content (Itoyama et al., 2007). The harmonicity refers to the relationship of frequencies where there are integer multiples of the fundamental frequency. Many guitar players also use the inharmonic components to act as percussion.

The tap control locations are designed to be as ergonomic as possible, particularly considering existing research on injury due to guitar playing (Marmaras and Zarboutis, 1997). This study suggests that for the control parts of the guitar in particular, these should not cause wrist flexion or extensions.

There has been research undertaken regarding the location and types of percussive guitar gestures (Martelloni, McPherson and Barthet, 2020). Several interviews took place to determine the most likely hand positions used for percussion. These are collated and summarised in a diagram as can be seen in Fig 1. This is a very good starting point for deciding which areas to include in a novel dataset as well as being very specific on the hand positions used in each one. The participants consisted of 4 men and 1 women, so the range of percussive styles explored may not represent the wider guitar playing community especially in other cultures.

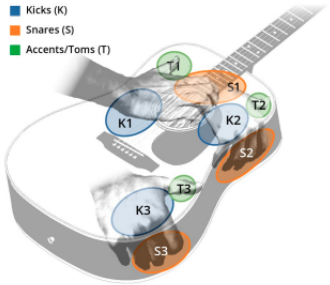

Fig. 1. Location of common percussion areas (Martelloni, McPherson and Barthet, 2020)

Datasets have been produced previously on percussive onsets occurring on different parts of a guitar as part of a design of an augmented guitar to produce percussive sounds (Martelloni, McPherson and Barthet, 2021). However, the method of recording these percussive taps is through piezoelectric sensors placed at various locations not including an under-saddle position. As the proposed *control taps* would be from areas not used for percussive guitar, it would be difficult to combine this dataset with new recordings.

There is however, a dataset that contains guitar playing through a hexaphonic pickup for use as an augmented dataset (Xi et al., 2018). This covers a wide range of genres, musical styles recorded by six professional musicians. This wide range of different notes, chords and picking/strumming patterns provides a good example of the context that the onsets would be presented in. The hexaphonic pickup is a magnetic coil type which is clipped onto the guitar which picks up vibration in the strings themselves by causing a change in the magnetic flux. This is in comparison to the piezoelectic sensor the physical vibrations in the saddle of the guitar where the compressions in the sensor produce an electrical signal (Lemme, 2009).

The latency of the system needs to be low enough to not have a detrimental effect on playability. For musicians even a delay of 10ms between musical beats (Jack, 2018) is noticeable. This paper however focusses on musical latency whereas the system proposed will be used for control of effects, so the tolerance may well be higher.

The parts of the guitar that are focussed on are the front body, divided into the lower and upper bout, the bridge, and

the side (Bay, 2013). The positions of these can be seen in Fig. 2.

The means of real-time communication is determined by the protocol type used by the HyVibe system. It has the provision of MIDI over Bluetooth Low Energy (BLE) to control certain parameters. This will make use of libraries built for this purpose, however the overall architecture must be understood. In a BLE 'connection', the 'central' device scans for packets on the network from devices which could be connected and is responsible for starting the connection. The 'peripheral' accepts connections. In this scenario the HyVibe system is the 'central' device and controls all the settings. The made benefit of this protocol is its low energy capability, with a power consumption approximately 1.5 times lower than the similar protocol 'ANT+'.

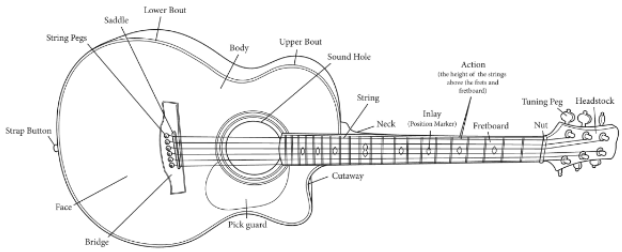

Fig. 2. Labelled parts of an acoustic guitar. [3]

III. Methodology

The system will make use of the HyVibe System Installation Kit which consists of an under-saddle piezoelectric pickup, main controller unit, a connection panel for audio and charging, and two actuators (HyVibe SA, 2021). This system is installed on an existing electroacoustic guitar, which will be referred to as the "Tanglewood", with the piezoelectric sensor replaced. The main controller unit and connection panel are not installed due to incompatibility with the existing slots in the body of the guitar. Instead, the cable for the piezoelectric sensor is fed through a slot in the body to be connected to the rest of the electronics which remain outside the guitar. This experimental setup can be seen in Fig. 3.

The first experiment is to test the performance of a CNN model to detect onsets and tune the hyperparameters. There will be one *control tap* used to trigger a parameter change on the HyVibe system. Based on the percussive guitar research, it is decided to have the *control tap* as a single-finger tap on the bridge area, to the side of the saddle, the location of which is detailed in Fig. 4. This will be known as *bridge tap*. The **initial** dataset is created which involved stopping any movement of the guitar strings by baffling them with a cloth. 100 *bridge taps* are recorded and hand annotated, these are varied as much as possible in terms of the tap velocity and the exact location of impact on the bridge, each tap uses either the 1st or 2nd finger. The dataset is shuffled to have a random selection of 80% for training with a remaining 20% reserved for offline testing.

Fig. 3. The Tanglewood guitar setup with the HyVibe system plugged in. The approximate location of the internal brace is shown in blue.

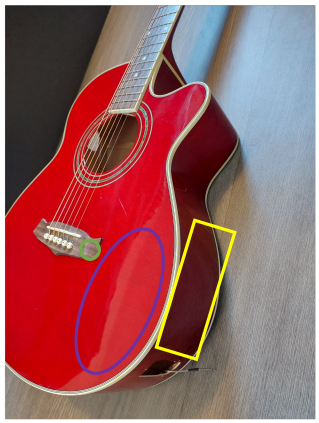

Fig. 4. Location of the control/bridge tap (green), lower-bout tap (purple) and side slap (yellow)

As seen in Fig. 5, the analogue signal output from the piezoelectric sensor is amplified by the HyVibe system before being converted to a digital signal using an external audio interface. The audio interface used is an Audient iD4 using the front D.I. with a gain setting of 5. The datasets are recording into a Digital Audio Workstation (DAW), Logic Pro, along to a metronome set at a rate of 60bpm. The audio files are recorded at a sample rate of 44.1kHz, 24-bit dynamic range in WAVE uncompressed format. The onsets occur at approximately 1 second intervals however these intervals are not accurate. Therefore, these files are imported to another DAW, Audacity, where the onsets are labelled manually and exported as a text file with each onset's absolute time position in seconds. This text file has two columns relating to the onset start and end time, as this is annotated as being instantaneous, only the first column is used.

It is important that the control taps are not too audible when compared to normal guitar playing as this could distract from a performance, the average amplitude of these compared to normal guitar playing will be calculated.

A script is developed to train a neural network model and evaluate its performance, this is written is Python using the TensorFlow library. This, along with the datasets can be found

at https://github.com/whorwood/MastersProject. To train the detection system, the audio signal is first converted to the frequency domain via a Short-Time Fourier Transform (STFT) using the 'librosa' Python library. The output is windowed using 15 time-bin wide sections shifting by 1 each time with a corresponding ground truth relating to the presence of an onset at the centre of the window.

The neural network itself is based around the structure of (Schluter and Bock, 2014). The model is started with the structure: convolution (3x7) with 10 features, maxpool (3x1), convolution (3x3) with 20 features, maxpool (3x1), flatten, Dense (256), Dense (1). The hyperparameters of the neural network are refined by testing their performance in a tap onset detection task. First, the hop size of the FFT is decreased to 1024 from 2048 improving the F-measure from 0.9895 to 0.9967. This is decreased again by half to check this trend, however a hop size of 512 reduces the F-measure down to 0.980 so a hop size of 1024 is fixed. Next, the size of the filters are increased to see their effect on accuracy. The hypothesis is that this will increase the accuracy as the filter size will then be more proportional to the input size. The filters are set to sizes of (12,7) and (12,3). With all other parameters unchanged, this in fact decreases the detection accuracy to an F-measure of 0.9931. The network depth is also reduced to a single layer to assess the impact and as expected this also decreases the overall accuracy to 0.9861, not by a huge margin however the initial structure must be as refined as possible. This simplification of the network will of course reduce computational time so is considered for the real-time system. Finally, the number of neurons is the final dense layer is both decreased to 128 and increased to 512, both with the effect of decreasing overall accuracy.

The model and weights from each training session are saved for future evaluation. The model itself is saved in JSON, a human-readable data format designed to be lightweight and cross-compatible (Nurseitov et al., 2009) and the weights are contained within the Keras specific H5 format.

Two non-control onsets are now introduced, a single-finger tap on the lower-bout area and a slap-like gesture using the underside of the hand on the side of the guitar body, both of which are illustrated again in Fig. 4. These are two common areas where percussive guitar would occur, thus it is important to see if a system could detect the differences between each type of onset location. This dataset now consists of 100 *bridge taps*, 100 *lower-bout taps* and 100 *side slaps*. Again, the strings are baffled in all scenarios hence the dataset name **Muted_String**. The ground truth is now an array of 3 integers, corresponding to an occurrence of each type of onset. The output of the model is modified to a multi-label sigmoid function, so dense(3).

Next, the **Muted_String** dataset is replicated but without the strings being muted, so they are free to resonate as they would in normal guitar playing, this is called **Open_String**. No fingerboard fingering occurs during this dataset so not the resonance of the strings is related to the most commonly used open tuning (Sparks, 1997), E (82.41 Hz), A (110.00 Hz), D (146.83 Hz), G (196.00 Hz), B (246.94 Hz), E (329.63 Hz) (Suits, 2022). The model is then evaluated on this new dataset by training and testing.

A more realistic scenario for the *control taps* to occur is whilst or directly after the strings have been strummed or picked with various fingerings on the fretboard. To test the model's performance on this in a controlled manner, an **Augmented** dataset is created. This is done by overlaying the **Open_String** dataset with sections from the "GuitarSet" dataset. The "GuitarSet" dataset is created using a hexaphonic pickup which records each string individually. These 6 tracks are summed together to create a single output like that of a normal mono-output pickup. The gain of the audio being overlaid is adjusted to be at the same relative amplitude as the onsets. This is achieved by recording a similar style of playing on the acoustic guitar and measuring the gain, then matching the gain of the "GuitarSet" to this. Excepts from the 5 musical styles in the dataset from the first player are overlaid onto each track.

As it is now established how well the model can perform at detecting each separate type of onset, the model is simplified into a binary classifier either detecting a *control tap* or anything else (guitar playing, *lower-bout taps* or *side slaps*). This is first evaluated on the **Open_String** dataset, then this dataset is appended by a further 100 *control taps* to ensure that when performing the binary classification the dataset is balanced between the two classes. This is called **Open_String_Balanced**. These additional *control taps* are overlaid with the same **GuitarSet** excerpts as before, creating an **Augmented_Balanced** dataset to be compared with the multi-label classification.

The dataset is still fairly limited with 400 total onsets, to increase the diversity of the dataset a further 200 *control taps*, 100 *lower-bout taps* and 100 *side slaps* are recorded and annotated. As before these are augmented, creating an **Augmented_Extended** dataset.

Another guitar, manufactured by *Lag* and with a HyVibe system installed is used to create a new dataset following exactly the same procedure as **Open_String_Balanced**.

To test the feasibility of a real-time classification system, a prototype was developed to recognise the onsets of three positions of tapping in real time and send one MIDI message to the HyVibe system. The real time detection system is written in Python and sends serial messages to an ESP32 development board which is programmed in C++ to send a MIDI message whenever a serial message of '1' is received. If a threshold of 0.5 is reached for a control tap prediction, and the previous prediction was 0, then the serial message is sent. This then triggers the MIDI message of type 'note-off' to be sent to the HyVibe system over BLE.

A special dataset is created to test the taps along/away from the internal brace structure which involved placing tape as close as possible along the location of the brace and creating a recording of 100 taps along this brace and another 100 not along the brace. These two datasets are named **Along_Brace** and **Away_Brace** respectively.

The final stage of testing involves using the *Lag* guitar as an end-user would with a combination of typical guitar playing, percussive taps and control taps. To maximize reproducibility potential, a defined set of chords are played along with specific tapping locations. The order of playing is as follows: G Major, D Major, A Minor, *bridge tap*, G Major, D Major, A Minor, *lower-bout tap*, G Major, D Major, A Minor, *side slap* with 1 second between each event. This entire sequence is repeated 25 times strummed using a plectrum and another 25 times strummed using a finger.

To check that the real-time system is performing as expected in an automated manner, pre-recorded audio from a dataset that the model being used is trained on is looped back into the audio interface to simulate a real-time playback on the guitar. Each onset detected sends out two MIDI messages, a note on followed by note off which is recorded in the DAW under a virtual instrument track. These message locations are then checked against the onsets in the recording to calculate an F-measure.

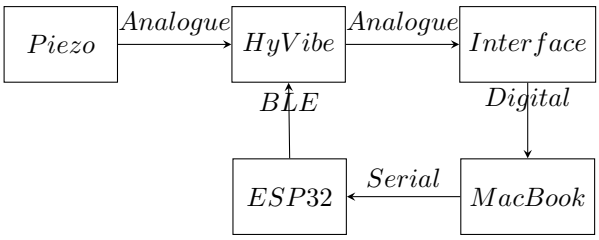

Fig. 5. An overview of the system architecture.

IV. Results & Discussion

The best accuracy for the hyperparameter selection for the initial tests was found to be 0.9934 with a (3x7) and (3x3) filter size, FFT size of 1024 and dense layer of 256 neurons. At a sample rate of 44.1kHz, this relates to an actual window width of 93ms, which is already established as being ideal for music related tasks (McFee et al., 2015).

The first evaluation involves the **Muted_String** dataset. This was run for 10 epochs with a batch size of 32, the epochs are established based on the number required for the loss to reach the 'knee'. The F-measure for each class is 0.9770, 0.9691 and 0.9711 with an average of 0.9724. There are more false positives than false negatives for the *control tap* and *lower-bout tap* but the opposite for *side slap* suggesting that there is more similar frequency and spatial characteristic between the *control taps* and *lower-bout taps* and that *side slap* has a more unique response. This intuitively makes sense as the location of the *side slap* is on a completely different face of the guitar as well as using the back of the hand rather than a finger tap.

In terms of the amplitude of the onsets, these on average sit around the -32dB point for the open string *control taps*. There will be a huge range of amplitude for guitar playing, but a test of playing chords with a plectrum had a range of approximately -24dB to -22dB. This at most represents a 10-fold increase in the SPL of the signal which is more than enough to be masked perceptually almost completely.

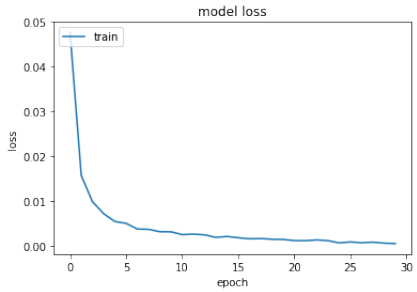

Fig. 6. Loss function over time for the multi-label classification of the **Open_String** dataset

For the **Open_String** dataset, after 30 epochs the F-measures are 0.9877, 0.9797 and 0.9691, with an average of 0.9788. The loss function for this training can be seen in Fig. 6. The initial assumption was that this would decrease the accuracy with the added frequencies from the resonating strings however there is actually a slight increase. When looking at the spectrogram of the guitar onsets with the string sounds compared with none, the main structure of the onset is fairly low frequency, so the resonance does not interact enough for differentiation of the onsets to be obscured.

For the **Augmented** dataset, after 30 epochs the F-measures are 0.6143, 0.5973 and 0.5688, and average of 0.5935. This augmentation causes a significant decrease in accuracy as now the signal overlaying the onsets is higher in amplitude than the freely resonating strings before. The number of epochs is increased to 50 which improves the average F-measure to 0.6639.

Next, for the binary classification model, for **Open_String** this achieves an F-measure of 0.9782, and **Open_String_Balanced**: 0.9901. The performance remains similar to the multi-label classfication for the unbalanced dataset which suggests that actually all three onset types can be differentiated well otherwise the model would see a bigger increase in performance when the two percussive types are combined to one label. Having a balanced dataset increases the accuracy quite a lot as before there was probably bias towards the other onset types.

Augmented_Balanced has an F-measure of 0.8533. As expected, this is less accurate as there are now instances of other frequency content overlapping the control taps making them harder to distinguish. A good example of this can be seen in Fig. .

Augmented_Extended has an F-measure of 0.8802. It can be seen here that in this instance, the larger dataset increased accuracy by a small margin. This is likely due to the increase in variation within the dataset and an increased generalisation as a result. This is increased even further still with a random dropout of 20% added between the two dense layers to an F-measure of 0.888. Being still a relatively small dataset, this helps to reduce overfitting.

The real-time script has to complete each prediction step in time for the next audio buffer. Currently, that script is being executed on a 2.9 GHz Quad-Core Intel Core i7 based system. Initially, each audio buffer was incrementing by a single STFT time bin each iteration which did not allow sufficient time for the predictions to take place. The other extreme considered was to increment by the full network input width of fifteen bins, however if the onset extends across the boundary of two windows, there might not be enough information for a correct prediction. When looking at the spectrogram output, it was noticed that the majority of the frequency content of the onset is contained within three consecutive bins. Therefore, a compromise was made of iterating 12 bins so that there is always an overlap of 3.

The real-time system is also tested for latency in three scenarios, first when triggered by an onset tap, then from a keyboard press from Python, and finally a button press directly from the ESP32 board. These scenarios will determine where the sources of latency occur. The latency for the fist scenario is measured by recording the audio output from the guitar and measuring the time difference between the first audio peak in the onset and the first audio peak in the metronome. The second scenario is measured by sending a MIDI message internally to the DAW on a key press (enter) which is recorded alongside the audio output. For the third scenario the button is pressed at the same time as a note on a MIDI keyboard, which is recorded in the DAW. Each scenario is tested 5 times and the average latency is calculated. For the onset trigger, the total latency is a fairly substantial 1167.4ms. Comparing this with sending the serial message directly from the Python script via a keyboard press, the latency is 1037.4ms. Finally, when the button is used, the latency is less than 1ms. There is a serious bottleneck in the system at the serial communication layer. The onset detection itself only accounts for an average of 130ms of latency. Other methods to relay the signal from Python may have to be considered.

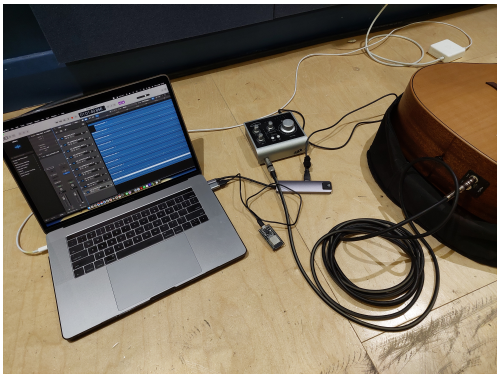

Fig. 7. The experimental setup for the LAG guitar

With an initial subjective test of the real-time system, it seemed that percussive taps occurring at positions over the internal braces were more likely to be misclassified as control taps, hypothesised that due to the acoustic resonance here being more similar to the bridge which is also structurally attached to the braces. The is evaluated further by creating 2 new datasets of percussive taps, one with taps occurring above the internal braces and one with taps occurring away from the internal braces. The F-measure is then calculated for each of these datasets. Figure shows the approximate location of these braces in the guitar under test, this will vary with different guitars, but a similar principal can be applied. This hypothesis is disproven as the F-measure for the dataset over the brace is actually slightly higher than the one away from the brace at 0.9709 compared to 0.9518. It could be that the proximity of the tap to the bridge has more of an effect than whether it occurs over a brace, this would need to be verified by further experimentation.

When using the weights trained on the **Open_String_Balanced** dataset and using this to predict control onsets on the LAG guitar, this performed poorly with an F-measure of 0.240. This can be improved slightly be decreasing the classification threshold from 0.5 to 0.1 reaching a maximum F-measure of 0.264. This shows that the harmonic resonance produced by taps to the equivalent locations on different guitars vary significantly and that the generalisation of the model is low. The reverse of this process is also tested, so using the LAG trained model to predict onsets in the **Open_String_Balanced** dataset. The F-measure was slightly higher at 0.4547 and did not increase with lower threshold values. When observing the spectrogram representation of an example of a *control tap* on the *Tanglewood* guitar compared to the *LAG*, it can be seen that the frequency response varies a lot between the two.

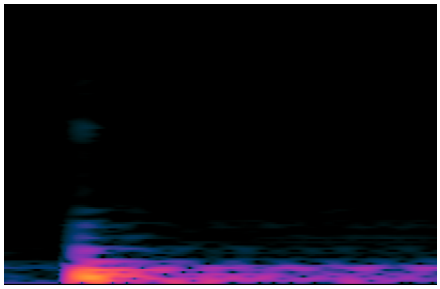

Fig. 8. Labelled parts of an acoustic guitar. [3]

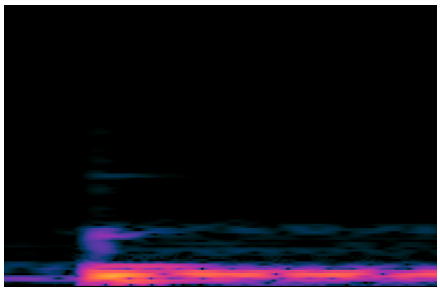

Fig. 9. Labelled parts of an acoustic guitar. [3]

The models trained on the augmented datasets are also tested in the same manner. For the LAG_Augmented trained

used on Augmented_Balanced, this achieved a maximum F-measure of 0.1358. For the other way round, the F-measure was even lower at 0.0773. This is similar to what is seen with the open string datasets and further suggests that the frequency characteristics vary a lot between the two guitars.

Both these tests show that the models trained on a single guitar type are too overfit when used to predict onsets on another guitar. To try to combat this, the model is trained on a combination of datasets from both guitars. Before, the training/test split is undertaken after the shuffling of the dataset, however now the datasets for both guitars must be kept separate for evaluation, so the shuffling occurs after the training/test split. The combined model is tested against the test set from both the Open_String as well as the LAG and the performance improved significantly achieving an F-measure of 0.9852 and 0.9832 respectively.

The weights trained on the augmented dataset are evaluated further by using the **LAG_playing** dataset, a realistic scenario involving a mixture of guitar playing and different taps. This again performed quite poorly with an F-measure of 0.273 at a threshold of 0.1. This was slightly surprising as the onsets themselves were produced in the same way as the open string dataset. The guitar playing sections are slightly higher amplitude than the average for the augmented audio causing different masking of the onsets.

The model is then trained using the **LAG_playing** dataset, which is the most likely continuing steps for implementing a releasable version of the system. For this to be realised fully, the dataset would have to be expanded across many different guitars and players, and playing techniques. When using this single dataset from one player on one guitar, an F-measure of 0.752 is achieved. The dataset would ideally be expanded to including a playing scenario on the *tanglewood* guitar however this could not be achieved due to the HyVibe system not being installed securely in this guitar meaning it has to remain on a flat surface during dataset collection.

The real-time system evaluation had an F-measure of 1, and there was perfect detection of all onsets. This is expected as the real-time system uses the same model implementation with the only difference being the method and libraries used for reading in the real-time audio stream.

Subjectively, using the system to control effects was quite temperamental in terms of the accuracy and false triggers which impeded the natural flow of guitar playing. Once the script is executed, this keeps running without any faults or errors and the predictions reliably complete on time.

V. Conclusions & Further Work

A CNN based approach developed for detecting downbeats in music transfers well to the task of detecting percussive onsets from a hollow bodied acoustic guitar.

The latency of the overall prototype system is high which for controlling some parameters such as starting and stopping loops will be much more noticeable than turning on and off effects such as reverb. This latency however is due to a serial interface between two devices which would be negated in a system with inbuilt BLE capability. The system works as expected as a whole with a complete solution from the control tap, through the real-time detection system which successfully changes parameters on the HyVibe system.

There is insignificant difference in performance when taps are positioned around internal structures in the guitar, in this case the brace connected to the bridge.

Changing guitar has a large effect on the accuracy of trained models and don't generalise well. By using a combined dataset for training, this improved the performance when using this model to predict for both guitars.

The realistic playing dataset would need to be expanded significantly to create one that was representative of a variety of guitar players and styles. With the datasets and models provided open source, this has the potential to be expanded by further studies.

The *control tap* is not noticeable when combined with 'normal' guitar playing due to the large difference in amplitude. The system was able to successfully detect taps that have minimal force.

To determine which locations and techniques for control taps would be the most suitable for guitar players, a user study would have to take place. The original aim of having two control taps is not met, however as it is determined that the system can successfully differentiate between different onset types, it is likely that a second control tap would be able to be introduced.

References

[1] Schluter, J. and Bock, S. (2014) "Improved musical onset detection with Convolutional Neural Networks", 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). doi: 10.1109/icassp.2014.6854953.

[2] Wu, C., Dittmar, C., Southall, C., Vogl, R., Widmer, G., Hockman, J., Muller, M. and Lerch, A. (2018) "A Review of Automatic Drum Transcription", IEEE/ACM Transactions on Audio, Speech, and Language Processing, 26(9), pp. 1457-1483. doi: 10.1109/taslp.2018.2830113.

[3] Bay, C. (2013) Glossary of guitar terms. Pacific, MO: Mel Bay.

[4] Jack, R., Mehrabi, A., Stockman, T. and McPherson, A. (2018) "Action-sound Latency and the Perceived Quality of Digital Musical Instruments", Music Perception, 36(1), pp. 109-128. doi: 10.1525/mp.2018.36.1.109.

[5] Rohit, A., Bhattacharjee, A. and Rao, P. (2021) "FOUR-WAY CLASSIFICATION OF TABLA STROKES WITH MODELS ADAPTED FROM AUTOMATIC DRUM TRANSCRIPTION", ISMIR.

[6] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014) "Dropout: a simple way to prevent neural networks from overfitting", Journal of Machine Learning Research, 15(1), pp. 1929–1958.

[7] HyVibe SA (2021) HyVibe System Installation Instructions. Available at: https://www.hyvibeguitar.com/wp-content/uploads/2021/05/HyVibe-Installation-Instructions-EN.pdf (Accessed: 12 August 2022).

[8] HyVibe SA (2022) MIDI Instructions. Available at: https://www.hyvibeguitar.com/airturn-midi-instructions/ (Accessed: 12 August 2022).

[9] Southall, C., Stables, R. and Hockman, J. (2017) "Automatic Drum Transcription for Polyphonic Recordings Using Soft Attention Mechanisms and Convolutional Neural Networks", ISMIR.

[10] Xi, Q., Bittner, R., Pauwels, J., Ye, X. and Bello, J. (2018) "Guitarset: A Dataset for Guitar Transcription", in 19th International Society for Music Information Retrieval Conference. Paris, France.

[11] O'Shea, K. and Nash, R. (2015) "An introduction to convolutional neural networks.", ArXiv, abs/1511.08458.

[12] Tosi, J., Taffoni, F., Santacatterina, M., Sannino, R. and Formica, D. (2017) “Performance Evaluation of Bluetooth Low Energy: A Systematic Review”, Sensors, 17(12), p. 2898. doi: 10.3390/s17122898.

[13] Martelloni, A., McPherson, A. and Barthet, M. (2020) ”Percussive Fingerstyle Guitar through the Lens of NIME: an Interview Study”, New Interfaces for Musical Expression (NIME).

[14] Marmaras, N. and Zarboutis, N. (1997) “Ergonomic redesign of the electric guitar”, Applied Ergonomics, 28(1), pp. 59-67. doi: 10.1016/s0003-6870(96)00032-4.

[15] Ellinger, J. (2014) MIDI Basics. Available at: https://people.carleton.edu/ jellinge/m208w14/pdf/02MIDIBasics_doc.pdf (Accessed: 12 August 2022).

[16] Nurseitov, N., Paulson, M., Reynolds, R. and Izurieta, C. (2009) “Comparison of JSON and XML Data Interchange Formats: A Case Study”, Caine, 9, pp. 157-162.

[17] Sparks, P. (1997) “Guitar Performance in the Nineteenth Centuries and Twentieth Centuries”, Performance Practice Review, 10(1), pp. 71-79. doi: 10.5642/perfpr.199710.01.07.

[18] Suits, B. (2022) Frequencies of Musical Notes, A4 = 440 Hz, Physics of Music Notes. Available at: https://pages.mtu.edu/ suits/notefreqs.html (Accessed: 12 August 2022).

[19] Itoyama, K., Goto, M., Komatani, K., Ogata, T. and Okuno, H. (2007) ”Integration and Adaptation of Harmonic and Inharmonic Models for Separating Polyphonic Musical Signals”, 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP ’07. doi: 10.1109/icassp.2007.366615.

[20] McFee, B., Raffel, C., Liang, D., Ellis, D., McVicar, M., Battenberg, E. and Nieto, O. (2015) ”librosa: Audio and music signal analysis in python”, Proceedings of the 14th python in science conference, 8, pp. 18-25.

[21] Martelloni, A., McPherson, A. and Barthet, M. (2021) ”Guitar augmentation for Percussive Fingerstyle: Combining self-reflexive practice and user-centred design”, Proceedings of the International Conference on New Interfaces for Musical Expression.

[22] Mounir, M., Karsmakers, P. and van Waterschoot, T. (2016) ”Guitar note onset detection based on a spectral sparsity measure”, 2016 24th European Signal Processing Conference (EUSIPCO). doi: 10.1109/eusipco.2016.7760394.

[23] Lemme, H. (2009) BuildYourGuitar.com :: The Secrets of Electric Guitar Pickups. Available at: https://a.pomf.cat/zjnuis.pdf (Accessed: 13 August 2022).