VIRTUAL MUSIC TUTOR

Interim Report

JANUARY 20, 2019
MO CHEUK YIN, ZHONG SHUN
# Table of Contents

Table of Contents.................................................................................................................. 1  
Table of Figures...................................................................................................................... 1  
Project Background................................................................................................................. 2  
Project Objective................................................................................................................... 3  
Project Methodology............................................................................................................... 4  
   Existing approach for guitar transcription ......................................................................... 4  
   Proposed modules .............................................................................................................. 5  
   Challenges and limitations ................................................................................................. 5  
   Support for multi-threading ............................................................................................... 6  
   Support for polyphonic detection ....................................................................................... 6  
   Lack of data to classify good technique playing from a bad one ....................................... 6  
What has been accomplished................................................................................................. 7  
   Frontend prototype .......................................................................................................... 7  
   F0 estimation (pitch tracking) ........................................................................................... 8  
   Technique classification .................................................................................................... 9  
      Convolutional neural network (CNN) ........................................................................... 9  
      Support Vector Machine (SVM) .................................................................................... 10  
   Evaluation ......................................................................................................................... 11  
What will be done .................................................................................................................. 13

# Table of Figures

Figure 1. Workflow of TENT ................................................................................................. 4  
Figure 2. Yousician User Interface ..................................................................................... 6  
Figure 3. Virtual Music Tutor Frontend Prototype .............................................................. 7  
Figure 4. CNN architecture ................................................................................................. 10  
Figure 5. CNN classification report (Hamm = Hammer-on, Pull = Pull-off) .................. 11  
Figure 6. SVM classification report ................................................................................... 11  
Figure 7. CNN normalized confusion matrix ..................................................................... 12  
Figure 8. SVM normalized confusion matrix ..................................................................... 12
Project Background

Research in artificial intelligence has impacted various fields in our life ranging from medicine, finance, robotics, machinery, to everyday consumer retailing, advertisement, urban planning, etc. In such fields, problems tend to be well formalized where a clear-cut metric can be devised to prove/disprove various approaches of AI and allow researchers to improve the algorithms. Music which is traditionally deemed a very subjective domain does not enjoy such advantages and therefore lacks objective metrics to quantify how good/bad one’s playing is. Using machines to gauge a person’s playing and provide constructive feedback can be done only to a limited extent. That is why music education heavily relies on tutors to assess the playing of students with their subjective perceptions.

People have resorted to technologies when it comes to music education, albeit to a very limited extent. Metrics such as volume, tempo, rhythm, etc. have been traditionally used as a rough guide to analyse a piece of music and there are existing music tutoring products that incorporate these approaches to assess a student’s playing. Branching teaching programs (e.g GUIDO ear-training system) (Holland, 2000) were developed to present pre-recorded materials to students, receive responses from them and compare literally the response with predefined answers, after which a certain branch will be selected to present the next materials. This is a very rigid process that lacks the essential interactive elements in music education that is required to tailor-made teachings according to a student’s performance and address the area for improvement. These programs are also limited to the field of music theory, for example ear trainings specifically because it more often than not requires only objective standardised answers. Match My Sound is a company that offers web API that matches students’ playing with the pre-set score or MIDI track in the system and provides automatic assessment. Simple note per note assessment and chord detection are implemented using signal processing to retrieve music information. However, the choice of using score or MIDI as the standard base limits the comparison and as a result it is not able to detect more advanced but common techniques in instruments, for example, the bending and sliding in guitar. Yousician is another company that provides similar services but on the mobile platforms. It gamifies guitar learning using very interactive UI but again, the assessment is limited to simple rhythm and chord recognition based on score or MIDI. Soundslice solves another problem of music learning by making it easier to accurately align music score with music tracks so that people can practice with much more ease because they can see what to play and how each bar sounds.

Given that learning different instrument presents diverse challenges and there are very few product in the market that focuses solely on guitar learning and the problems beginners will encounter. This challenges our team to come up with a truly intelligent “Virtual Music Tutor” which detects not only simple tempo/chord correctness in a student’s playing but also more nuanced subtle differences in the student’s playing compared to that of a tutor.
Project Objective

This project is to develop a system which can judge how similar a student's performance is to his tutor's. We want to target only on electric guitar with a focus on beginners’ note accuracy, rhythm, and most importantly guitar techniques, which is the most value-added feature that distinguishes our project from music tutor applications on the market place right now. Since no existing product on the market can assess guitar techniques accurately, our project can be a pioneer in the field and provide students practicing guitar a more convenient, and efficient way to grasp important but tricky techniques on their own.

The process of picking up guitar techniques is both tricky and confusing. Beginners oftentimes apply a technique wrongly or mix up different techniques because it is very hard for them to tell whether they are playing it correctly. Given that most practices happen at home without the oversight of a music tutor, the mistake will perpetuate throughout the week and not get corrected until the next time they visit their tutors, by when bad playing habits will have been formed and the mistake will have been deep-rooted. This will make correction afterwards much harder and learning progress much slower.

With this pain point of music tutoring in mind, we want to come up with a solution to nip beginners’ mistakes in the bud. A real-time application that receives a student’s playing track and compares it with a tutor's playing to provide instant feedback has the potential to solve the problem. We assume background noise to be a minimum and the tempo will not exceed BPM 120 for our application to have as short delays as possible.

There are indeed many aspects of music playing for an overall assessment, for example, note accuracy, chord with variation of voicings along the guitar neck, dynamics, rhythm, expressiveness, guitar accessories usage, etc. However, each of this can be a separate research topic and in our project, we choose to focus on monophonic guitar solo with no guitar effect other than overdrive/distortion and we evaluate only on single notes combined with proper technique.

In our interim stage, we only evaluate note by note with techniques. This will be further improved in the final stage when we will combine the above with rhythm assessment.
Project Methodology

The main task of the project is to evaluate guitar playing performed by a student, given a reference audio track or notation provided by a guitar tutor. We limit the scope of the project to only evaluate single note played with 7 classes of techniques, namely, bend, slide, trill (vibrato), hammer-on, pull-off, mute and normal (no technique).

Existing approach for guitar transcription

There are two important papers that give great guidance to our team on how to implement a guitar-focused performance evaluation system. Chen, Su & Yang (2015) proposed a framework for monophonic technique detection using Support Vector Machine (SVM) based on technique candidate selection from a fundamental frequency (F0) sequence. Su, Chen, Su and Yang (under review) later extended the work and proposed Technique-Embedded Note Tracking (TENT) for transcribing a monophonic guitar solo into notes with techniques from a F0 sequence extracted. They also implemented the technique classifier using convolutional neural network (CNN) rather than SVM.

Although they are doing guitar transcription, which seems to be different than our goal of evaluating guitar performance, yet the task of evaluation can be viewed as a comparison of notes transcribed by the system with certain levels of tolerance. Hence, the two papers provide valuable insight and direction to our project.

Figure 1. Workflow of TENT. Adapted from “TENT: Technique- Embedded Note Tracking for Real-world Guitar Solo Recordings” by T.-W. Su, Y.-P. Chen, L. Su and Y.-H. Yang, under review

In these two papers, it is assumed that we have a monophonic guitar solo audio track, and by extracting the melody of it or simply run it through a fundamental frequency estimation algorithm, we get a list of F0 estimations. Afterwards, we devise hand-crafted rules for selecting various segments of sequences that is potentially a technique played and hence considered candidates. This is the preliminary round of identifying playing techniques. With the help from a note tracking
algorithm, we can therefore identify notes from pitch sequences with their potential techniques. Finally, by feeding the candidates to a pre-trained data-driven technique classifier, we can classify the techniques played and arrive at a transcription of notes with their intended techniques.

We decide to adopt methods these researchers used to do pitch tracking, note tracking and technique classification in our project.

Proposed modules

Our system is comprised of a frontend component and a backend component. The frontend component, a web interface, will be responsible for handling user interactions, audio recording and showing instant evaluation results (e.g. what note and technique is being played) and final overall performance. The audio recorded here will be streamed and sent to server for evaluation. Our goal is to make this easy to use, so built-in microphone of a laptop or tablet is the most preferable choice. However, in order to eliminate the noise factor in the initial development, we will use a USB audio interface to capture guitar input to create a controlled environment for testing first.

The backend component is the main module responsible for evaluating student’s performance. We have two approaches on evaluation on student’s playing – note-by-note or synchronized evaluation.

Note-by-note evaluation refers to the approach where guitar playing is streamed to the backend and evaluation is performed at a regular interval (e.g. per 0.2 seconds) and intermediate evaluation is returned. The student hence can learn to play through the score based on logic which allows him/her to pass through each note given certain level of tolerance. The interface would make the note green if it is played correctly and red if not. This approach is to model a real-life scenario where guitar tutor will go through the score note by note and teach how to play each note for a beginner student.

The synchronized evaluation approach is more suitable for student practices, where rhythm of each note will also be assessed. The interface is similar to that of Yousician, where a student will be guided to play each note on the score with a metronome on.
We are still designing the details of each evaluation approach and will further explain this in the final report.

Challenges and limitations

Support for multi-threading

As potentially a lot of data will be streamed to the server from many users, handling requests and return evaluation responses at the same time on the same thread is not a wise choice. A multi-threaded approach needs to be employed to make sure it can maintain a certain performance level, especially in real-time. We need some time to design and implement this as we are not skilful in multi-threaded programming.

Support for polyphonic detection

Another elementary guitar playing skills apart from soloing, strumming, is also a very important aspect for a beginner guitar player. Strumming refers to “brushing over several strings to make sounds” and the result is a chord (3 individual notes or above combined). However our system completely rules out the evaluation of this as the complexity of polyphonic detection needs a whole new set of methodology. It will be in our future plan to support polyphonic evaluation.

Lack of data to classify good technique playing from a bad one

The proposed technique classification approach only classifies which types of technique is being played, but not how good a certain technique is played. As there is no existing dataset of guitar techniques which has good and bad labels on the same technique, it is very difficult for us to use the existing framework to train a classifier to identify good and bad techniques. Developing a technique dataset that includes good and bad techniques played by students can certainly be a good future direction to improve guitar playing evaluation.
What has been accomplished

In short, we have created a frontend prototype, experimented with 3 algorithms (pYin, Melodia and CREPE) in pitch tracking, i.e. F0 estimation and also 2 technique classification methods (CNN and SVM) which are suggested by the aforementioned papers.

Frontend prototype

![Virtual Music Tutor Frontend Prototype](image)

The frontend prototype is a web application built with Ionic and Stencil. As Figure 3 shows, the reference notation (either transcribed partially by existing approach or provided by tutor) is displayed above. The “note”, “technique” and “comment” sections are there to give instant responses to student as they are playing through the microphone. “Overall performance” shows the overall performance of student’s playing in terms of areas like note accuracy, technique accuracy. In the future, we hope to also highlight portions that the student has played wrong on the score so she can refer back to them and practice according to mistakes she has made.

1 [https://github.com/ionic-team/ionic-pwa-toolkit](https://github.com/ionic-team/ionic-pwa-toolkit)
F0 estimation (pitch tracking)

We have experimented with three algorithms for fundamental frequency estimation/melody tracking.

Melodia (Salamon, Gómez, Ellis, & Richard, 2014) tracks the melody contour by firstly extracting possible frequencies at each point in time via fourier transform. It then search for harmonic series of frequencies and determine the most possible pitch. Pitch contour can be generated and from which melody selection can be done based on the calculation of contour characteristics. Non-melodic contours will be filtered out. Unlike the following two monophonic algorithms, Melodia performs best when it is used to extract melody from polyphonic tracks, usually songs with accompaniment.

pYIN (Mauch & Dixon, 2014) is a probabilistic approach of the YIN algorithm to estimate fundamental frequency for speech and music. The YIN algorithm is a simple and effective frame-wise monophonic F0 estimation. It produces a single pitch estimate per frame based on a threshold parameter. pYIN replaces the threshold parameter of the original algorithm by a distribution and makes use of a hidden Markov model to find multiple fundamental frequency candidates associated with probabilities. pYIN achieves a more robust result due to this better candidate selection method.

CREPE (Kim, Salamon, Li, & Bello, n.d.), unlike the above two algorithms that operates on the frequency domain, is a deep convolutional neural network that processes directly on the time-domain waveform input to produce pitch estimate. There are six convolutional layers for a 1024-sample excerpt from the time-domain audio signal. It is very robust against noise and produces a better result than pYIN.

We are still looking for quantitative methods to evaluate these algorithms because very few guitar dataset comes with timestamped ground truth for evaluation. In our initial qualitative assessment, CREPE seems to produce more accurate results but it also requires intensive computational resources.
Technique classification

We compared the performance of two classification methods, CNN and SVM based on the confusion matrix and precision, recall and f-scores of the evaluation results. The dataset used is the Guitar Playing Techniques (GPT) dataset (Su, Yu & Yang, 2014), which contains 6,580 clips of single notes played with 7 frequently used playing techniques (bend, slide, hammer-on, trill, pull-off, mute and normal) and 7 different tones (combinations of distortion and overdrive). Specifically, the transient samples, i.e. the segments of transition between two notes, rather than the samples of a complete note, were used to trained the classifier. The reason is that most techniques occur in the attack phase of a note, which lies in the transition between two notes. The dataset covers a wide range of guitar timbre and a wide register of notes, which help to provide data diversity. This dataset is however less useful to tasks like solo transcription because they are note real-life examples of notes played in guitar solo.

There are in total 7797 transient 0.3 second samples. Due to a small bug in the implementation, of the 7797 samples, only 7299 of them has been included in the training process described below, but as this will not be impacted the evaluation results in a great extent and due to time constraints, we will evaluate the performance again after fixing the data pre-processing bug and present the statistics in the final report.

Convolutional neural network (CNN)

A CNN is implemented according to Su, Chen, Su and Yang (under review). In their paper, the CNN is trained using a different dataset which contains more real-life technique samples but only 1 guitar timbre, overdrive tone, is included. The purpose of the classifier is still the same, which is to identify the technique based on, for instance, a 0.2 second audio excerpt.

As shown in the paper, MFCC features, which are commonly used to represent timbre, and the pitch contour features, which are log-scale floating-point numbers based on MIDI note numbers converted from frequencies tracked, are most effective in being the input features. The first 13 MFCCs together with the z-normalized pitch contour are extracted from the audio excerpts.

The input features are extracted from each audio excerpt in the dataset and are then fit into the CNN as illustrated below with 2 1D convolution layers with pooling layers behind and a number of fully-connected layers, trained with 4-fold cross validation.
Support Vector Machine (SVM)

A SVM is implemented according to Chen, Su & Yang (2015). Again, in their paper, the CNN is trained using a different dataset which contains more real-life technique samples but the purpose of the classifier is the same as the CNN approach, which is to identify the technique based on, for instance, a 0.2 second audio excerpt.

3 sets of input features from the dataset are extracted – namely timbre, MFCC and pitch features.

The timbre features include the mean, standard deviation (SD), maximum, minimum, skewness and kurtosis of the following features: spectral centroid, brightness, spread, skewness, kurtosis, flux, roll-off, entropy, irregularity, roughness, inharmonicity, zero-crossing rate, low-energy ratio and their 1st-order time difference.
The MFCC features include mean and SD of the 40 MFCCs and its 1st-order time difference.

The pitch features include a sequence of log-scale floating-point numbers based on MIDI note numbers. The skewness, kurtosis, variance, min-max difference, mean and SD of the 1st-order time difference of the pitch sequences are also included.

These input features are fit into a multiclass support vector classifier with RBF kernel. The classifier is trained with 4-fold cross validation and the hyperparameters are tuned through the cross validations.

### Evaluation

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bend</td>
<td>0.605</td>
<td>0.804</td>
<td>0.690</td>
<td>219.0</td>
</tr>
<tr>
<td>Slide</td>
<td>0.435</td>
<td>0.665</td>
<td>0.526</td>
<td>233.0</td>
</tr>
<tr>
<td>Trill</td>
<td>0.750</td>
<td>0.021</td>
<td>0.040</td>
<td>145.0</td>
</tr>
<tr>
<td>Hamm</td>
<td>0.727</td>
<td>0.264</td>
<td>0.388</td>
<td>121.0</td>
</tr>
<tr>
<td>Pull</td>
<td>0.778</td>
<td>0.060</td>
<td>0.111</td>
<td>117.0</td>
</tr>
<tr>
<td>Mute</td>
<td>0.973</td>
<td>0.727</td>
<td>0.832</td>
<td>99.0</td>
</tr>
<tr>
<td>Normal</td>
<td>0.824</td>
<td>0.969</td>
<td>0.891</td>
<td>891.0</td>
</tr>
<tr>
<td>avg / total</td>
<td>0.741</td>
<td>0.717</td>
<td>0.666</td>
<td>1825.0</td>
</tr>
</tbody>
</table>

(Figure 5. CNN classification report (Hamm = Hammer-on, Pull = Pull-off))

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bend</td>
<td>0.939</td>
<td>0.966</td>
<td>0.952</td>
<td>238.0</td>
</tr>
<tr>
<td>Slide</td>
<td>0.872</td>
<td>0.812</td>
<td>0.841</td>
<td>160.0</td>
</tr>
<tr>
<td>Trill</td>
<td>0.970</td>
<td>0.970</td>
<td>0.970</td>
<td>66.0</td>
</tr>
<tr>
<td>Hamm</td>
<td>0.945</td>
<td>0.963</td>
<td>0.954</td>
<td>885.0</td>
</tr>
<tr>
<td>Pull</td>
<td>0.832</td>
<td>0.764</td>
<td>0.796</td>
<td>110.0</td>
</tr>
<tr>
<td>Mute</td>
<td>0.798</td>
<td>0.840</td>
<td>0.819</td>
<td>231.0</td>
</tr>
<tr>
<td>Normal</td>
<td>0.798</td>
<td>0.704</td>
<td>0.748</td>
<td>135.0</td>
</tr>
<tr>
<td>avg / total</td>
<td>0.902</td>
<td>0.904</td>
<td>0.902</td>
<td>1825.0</td>
</tr>
</tbody>
</table>

(Figure 6. SVM classification report)

We first compare the evaluation results as shown in the above classification reports. In our project, the most important metric in Figure 5 and 6 is recall rate, which tells of the relevant results, how many relevant samples are classified right. This essentially measures, assume a student is playing technique “A”, how good is the system in identifying she is actually playing technique “A”. The philosophy is that if a student is doing right, the system should be capable of telling her so. Precision score, which measures how well the system could classify a technique out of all possibilities, is relatively not as important. Therefore, we will focus on evaluating recall rate.

In summary, SVM attained a recall rate of 0.904 while CNN attained a recall rate of 0.717. In particular, SVM is good at recognizing Bend, Trill and Hamm techniques, in which it achieved 0.966, 0.970 and 0.963 recall rates in the 3 categories. It has a low recall rate of about 0.70 – 0.82 in recognizing Normal, Pull and Slide techniques. Normal is the best recognizable technique by the
CNN while on the other hand the CNN has surprisingly low recall rates of 0.021 and 0.060 in Trill and Pull.

![Normalized confusion matrix](image)

**Figure 7. CNN normalized confusion matrix**

![Normalized confusion matrix](image)

**Figure 8. SVM normalized confusion matrix**

As shown in the confusion matrices, SVM is the overall winner as all the diagonal entries have dark colours, while CNN has bad performance in recognizing Trill, Hamm and Pull techniques as seen from their pale colours. Particularly, Trill is easily confused with Normal, Hamm is easily confused with Bend and Slide, Pull is easily confused with Slide in the CNN results. Mute is confused with Pull, Normal is confused with Hamm and Slide is confused with Mute in the SVM results.

We conclude that SVM is a better candidate when performing classification based on this dataset. However, due to the dataset bug we reported above and the over-simplified convolutional models used, it is hard to abandon the use of CNN based on this evaluation. As a matter of fact, the main author of the TENT paper (Su, Chen, Su and Yang, under review) replied our team’s email saying that they have tried to train their CNN model using the dataset we used but the results are not optimal. Their guess is that the simple CNN models might not be adaptive enough to different timbre of guitar.
What will be done

CNN Classifier

In light of the promising evaluation results of SVM technique classifier, we will develop a technique classifier based on that and incorporate it into our playing evaluation mechanisms. In the meantime, we will have more investigation into the architecture of the CNN technique classifier to improve its accuracy. This could be done possibly with trying different combinations of convolutional and fully-connected layers, or even combining the use of fully-convolutional network or recurrent-based network as suggested by the paper itself (Su, Chen, Su and Yang, under review).

Back/ Frontend Architecture

We will implement the backend server evaluation mechanisms. As we have got the basic idea of what algorithms and classification approach works best, we will use CREPE to extract F0 sequence and SVM to classify techniques. The overall framework will be based on TENT. The first step will be to implement the note-by-note evaluation approach, and hopefully we will also implement the synchronized evaluation approach which also takes rhythm into account.

Quantitative System Metrics

We will also devise metrics for system evaluation. For instance, real-time response time of the evaluation in the frontend, accuracy of notes and techniques that the system can capture and other evaluation measures as suggested by the Music Information Retrieval community (Raffel et al., 2014).
References


