MUSICAPP, A MUSIC SHEET TRANSCRIBING MOBLIE PLATFORM USING MACHINE LEARNING AND NATURE LANGUAGE PROCESSING

Daniel Wang[1] and Yu Sun[2]

[1]Troy High School, 2200 Dorothy Ln, Fullerton, CA 92831
[2]California State Polytechnic University, Pomona,
CA, 91768, Irvine, CA 92620

ABSTRACT

As technology advances, we have found more practical uses for it. This ranges from such things as cleaning the house using machines to serving restaurants with robots. Using technology, what if we can use machines to automatically write sheet music for us, transcribing it from audio [1]. This paper designs an application to do exactly that. We used Java to write a program and app that would be able to transcribe audio into sheet music and store it on an app. We applied our application to multiple cases and conducted a qualitative evaluation of the approach. The results show that it is possible with some fine tuning and may be usable in the near future.

KEYWORDS

Music, Sheet Music, Audio to Sheet Music, Flutter.

1. INTRODUCTION

The program in mind is being made into an app. We believe the usage of it will be able to transcribe audio into sheet music that will be able to be saved in an app [2]. I think it will be popular with a certain niche but will be helpful in that certain niche. The benefits of the app will be the ability to easily make audio into sheet music. This will possibly benefit the people that want to write music using a computer and allow them to easily edit the music with different recordings and cut time by needing to manually write the music. The only consequence that I foresee happening is the possible mistranslation of music that can cause the sheet music to mess up and transcribe the wrong notes. I think this topic is very important as music is a massive part of culture and ways to improve the accessibility of music is extremely important.

Some of the existing techniques and systems that have been proposed to that were the same as mine are the numerous programs and apps. They are about the same as mine, being able to upload music using a WAV or MP3 file [3]. They also have the feature to record music and have it uploaded to the site. However these programs are all tied to the internet and computer which might make it difficult to record. Their programs might lead to some issues that are caused by bad recording methods because you have to record it on the computer. This is inferior to the computer as you would need your instrument near your computer to record the music, while the app being on a phone will allow you to record it wherever you want.

David C. Wyld et al. (Eds): SIP, CRBL, WiMNeT, BIOM, EDUPT, NLDM, SOFTFM, COMIT - 2022
pp. 87-95, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121709

My tool is an app that is able to record and transform WAV files into sheet music [4]. Our method was inspired by other tools but there are some other good features. First, our music is on the phone, allowing for easier recording and uploading. Secondly, our app allows you to record the music on the device, which allows for a quicker and easier way to get your sheet music as opposed to recording and uploading a WAV file every time. This provides an easier method than other methods because we allow our app on mobile devices and allow them to record them as well, allowing for easier recording.

We took 3 sample cases, Twinkle Little Star, Final Duet, and Electronic Music (Midi) and passed them through the app which converted them into sheet music and compared these sheet music with the actual sheet music from the songs to see how accurate our conversion was [5]. All three test cases are vastly different sounding. Twinkle Twinkle little star is a simple piano rift that everyone has heard. Final duet is a duet between a piano and a violin and electronic music uses digital synthesizers with a midi interface to create piano music.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the creation of the app and experiment; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the conclusion remarks, as well as pointing out the future work of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Converting Audio to a Midi File

The first challenge was turning audio into computer midi files. We need it to input it into our program so our program can actually read the audio data [6]. A midi file is a sheet music for the computer. A midi file pretty much holds musical information such as note numbers, pitch, duration, and other musical information for a piece. A midi file is organized by event messages that carry information for each note. Using a sequencer, we can view and edit the audio data as we would like. One way we can think of it is by using a computer to record a pitch and use a system to convert the pitch and test it to see if the computer recognizes it. There are points in the audio file where there are recorded key-frames that are called samples. These samples hold a pitch that has been recorded by our computer. Training a program to read in audio data becomes hard because in a single song, there could be a million samples.

2.2. Converting a Midi File into Sheet Music

While midi holds important note information pertaining to the particular piece you created, parsing through that data in attempts to make it into sheet music poses a problem. Firstly with the midi file format, it also comes with someone figuring out the notation, what is quarter note, ace note, half note. This happens because midi doesn't use the standardized format for time signatures. A time signature is the traditional way we signify how many beats are in each measure. A measure is a section of a song. It records it in a different way. It records audio duration in beats. The issue is that midi uses a different time measurement than normal. It uses ticks instead of beats and is recorded in delta time. It doesn't use a tick of normal time, it uses a tick of delta time, a measure of time from now to the last time a midi event happened. The main

challenge that comes with converting midi to sheet music is the conversion of time between midi standard time and regular time signature.

2.3. Connecting Our Powerful Musical Backend to a Comfortable Front End Available for Other Users

The third challenge is the creation of the app ui, we used flutter to create an android app using an android emulator [7]. Flutter is constantly used for creation of android and IOS app and doesn't supply much back end resources for audio or musical uses [8].We had to create the back end that wasn't linked with flutter and is on the google collab that is able to pull online python packages capable of overcoming the challenges needed [9]. Since flutter and google collab are not linked by any direct path or package, we had to create our own way to send audio information from the app into the converter and from the converter back into the app. One connection could be firebase but we needed another key to help post the image and communicate from flutter to the google collab server.

3. SOLUTION

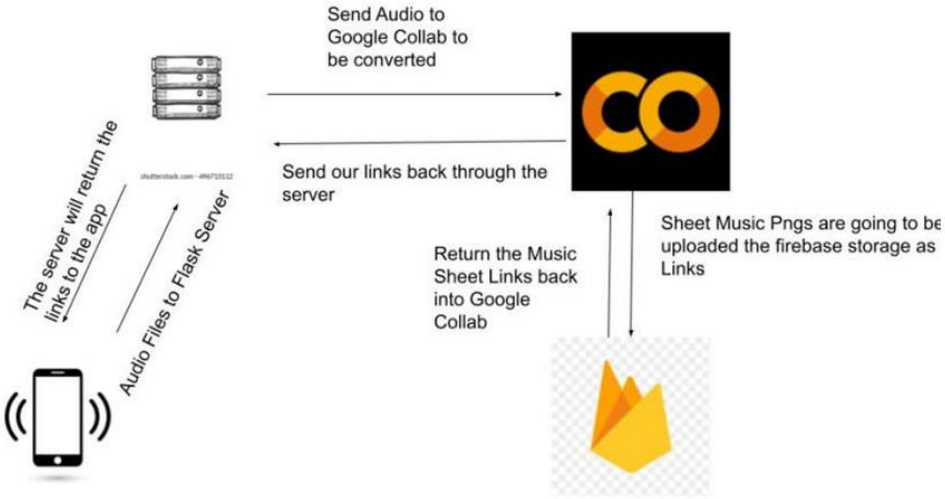
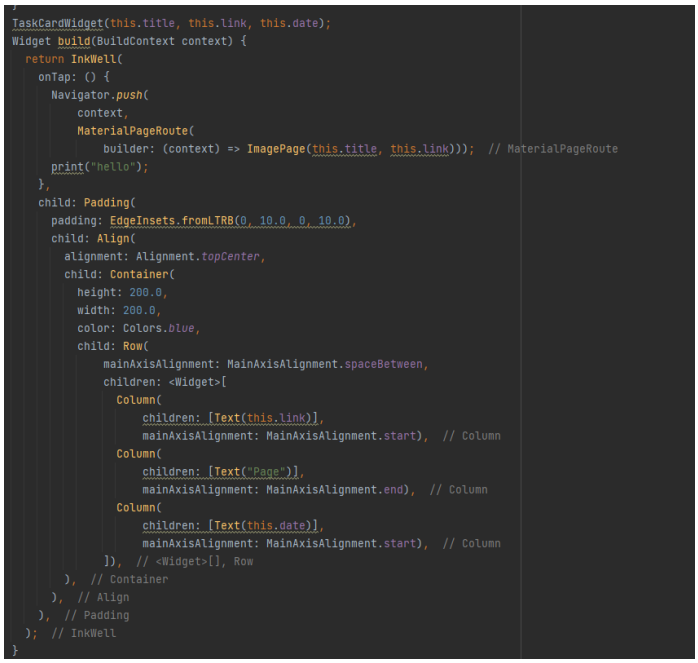


Figure 1. Overview of the solution

This is by running the audio file through the magenta package which uses Onsets and Frames to convert the audio to a midi file. Once we have a midi file our computer can start calculating the correct musical notation. For this we can use the music 21 library to convert the midi note number and tick format to western musical notation. Lastly, in the google collab, we use firebase service storage to upload our image to firebase [10]. We access the storage server service by using the firebase python package. In firebase, the image is hosted to the firebase server, meaning we have a web link to our image. That web link is sent back to the google collab which is then returned to the server and finally returned to our flutter app [11]. This image is displayed in the user on a new screen for our user using an image widget.

The first component we use to create our app is flutter. Flutter is a plug in for android studio code that easily allows people to make UIs for android and IOS applications. For this app, we made different screens, using dart files in flutter, each having its different use. We had a record page, a save notes page, and a sheet music viewing page. For our UI, we used the flutter built in widgets

and for app functionality such as connecting to the internet and sending files to the firebase. We used dart packages.



```
TaskCardWidget(this.title, this.link, this.date);
Widget build(BuildContext context) {
  return InkWell(
    onTap: () {
      Navigator.push(
          context,
          MaterialPageRoute(
              builder: (context) => ImagePage(this.title, this.link)));  // MaterialPageRoute
      print("hello");
    },
    child: Padding(
      padding: EdgeInsets.fromLTRB(0, 10.0, 0, 10.0),
      child: Align(
        alignment: Alignment.topCenter,
        child: Container(
          height: 200.0,
          width: 200.0,
          color: Colors.blue,
          child: Row(
              mainAxisAlignment: MainAxisAlignment.spaceBetween,
              children: <Widget>[
                Column(
                    children: [Text(this.link)],
                    mainAxisAlignment: MainAxisAlignment.start),  // Column
                Column(
                    children: [Text("Page")],
                    mainAxisAlignment: MainAxisAlignment.end),  // Column
                Column(
                    children: [Text(this.date)],
                    mainAxisAlignment: MainAxisAlignment.start),  // Column
              ]),  // <Widget>[], Row
        ),  // Container
      ),  // Align
    ),  // Padding
  );  // InkWell
}
```

Figure 2. Screenshot of code 1

(THIS IS UI CODE) This is the code to create a new custom task card widget. It stores all the info for our converted sheet music. We can see a couple of UI flutter mechanics in here such as container, row, and text.

```
void pickFiles() async {

  _resetState();
  try {
    _paths = (await FilePicker.platform.pickFiles(
      type: _pickingType,
      onFileLoading: (FilePickerStatus status) => print(status),
      allowedExtensions: ['mp3', 'wav'],
    ))
        ?.files;
    _uploadFiletoServer();
  } catch (e) {
    _logException(e.toString());
  }
}
```

Figure 3. Screenshot of code 2

This is code that examples dart packages and in this function, we use file picker package to open the native file picker on either android or IOS devices. We can see that we only allow mp3 and wav files into our program. After we pick a file, our flutter code calls a very important function called uploadFiletoServer, this allows us to send our files to the ngrok server. The ngrok server is the second major component in our app, it provides a tunnel to our web-page that holds our music files in its directories. We use the flask python package to capture the music files from our server into our google collab with python code.

Our third major collaboration is the ability to turn audio files into sheet music. This is by calling the function, generate sheet music. This takes in a file path that comes from flask server and first what generate sheet music does The first thing that generate sheet music does is convert our audio file to midi. We do this by using the function generatemidi, which is going to use the magenta package to convert a solo piano recording into a midi sequence. The way magenta processes this is by using a new model they created uses onset and frames. This model tracks notes across two stacks of programs. One stack is detecting when a note begins and if a note is currently active. Using this output, we can restrict some notes that are picked up by the detector that might not be there.

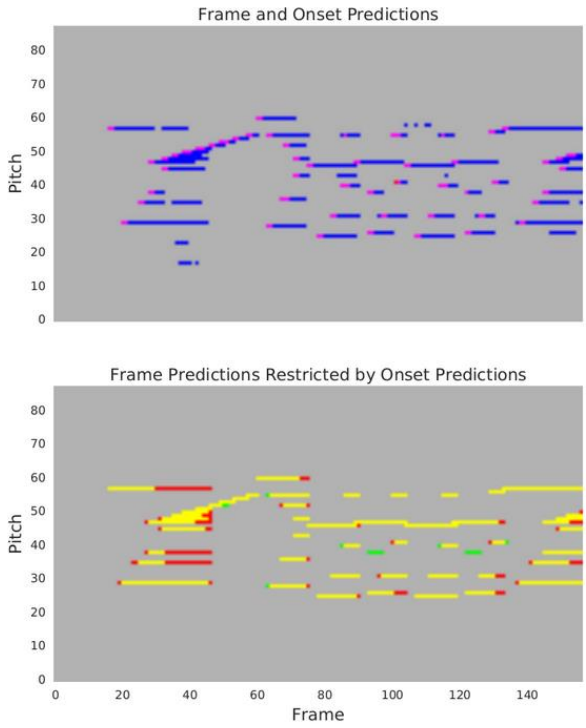


Figure 4. Graph of predictions

The first image in blue and magenta is the prediction of notes without restricting it by onset predictions every time a note is not actually playing, the magenta block is not at the front of the note. The image below shows the results of restricting our prediction by our onset prediction which in turns yields more accurate midi data. After we return the midi data to generate sheet music, we then pass it to the function midi2sheet_music This is going to take in a midi path and use the music 21 package to convert the midi information into a png image of the sheet music. The music 21 library was created in the university of MIT and is used as a tool kit for computer aided musicology. We can use the python library to make sheet music directly from python using the note class but for us we wanted to convert a whole midi sequence so we have to use the converter class which is a part of the music 21 library. This class is a midi converted into a music 21 object. Once it's in the music 21 library, we can write it to a png file which is returned to generate sheet music. Then generate sheet music and upload that png to firebase storage. Then it will take the image links and turn it back into ngrok server, then using the flutter which will be displayed using the image screen.

4. EXPERIMENT

4.1. Experiment 1

The first experiment that I had in mind was that we were going to take an audio file and convert it into sheet music using our app, then compare it against the written sheet music piece. To analyze this, we will be listening to both of the pieces played to see if there are any audio differences or if there is any notation difference. The first piece we are going to examine is a simple melody on a solo piano. This piece is going to be a twinkle twinkle little star. This is a base case because twinkle twinkle little star is a simple song that the AI should be able to recognize.

This is the result of the generated sheet music using our app, we can tell that this has visible different notation than our original notation. Although listening to both of the pieces side by side, we can see.

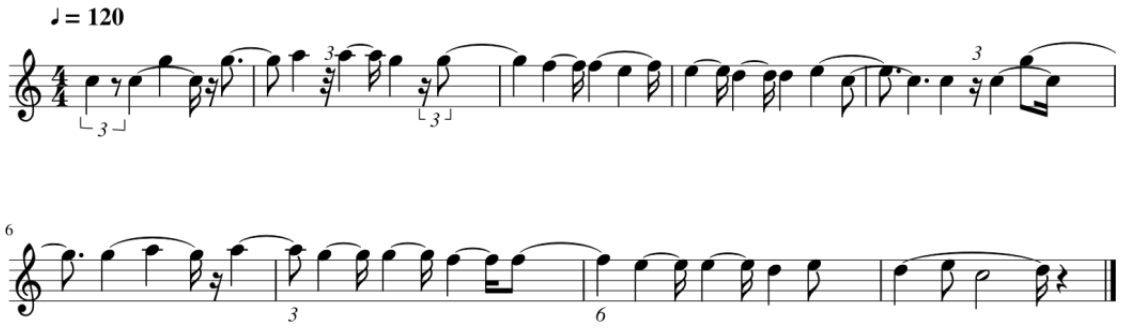


Figure 5. Music pieces 1

4.2. Experiment 2

This experiment is going to use our app to generate sheet music of the same twinkle twinkle little star melody. This melody is written in 4/4 time meter and one of the easiest melodies to play on piano. The variable that we are changing is going to piano timbre. Timbre is an umbrella term that usually stands for the sound of the instrument. For instance, a grand piano has a distinct sound, so does an electric guitar. We can use the computer to create synthetic timbres and use a simple midi sequencer, so we created an audio file that uses the same melody but instead of a grand piano playing, its a bright synth lead.

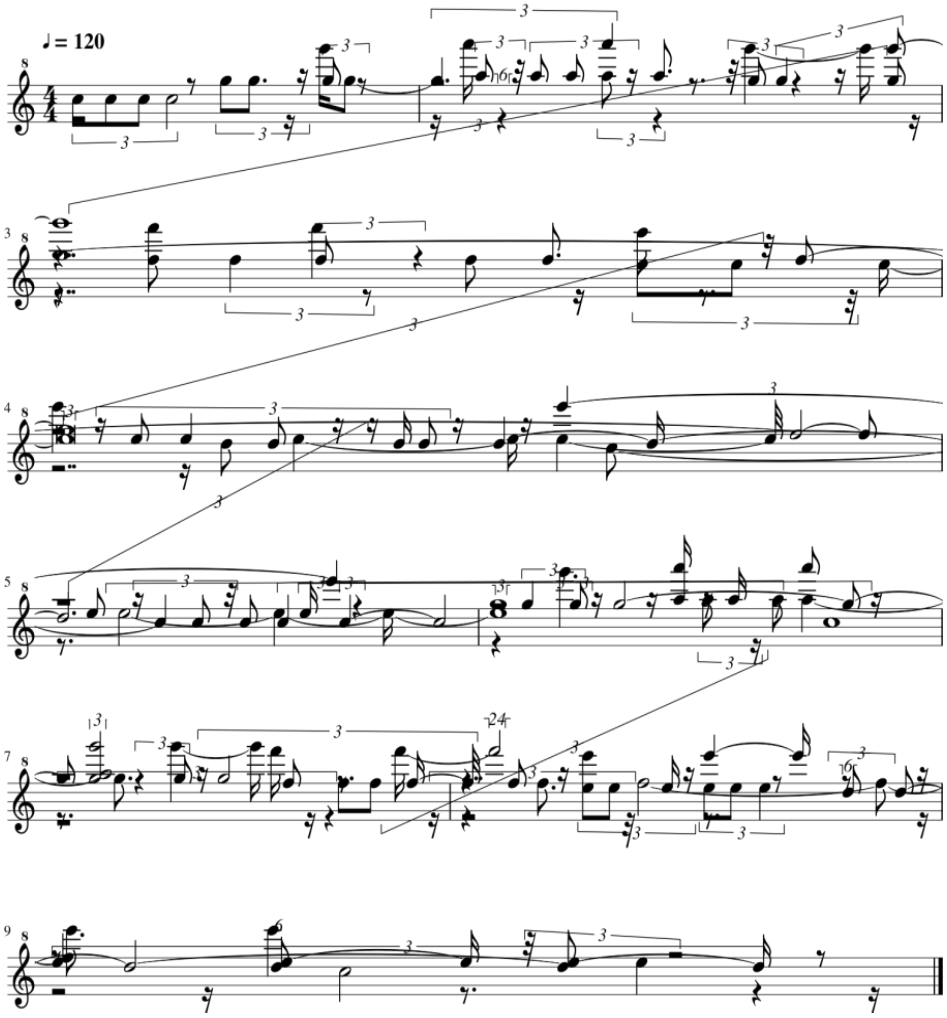


Figure 6. Music pieces 2

This is the generated sheet music from our app from the timbre shifted audio file. We can already see a similarity from both our results that seems to overcomplicate the notation of the sheet music. We can also see there are more lines there as more measures have more notes in it than the last result. Playing this sheet music alongside the bright synth lead.

Analyzing the results we saw from our experiments, we can determine that challenge one was successfully met. We were able to convert audio into a midi file. We were able to see this by checking the output and comparing the midi file to the original midi file of twinkle twinkle little stars. In both cases, timbre is shifted, the audio was actually able to recognize the midi information. Converting that midi to sheet music also worked but didn't meet expectations [15]. There are a couple issues converting a full midi file into music 21 library. One is that it overcomplicates the notation and also that there is no bass clef included. Although for simple piano tracks such as twinkle twinkle little star this played in solo piano. We were able to receive sheet music that actually sounds like the original sheet music so overall, this app could be used as a sort of practice for beginners that are unaware how to notate their piano melodies.

5. RELATED WORK

In this paper, the four scientists Fremery, Clauson, Ewert, and Muller researched two different approaches to automate a sheet music to audio identification [12]. Their pieces used various instruments with different complexities. The two approaches they used are first in OMR software, which is an optical music software that extracts information from the musical symbols as the program scans the music sheet. The second approach is a music to audio matching search, which compares musical content to a sequence of bars that we're currently looking at. In the end, the scientists proved that both approaches seem to be more useful in certain situations than others. This research paper took a much larger experiment pool of various tracts, pages of music, with varying different complexities. Our work focused more on mainly simple melodies. In our work, the strength is that we actually showed the results of our generated sheet music, but since they're experimental pool was so large, they were unable to show the sheet music in the research paper.

6. CONCLUSIONS

In this paper, we discover more automated use of machine learning to generate the music sheet base [13]. And we used advanced dev technology in dart based mobile dev framework names flutter. The way that flutter is connected to the server to send audio files is to use the HTTP package. We use the HTTP package for the flutter app to upload our audio file into the flask server. We use ngrok to host our server which is currently running on our google collab. After we upload our audio file from our app to our flask server, our google collab takes the audio files from the ngrok server. Then we use python code to take this audio file and convert it into an image of sheet music. The application is designed to perform all the functions mentioned before. We made this music app and published it on Google Play and did a lot of analysis with the app. Next step we are ready to publish the App to the Ios platform [14].

REFERENCES

[1] Saling, Lauren L., and James G. Phillips. "Automatic behaviour: efficient not mindless." Brain research bulletin 73.1-3 (2007): 1-20.
[2] Dorfer, Matthias, et al. "Learning audio–sheet music correspondences for cross-modal retrieval and piece identification." Transactions of the International Society for Music Information Retrieval 1.1 (2018).
[3] Liu, Qingzhong, Andrew H. Sung, and Mengyu Qiao. "Detecting information-hiding in WAV audios." 2008 19th International Conference on Pattern Recognition. IEEE, 2008.
[4] Islam, Rashedul, Rofiqul Islam, and Tohidul Mazumder. "Mobile application and its global impact." International Journal of Engineering & Technology 10.6 (2010): 72-78.
[5] Loy, Gareth. "Musicians make a standard: the MIDI phenomenon." Computer Music Journal 9.4 (1985): 8-26.
[6] Li, Dongge, et al. "Classification of general audio data for content-based retrieval." Pattern recognition letters 22.5 (2001): 533-544.
[7] Chen, Qiuyuan, et al. "How should I improve the UI of my app? a study of user reviews of popular apps in the Google Play." ACM Transactions on Software Engineering and Methodology (TOSEM) 30.3 (2021): 1-38.
[8] Garg, Shivi, and Niyati Baliyan. "Comparative analysis of Android and iOS from security viewpoint." Computer Science Review 40 (2021): 100372.
[9] Gunawan, Teddy Surya, et al. "Development of video-based emotion recognition using deep learning with Google Colab." TELKOMNIKA (Telecommunication Computing Electronics and Control) 18.5 (2020): 2463-2471.
[10] Khawas, Chunnu, and Pritam Shah. "Application of firebase in android app development-a study." International Journal of Computer Applications 179.46 (2018): 49-53.

[11] Payne, Rap. "Developing in Flutter." Beginning App Development with Flutter. Apress, Berkeley, CA, 2019. 9-27.
[12] Ewert, Sebastian. Signal Processing Methods for Music Synchronization, Audio Matching, and Source Separation. Diss. Universitäts-und Landesbibliothek Bonn, 2012.
[13] Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." Science 349.6245 (2015): 255-260.
[14] Wetchakorn, Thara, and Nakornthip Prompoon. "Method for mobile user interface design patterns creation for iOS platform." 2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE). IEEE, 2015.
[15] Tanprasert, Thitaree, et al. "Midi-sheet music alignment using bootleg score synthesis." arXiv preprint arXiv:2004.10345 (2020).