Marcos Madrigal Albores - 1004731347

Nancy Li - 1004877868

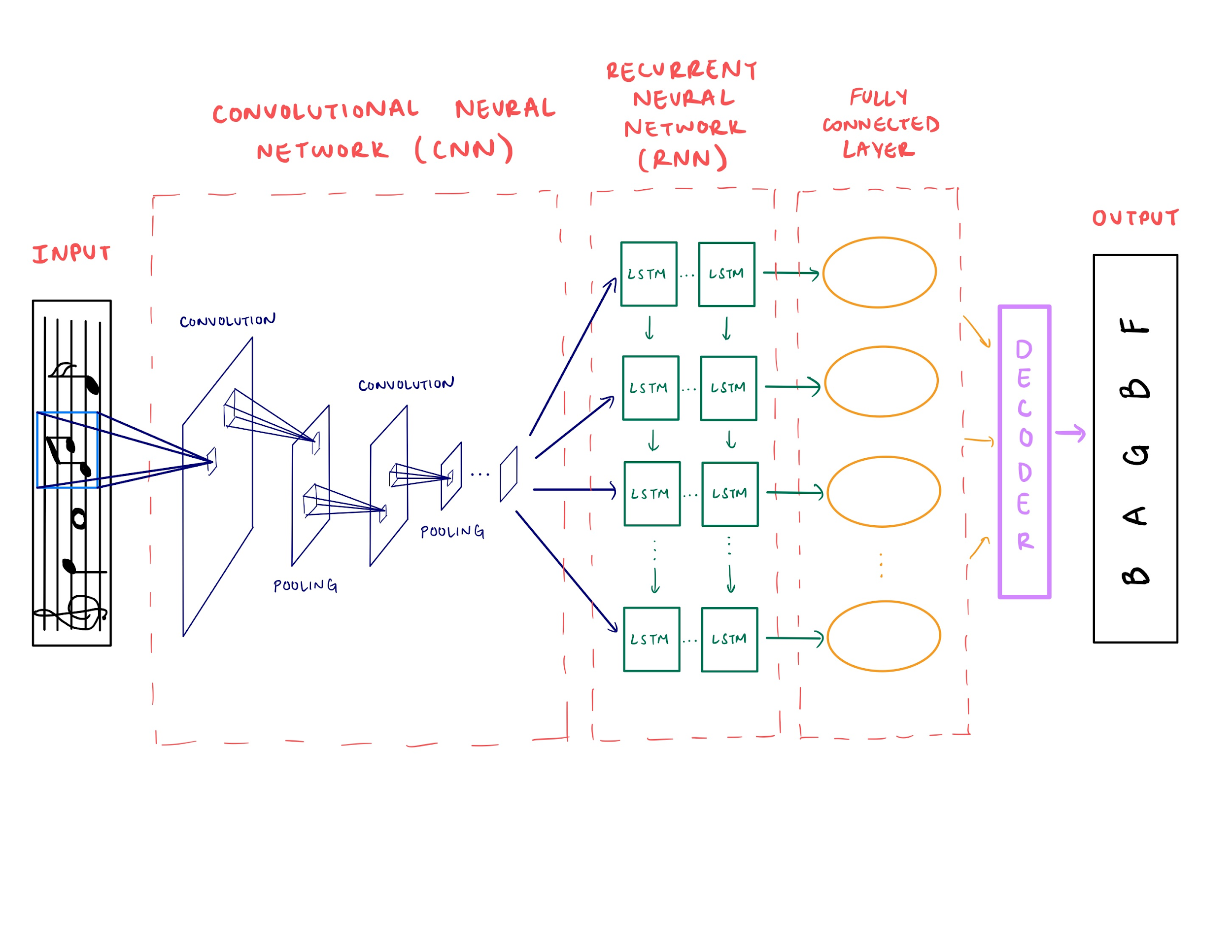
Mirza Nahiyan - 1005188075

Fourth team member is dropping this course[[1]](#footnote-0)

**1. Introduction**

Have you ever wanted to play the piano but didn’t have the time to learn to read musical notes? Reading sheet music can be a major learning obstacle for beginners. We want to create a music sheet transcriber that makes sheet music easier to understand for those who cannot sight read but simply want to play their favourite songs. A successful implementation of our model could also help to solve related problems of digitizing sheet music records, or evaluating music theory homework for students. We aim to use Machine Learning classification algorithms to create a model that takes an image of sheet music and outputs the corresponding notes as alphabets. Machine Learning is an ideal tool for this project because it is a subset of the image classification problem which is a solved problem that has an efficient Machine Learning solution. Hardcoding note recognition is a near impossible task due to the myriad of different styles and formatting for the music sheets and notes.

**2. Illustration / Figure**

****

**3. Background & Related Work**

Our project lies in the field of Optical Music Recognition (OMR) which aims to computationally read music notation from images. A study published in 2012 by Rebelo et al. [1] describes traditionally used algorithms which are used in a multi-staged approach involving image preprocessing, symbol classification and musical notation reconstruction. Common preliminary steps include staff line removal, locating and isolating individual musical symbols, and finally interpreting the musical semantics from the graphical objects detected as well as their positional information. This approach uses complex code, making it difficult to generalize the models to any piece of sheet music.

A model proposed in a paper by Calvo-Zaragoza et al. tries to solve the OMR problem for printed monophonic (one melody) scores with a holistic end-to-end approach using supervised learning and a Convolutional Recurrent Neural Network (CRNN) [2]. The model takes in a PNG image of an incipit (first few notes in a piece of music) as input to a Convolutional Neural Network (CNN) for image processing and feature extraction. The output channels from the last layer of the CNN are concatenated into a single image and used as input for a Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) units. The RNN then analyzes columns of the image to produce the predicted sequence of musical notes as output. The model achieved a sequence error rate of 12.5%, where sequence error refers to the ratio of predicted sequences that had at least one error.

**4. Data Processing**

For our project, we decided to use the Printed Images of Music Staves (PrIMuS) dataset [3], comprising approximately 80,000 incipits. Each piece of data has 5 representations but out of those, a png file represents the actual image and a semantic encoding represents the semantic labels, both of varying size across the set. It is also worth noting that even though the images are portrayed in black and white, the images have 3 channels.

With such a complex dataset (at least by the standards of those seen and used in the course) a simplification was in order, thus our goal for the data processing part of this project was to create a dataset that was compatible with the data loader implementation given by Pytorch, because that would provide us with a familiar method for using the data that was widely used in labs throughout this course.

In order to use this dataset and accomplish our previously stated goal, we had to standardize the data to a common maximum image size, which is 154x1200 pixels and do custom modifications depending on the model we were currently testing. One such modification was the need for a standard max label size, which had to be implemented in order to train the 3 layer CNN baseline model; in order to appeal to this model we had to pad both the labels and data accordingly. We used integer encoding due to the high number of possible notes (values) our multi target labels could take.

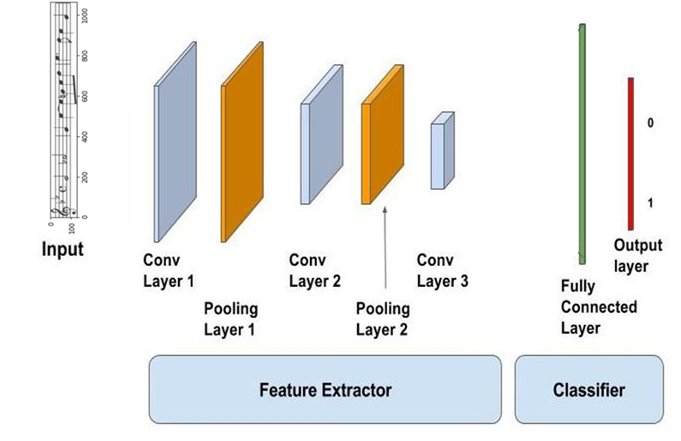
In the end we obtained 3 datasets, one small dataset for overfitting purposes and two larger ones for training which were used with an 80-10-10 (train-validation-test) split.

**5. Architecture**

To solve our OMR problem, we designed a CRNN. The CNN is used for feature extraction and its output is fed into a RNN that handles the sequential aspect of the problem. The output from the RNN is then passed to a fully connected layer that will make a prediction on each column of its input. The CNN consists of 4 convolutional layers with 3x3 kernels and 3 max pooling layers between them with 2x2 kernels and a stride of 2. The first convolutional layer has 5 output channels and every subsequent one has double the last. There is also batch normalization and a ReLU activation function applied after each convolutional layer. For the RNN, we used 3 bidirectional LSTM layers with 128 hidden units each that allowed the model to better predict the output since it has a more thorough understanding of a note’s context using information from both the past and future. A logarithmic softmax is then applied to the output of the model to get the probabilities required to decode the final prediction. Training was done using the Connectionist Temporal Classification (CTC) loss function and an Adam optimizer.

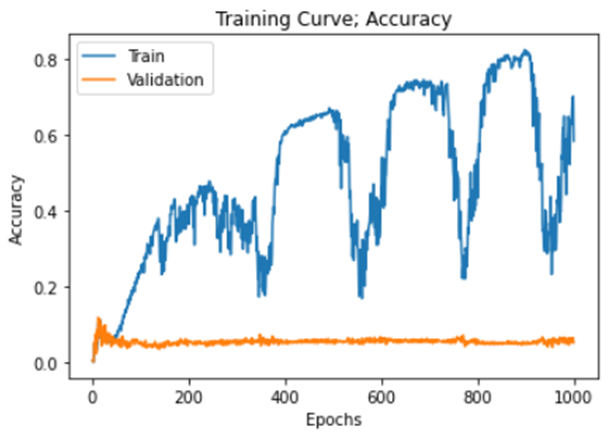
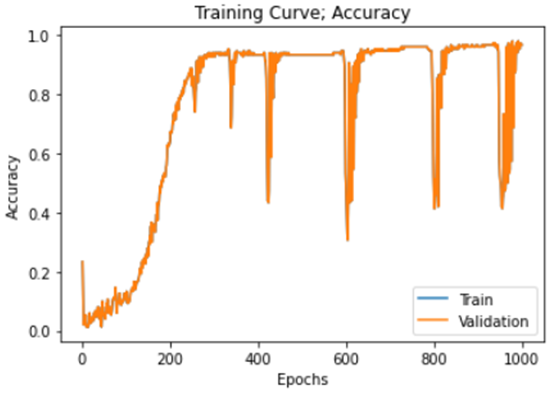
**6. Baseline Model**

Given our extensive use of CNNs for image classification in this course, we decided to have our baseline model be a simple Convolutional Neural Network. This CNN has 3 convolutional layers, 2 max pooling layers after each of the first 2 convolutional layers and 2 fully connected layers at the end; we used learning rate of 0.001, a kernel size of 5 across all layers and an MSE loss due to the multi target nature of our project.



*Baseline CNN model*

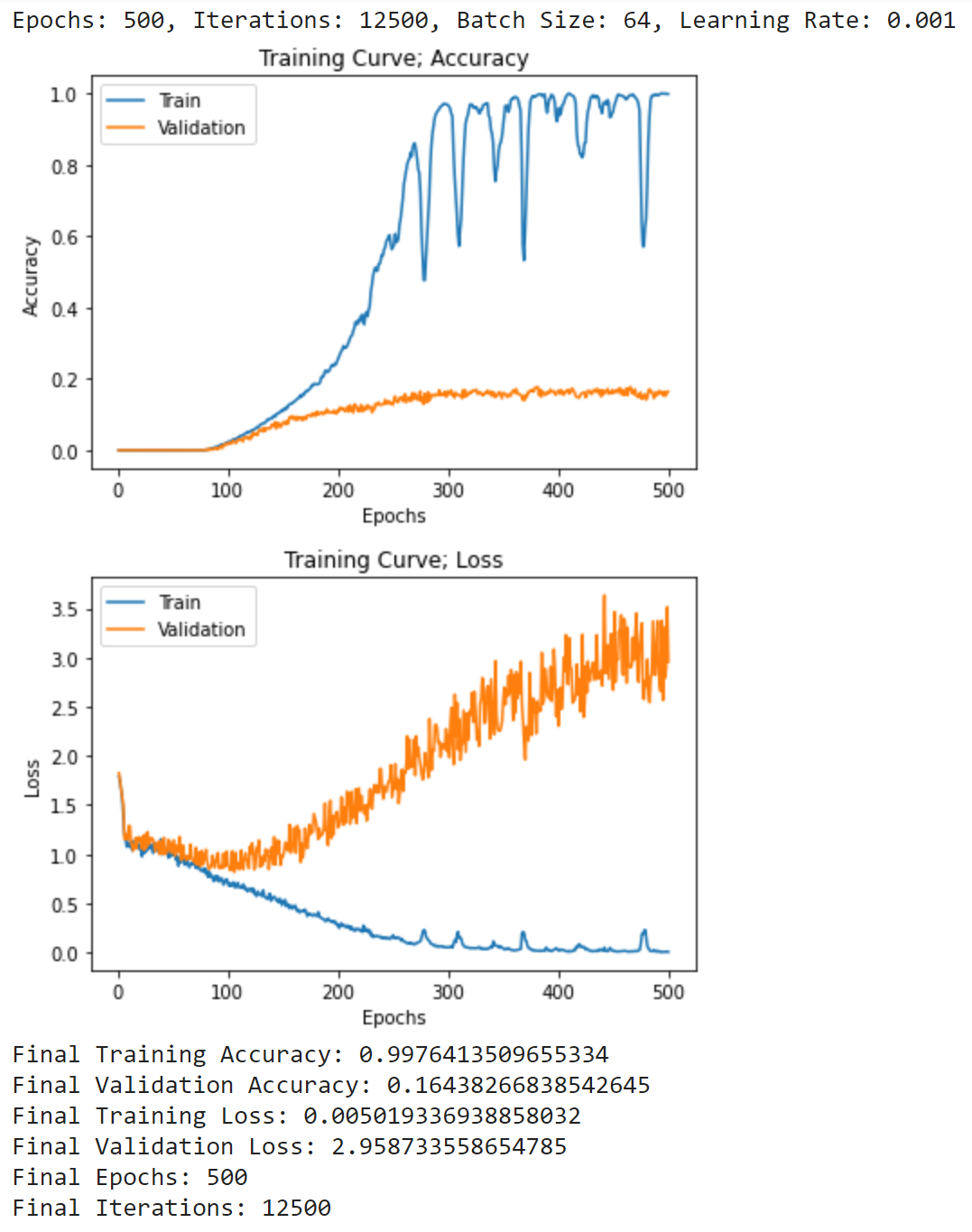
The model overfitted (100% training accuracy) over the smallest (overfitting) dataset of 10 samples over 500 epochs and was able to achieve a remarkable (due to the complexity of the problem) 80% max training accuracy and 5 to 7% validation accuracy over 1000 epochs, with a stable 5% testing accuracy. This only highlights the importance and need of a more complex architecture to solve the difficult problem stated in this project, as the one in our main model.

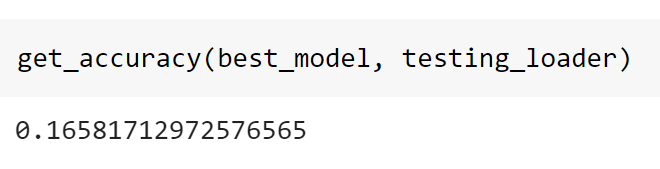


*Baseline Overfitting Baseline Training over 1000 epochs*

**7. Quantitative Results**

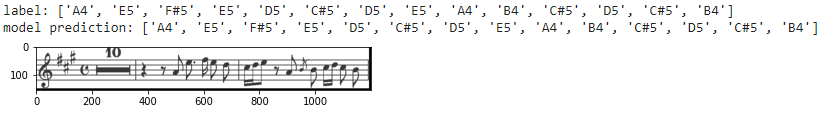
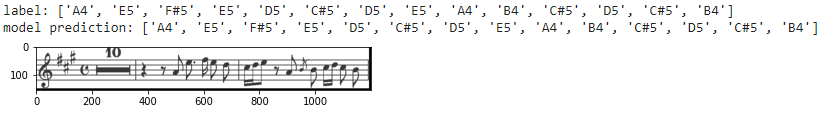
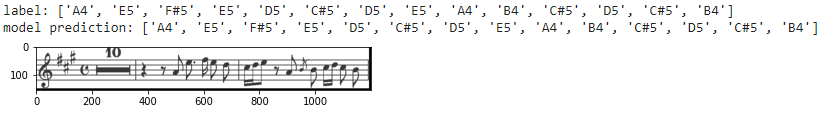
To illustrate how our model performs, we chose to compute accuracy as the number of notes predicted correctly within each label, or in other words the percentage of correct notes in the right places. Using this method and the Connectionist Temporal Classification (CTC) loss function to compute loss, we trained our model on ~1600 images and obtained the training curves below. Our best model achieved around 99% training accuracy and 17% validation accuracy. After using this model and testing it on the testset of withheld data, we achieved a testing accuracy of around 16%.

****

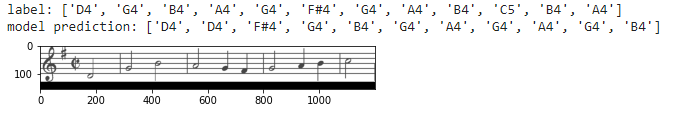
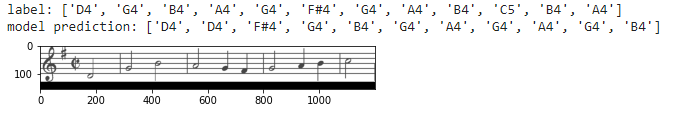
****

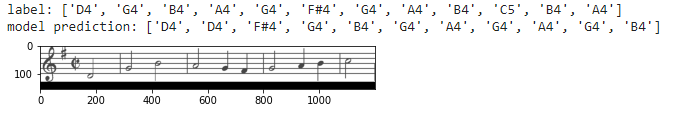
**8. Qualitative Results**

As a qualitative result of our model, the figure below shows one sample output given an image from the training set. Our model yielded 100% accuracy on this image, as it predicted every note correctly. This sample output shows that our model was able to learn from the training set well, predicting every note correctly, with matching lengths to the original label.



The figure below shows one sample output of our model evaluated on the validation set. Our model predicted two notes, or 16.7% of notes in the label correctly, which is approximately equal to the model’s final validation accuracy. The figure shows that the model predicted the first and second last notes correctly, but misclassified the notes in between. The model also failed to identify the last note, and produced a prediction with length shorter than that of the label. Most output of the validation dataset followed similar patterns. The model therefore is more likely to be able to correctly classify individual notes at the beginning and end of images. Contrarily, the model is unable to correctly classify notes consecutively, and sometimes misses a note, thus producing a shorter output.

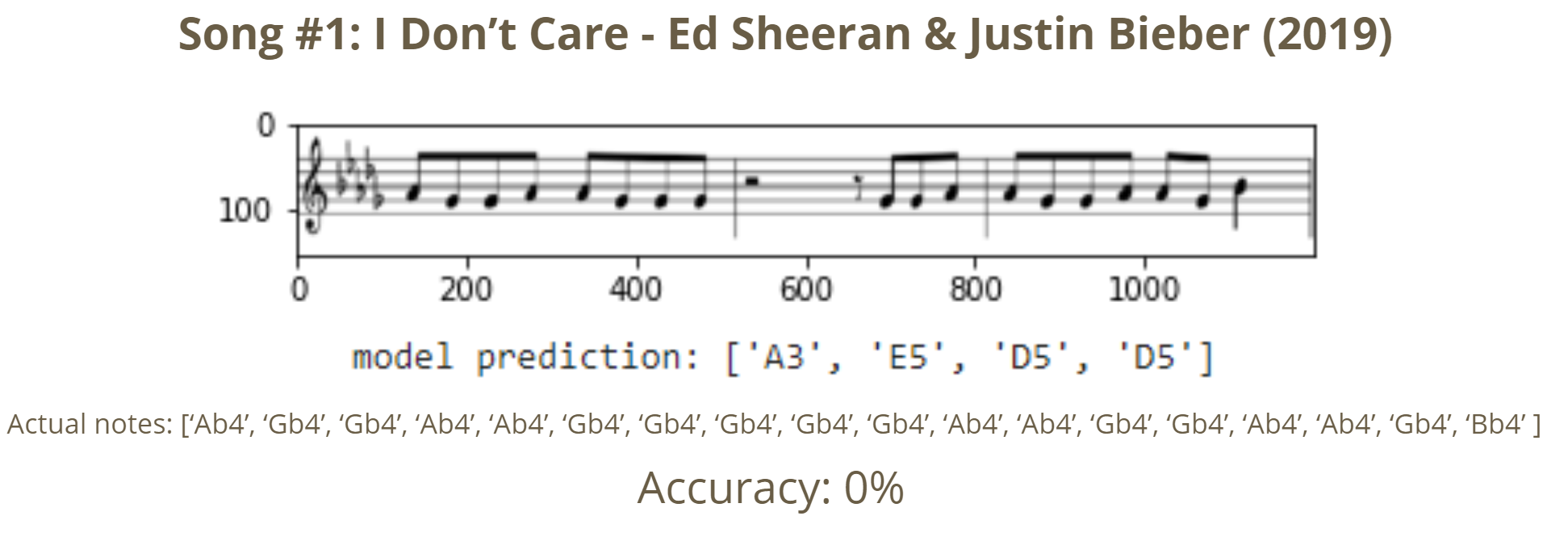
****

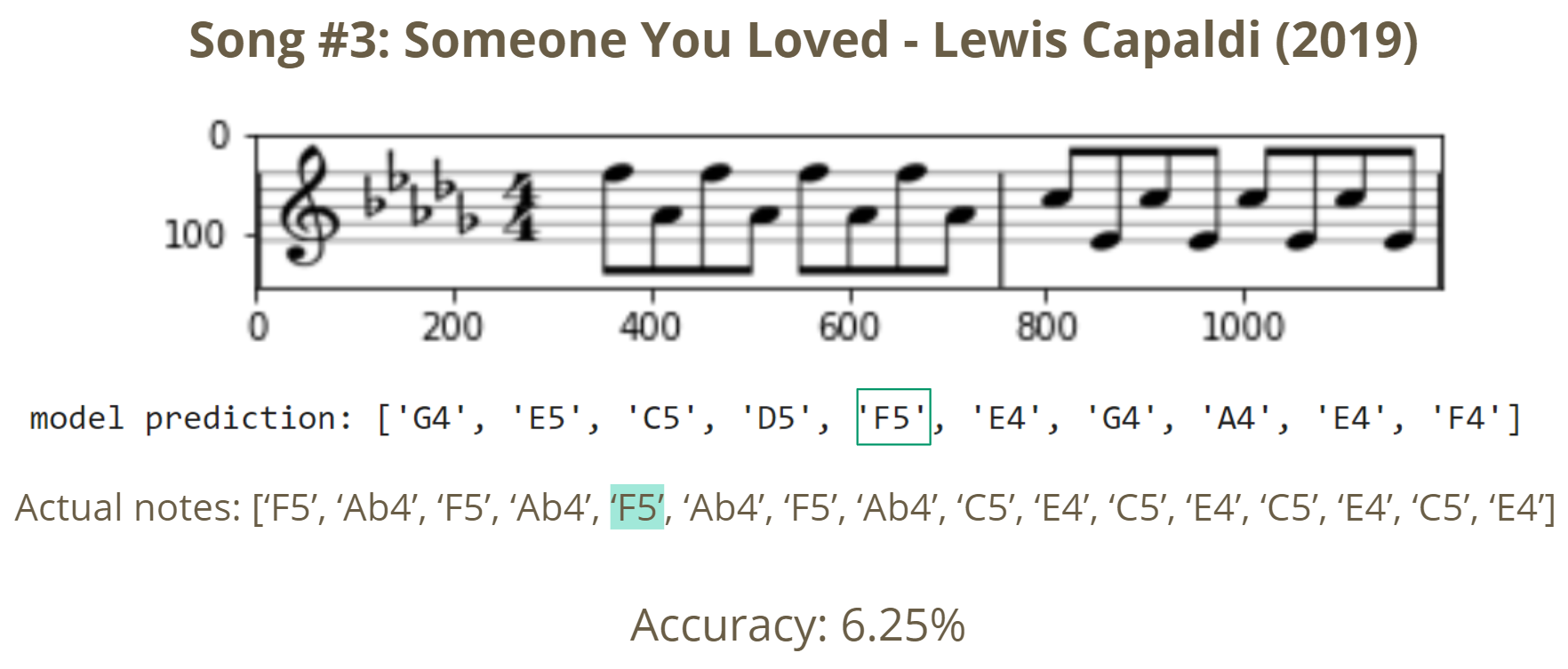
****

Overall, the model is able to correctly classify complicated sheet music from the training data, but achieves low accuracy on a simple validation data. These results indicate that our model is overfitted and has difficulty generalizing which is why it has poor performance on validation data.

**9. Evaluate Model on New Data**

To evaluate our model on completely new data, we chose to use sheet music found on Google for songs released after the PrIMuS dataset was created in 2018. The songs we chose were I Don’t Care by Ed Sheeran & Justin Bieber and Someone You Loved by Lewis Capaldi which were results from searching for the top songs of 2019. To preprocess the data, we screenshotted part of a staff for each song and resized them to be 154x1200 pixels. We then passed each of these images to our model and got the results shown below. After comparing them to the actual notes by eye, our model obtains an accuracy of 0% on the first song and 6.25% on the second song.





**10. Discussion**

To understand the context of our results, we must first consider the data itself and the problem we were tackling. The PrIMuS dataset was specifically designed for a research paper that used a complex model to achieve a sub 90% accuracy on the validation set, trained over long hours on powerful GPUs [2]. Our model was expected to underperform this reference model from the beginning due to the unfamiliarity of the multitarget sequence labeling nature of the problem and the hardware limitations we had. However, these complications made this a project worth investing our time into due to the potential experience and knowledge to be gained.

The dataset taught us the importance of processing your data, for it can make or break your project. How you handle your data can highly simplify your problem and greatly improve performance, but it also depends on the model you are working on, as you may need differently processed data for different models (baseline, main, etc.) . This also means that if you know how to handle your data and load it properly according to your available resources you can bypass hardware limitations, this we learned the hard way.

Due to the complexity of the project the baseline had to be a CNN itself. The original plan was to have a CRNN but since it was also our main model, they would only differ by hyperparameter tuning. Working with this baseline model made us realize the importance of the sequence handling part of the CRNN for this particular project, the reported accuracies had a great performance gap; this proved that in order to solve this OMR project we needed both feature extraction and sequence detection even though the data was presented in the form of images. From this we can draw the conclusion that mixed architectures allow us to better solve specific and complex problems that couldn't be solved by means of its parts.

As a discussion closure, our model performed below expectations but still performed considerably well for our available resources, time and knowledge. The field of OMR is one that needs attention going into the future as the world of music expands beyond the capability of human hands to keep up. We need the help of Machine Learning to allow for better accessibility and storage of music sheet data.

**11. Ethical Considerations**

Under Canadian Law, a musical composition is protected by copyright if its author is still living or died less than 50 years ago. Using our model to redistribute the transcribed music online may violate this law. If one obtained the creator’s permissions and returned a portion of royalties [4], one could legally use our model to transcribe a copyrighted music sheet and distribute it online for a fee or profit through advertising on the digital platform. Therefore, it may be important to limit our model to personal use or only transcribe unprotected music sheets. In terms of training the model, it is legally acceptable to use copyrighted music to train a machine learning model [5]. However, there are ethical concerns around obtaining copyrighted music sheets through illegal downloads. Therefore, we should collect our training data through legal means, or use sheet music which creators have made freely available.

**12. Project Difficulty / Quality**

This project is a subset of Optical Music Recognition problem, an important research field that investigates music score recognition systems [6]. Years of research in the field has demonstrated that in order to analyze a music score, an image analyzing component and a sequence identifying component both need to be present in the model [2]. This project was difficult for us because of four main reasons.

First, music notation consists of many symbols and representations, such as notes of varying lengths, accidentals, and rests. This made data preprocessing difficult, as we had to come up with a way to uniquely encode symbols contained in the labels so that the model is able to predict and produce similar outputs.

Second, a complex model was needed to detect and classify various sequences of notes and symbols. Both CNN and RNN layers needed to be more sophisticated and complex than the models we have used in class before such as those that classify hand gestures or spam messages. This added complexity made it difficult to come up with a suitable model architecture. The training time was also increased as there were many parameters and model hyperparameters to train and tune.

Third, the model was a multi-output classifier: for one input image, it needed to output a sequence of predictions represented in that image. As the model may sometimes fail to identify notes and produce an output that does not match the size of the label, we could not use the loss and accuracy functions of general classification problems. For this reason, we had to come up with new approaches to compute the loss and accuracies that we have never learned in class.

Lastly, in the implementation and training phase, we tried both custom CRNN models and transfer learning techniques. Both approaches encountered the hardware limitations that prevented us from building and training a more complex CNN model suitable for our dataset. Due to these challenges, it was very difficult for us to come up with a high performing model that transcribes music sheets.

**13. References**

[1] A. Rebelo, I. Fujinaga, F. Paszkiewicz. *et al.* “Optical music recognition: state-of-the-art and open issues” *Int J MultiMed Info Retr,* 1, 2012. [Online]. Available: <https://doi.org/10.1007/s13735-012-0004-6>. [Accessed: 12-Feb-2021]

[2] J. Calvo-Zaragoza and D. Rizo, “End-to-End Neural Optical Music Recognition of Monophonic Scores,” *Applied Sciences*, vol. 8, no. 4, p. 606, Apr. 2018 [Online]. Available: <http://dx.doi.org/10.3390/app8040606>. [Accessed: 12-Feb-2021].

[3] *PrIMuS dataset*, 2018 [Online] Available at: <https://grfia.dlsi.ua.es/primus/> [Accessed 12 February 2021].

[4] Government of Canada, “What is copyright?,” *Canadian Intellectual Property Office*, 07-Sep-2016. [Online]. Available: <https://www.ic.gc.ca/eic/site/cipointernet-internetopic.nsf/eng/wr03719.html>. [Accessed: 12-Feb-2021].

[5] M. Stewart, “The Most Important Supreme Court Decision For Data Science and Machine Learning,” *Medium*, 29-Jul-2020. [Online]. Available: <https://towardsdatascience.com/the-most-important-supreme-court-decision-for-data-science-and-machine-learning-44cfc1c1bcaf>. [Accessed: 12-Feb-2021].

[6] Jiří Novotný and Jaroslav Pokorný, “Introduction to Optical Music Recognition: Overview and Practical Challenge”. [Online]. Available: [paper6.pdf (ceur-ws.org)](http://ceur-ws.org/Vol-1343/paper6.pdf). [Accessed: 9-April-2021].

1. Our fourth team member has planned on dropping this course and thus did not directly contribute to this report and contributed less to the project overall. [↑](#footnote-ref-0)