Attention Based Mechanisms for Multi-Instrumental Automatic Music Transcription

Jacques Chen
The University of British Columbia
2366 Main Mall 201, Vancouver, BC V6T 1Z4
jacquesc@student.ubc.ca

Aritro Roy Arko
The University of British Columbia
2366 Main Mall 201, Vancouver, BC V6T 1Z4
arko30@cs.ubc.ca

James Tang
The University of British Columbia
2366 Main Mall 201, Vancouver, BC V6T 1Z4
tangytob@student.ubc.ca

Abstract

In this report, we will show deep learning methods for automatic music transcription (AMT). We will demonstrate the use of attention mechanisms on audio music data and compare various architectures on large datasets. We will also discuss different representations of audio data such as mel spectrograms, and combined frequency and periodicity (CFPs) representation. In the end, we trained a novel model with self-attention mechanisms that performed similarly to the current state-of-the-art model in both a piano transcription dataset and a multi-instrument transcription dataset.

1. Introduction

Automatic Music Transcription (AMT) is a task which involves converting audio data into music notation. The notation can be in the form of sheet music, piano rolls, guitar tablature charts and Musical Instrument Digital Interface (MIDI) files. AMT is very important for a variety of reasons. This includes Music Information Retrieval (MIR), generation of music, music education and so on [1].

However, the conversion is difficult due to uncertainty caused by complicated multi instrumental music and noisy lower fidelity recordings. There are also many qualities in music that we could measure such as rhythm (which is the pulse of the music), pitch (which is the frequency of the note), dynamics (which is the volume and the change in volume) and timbre (the perceived sound quality of a music note). These qualities can be hard to measure, especially since human musicians are often imprecise. For example, flutes often sound sharp (which is slightly higher than their normal pitch) or flat (which is slightly lower than their normal pitch) based on the changing temperature of the room. Rhythm is also often more felt than precisely notated and timing can be inconsistent.

Most of the existing works tend to focus on single instruments [1], [3], [14], while most music in general uses multiple instruments. Neural networks have been applied for piano transcription. Recurrent neural networks (RNNs), convolutional neural networks (CNNs) and fully connected neural networks have been used to learn regressions for audio datasets with labeled inputs [2], [11], [6], [5]. The current state of the art (SOTA) model for piano transcription [7] calculates onsets and offsets times (the start and end times) of note and pedal events with very good precision. However, there has been very little work on multiple instrument music transcription. Among the few works, [15] discusses the multi-instrument AMT method, with signal processing techniques specifying pitch saliency, novel deep learning techniques, and concepts partly inspired by multi-object recognition, instance segmentation, and image-to-image translation.

We base our approach on the previous SOTA piano transcription model in [7]. However, it has some key extensions. We tried to add improvements introduced by multi-instrument models to this architecture. Most music transcription models do not take the raw audio as input directly. For example, [7] converts the raw audio into mel-spectrogram (a frequency representation) before feeding it to the model. Instead of using mel-spectrogram, we, like other multi-instrument music transcription systems [15] [14], converted the raw audio waveform into Combined Frequency and Periodicity (CFP). However, for comparison, we also used mel-spectrogram instead of CFP and fed to

the model. In addition, we added self-attention mechanisms to the model which was proven effective in [15] to capture longer distance dependencies in frequency data, especially when working with multiple instruments.

To summarise, our models' base architecture follow the architecture in [7]. However, we have changed certain aspects of the base model to compare how the performance varied. The main contributions of the research are threefold:
1) Instead of using a mel-spectrogram, we processed the raw audio waveform and attempted to train using CFP.
2) We added a self-attention mechanism to the model.
3) We tested the baseline model from [7] and our modified model with a multi-instrumental dataset.

2. Related Works

As discussed briefly in the previous section, there are lots of work on the piano transcription system. However, less works have been done on multi-instrument music transcription. In this section, we briefly describe the previous works that are relevant to our work.

[7] is currently the SOTA model for piano transcription that outperformed Google's onsets and frames based system [3]. In systems prior to [7], audio recordings were split into audio frames using discriminative models. It enabled to predict presence or absence of onsets and offsets in each frame. However, this restricted the resolution of the transcription to frame hop size. In addition, misalignment in onset and offset labels in the recordings made the detection of onset and offset times challenging. However, [7] proposes a system that transcribes both note and pedal events. Instead of predicting the presence of onsets and offsets, it predicts the continuous onsets and offset values of note and pedal events. It achieves excellent results on the MAESTRO dataset [4].

Among the few works on multi-instrument music transcription, [15] proposes a self-attention mechanism for multi-instrument music transcription. AMT research is broken down into certain levels: multi-pitch estimation (MPE) i.e frame level transcription on pitches; note tracking (NT) which is note level transcription on pitches, onsets and duration; multi-pitch stream (MPS) also known as stream level transcription on notes and streams. Multi-instrument music transcription is a kind of MPS. Each stream is an instrument. In this paper, they address the problem of identifying the instrument for every note in addition to proposing an improved, self attention based multi-instrument AMT model that achieves SOTA performance on MPS. Moreover, the proposed model is flexible to all the sub-tasks of multi-instrument AMT. This includes multi-instrument note tracking etc.

In our proposed solution, we include certain aspects of the models described above. In addition, our system includes a local self-attention method as described in [8]. Primarily, [8] focuses on creating a model architecture based on self-attention. The model is applied to a sequence modeling formulation of image generation with a likelihood that is easy to deal with. Where this local self-attention stands out is that, it restricts the self-attention method to attend to local neighborhoods which enables the model to deal with large sized input in spite of maintaining larger receptive fields per layer compared to normal CNNs.

3. Approach

For our approach, we will use the approach described in [7] and combine this with approach with CFP and attention approach based on [15]. The attention mechanism is based on the local attention mechanism described in [8]. We will break this section into multiple parts.

3.1. Dataset and Preprocessing

We will be performing experiments on 2 datasets, MAESTRO [4] and MusicNet [12]. The MAESTRO data set consists of 200 hours of piano performances in the form of wav files. The labels for this dataset are given as MIDI files and include information needed to reproduce the music digitally (onset, offset , volume, key pressure/velocity, pedal information etc.). MusicNet contains 34 hours of chamber music performances with multiple instruments in the form of wav files with labels that contain the notes and their onset and offsets, the instruments being played, and the note rhythm type. Since MusicNet does not contain velocity data, we will discard this label for our model, and only focus on the onset and offset labels. Likewise, MAESTRO only contains one type of instrument, so we will discard instrument labels in MusicNet. The piano transcription model described in [7] converts the data in MAESTRO into log mel spectrograms, which are then used as input to the model. A spectrogram is created by doing windowed FFT (fast Fourier transforms) on segments of time.

FFT converts signals from the time domain to the frequency domain. In the spectrogram in Figure 1, the time is given on the x axis, the frequency is on the y axis, and the amplitude is given as the color. Mel is a scale of pitches that are judged by listeners to be equal in distances from one another. We will experiment with integrating CFPs (Combined Frequency and Periodicity) inspired by the approach in [15]. CFPs are a representation like spectrograms but they also contain periodicity information. The reason for using this representation is that a pitch contains both frequency, periodicity, and harmonicity, which means that time domain representations are as important as frequency based representations [13]. The CFP representation can thus hopefully better capture elements of the sound in multi-instrumental music. Both representations have dimensions $T \times F \times C$, where T is the number of time frames, F is the

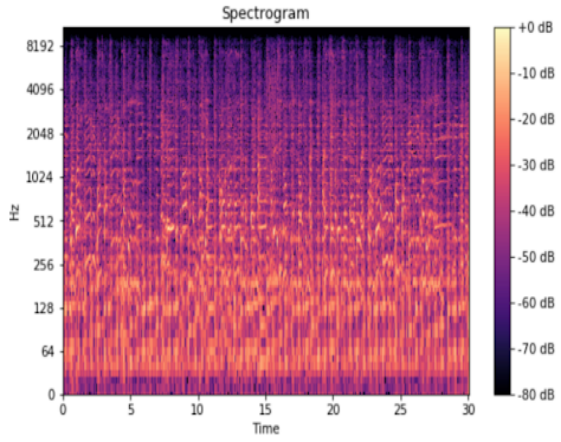

Figure 1. Mel-spectrogram, from [10]

number of frequency bins, and C is the number of channels (1 for Mel, 2 for CFP).

3.2. Loss

We adapt the loss used in [7], consisting of three losses: onset regression, offset regression, and frame classification. The frame classification loss is widely used in previous works [2] [11] [6] [3] and is simply a binary prediction at every pitch and time frame for if a note at that pitch is played or not. The target and predicted output is thus a $T \times K$ matrix where K is the number of piano notes (pitches), and T is the number of time frames. We use the following loss function:

$$l_{fr} = \sum_{t=1}^{T} \sum_{k=1}^{K} l_{bce}(I_{fr}(t,k), P_{fr}(t,k)) \tag{1}$$

where I_{fr} are the binary targets, P_{fr} are the predictions, and l_{bce} is the binary cross-entropy defined as:

$$l_{bce}(y,p) = -y \ln p - (1-y) \ln (1-p) \tag{2}$$

Following [7], we employ a system of predicting continuous onset and offset regression times . The benefits of this system is that it is more robust to labelling errors, is less sensitive to large frame size, and can predict more accurate onset and offset times within a given frame.

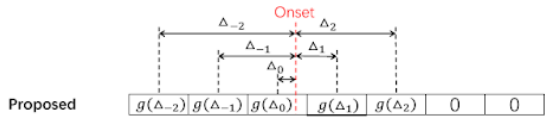

Figure 2. Training targets of the system, image from [7]

We predict the time distance between the centre of the containing and neighbouring frames to a precise onset or offset target. Letting Δ be the length of the frame, and Δ_i be the the time differences, where i is the index of a frame, we encode the time differences targets following g:

$$g(\Delta_i) = \begin{cases} 1 - \frac{|\Delta_i|}{J\Delta}, |i| \leq J \\ 0, |i| > J \end{cases} \tag{3}$$

where the hyperparameter J controls the number of neighbours. We will use $J = 5$, following [7]. In training, our outputs and inputs are again of shapes $T \times K$, where G_{on} and G_{off} are the respective onset and offset regression targets, and R_{on} and R_{off} are the predicted values. Let l_{on} be the onset regression loss and l_{off} be the offset regression loss:

$$l_{on} = \sum_{t=1}^{T} \sum_{k=1}^{K} l_{bce}(G_{on}(t,k), R_{on}(t,k)) \tag{4}$$

$$l_{off} = \sum_{t=1}^{T} \sum_{k=1}^{K} l_{bce}(G_{off}(t,k), R_{off}(t,k)) \tag{5}$$

We again use binary cross-entropy loss since all the values are between 0 and 1. The total loss is thus:

$$l = l_{fr} + l_{on} + l_{off} \tag{6}$$

3.3. Model Architecture

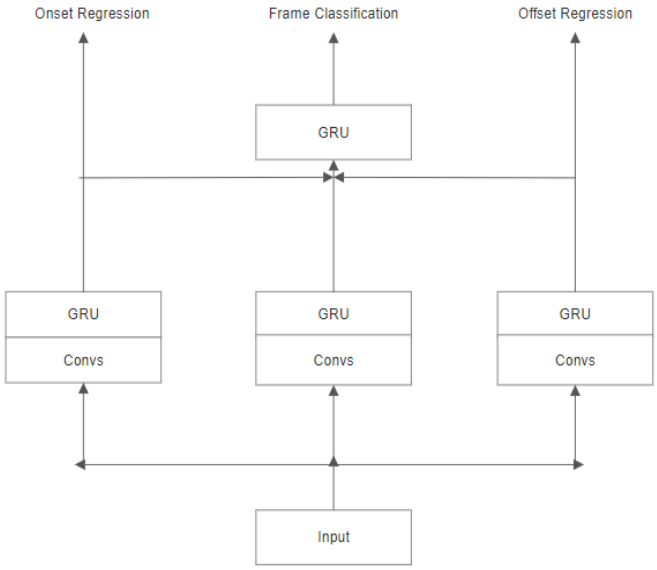

Figure 3. Architecture with BiGRU units (CFP-GRU and Mel-GRU)

Our model uses 3 acoustic models, where each acoustic model is in charge of a certain component of the loss: onset

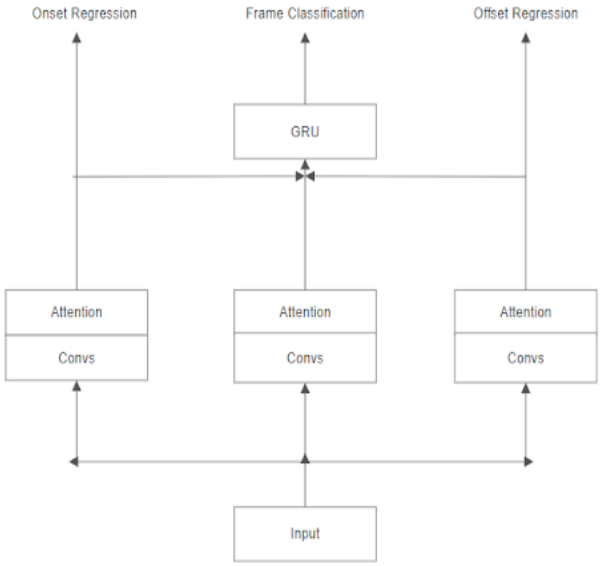

Figure 4. Architecture with self-attention (CFP-Att and Mel-Att)

regression, frame classification, and offset regression. Each acoustic model consists of 4 convolution blocks to learn features, and each convolution block consists of 2 convolution layers with pooling. Whereas the model in [7] followed the convolution blocks with bi-directional gated recurrent units (BiGRU), we used a multi-headed self-attention block as used in [15], to hopefully better capture long-term features of the sound, especially in the multi-instrumental case:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{n}}V) \quad (7)$$

where n is the number of heads, and Q, K, and V are the feature maps of our inputs transformed by the convolutions. However, we face the same problem as [15], where the matrix multiplication QK^T is too computationally expensive, and therefore similarly adopt a local self-attention method inspired by [8]. We will apply 1D local self-attention, where we split our feature map input into query blocks of length l_q, and apply self-attention on only that block and a memory block of length l_m, which consists of the inputs in the map directly before the query block in raster-scan order. We also had the option of doing local-2D attention as done in [15], but found 1D more computationally feasible while still allowing us to set l_q and l_m to be long to enough to capture at least one row in our input.

The entire self-attention block is then formulated as [15]:

$$q_a = \text{layernorm}(q + \text{dropout}(\text{1dAtt}(Q, K, V))), \quad (8)$$

$$q' = \text{layernorm}(q_a + \text{dropout}(W_1\text{ReLu}(W_2 q_a)) \quad (9)$$

where $Q = W_Q q$, $K = W_K q$, and $V = W_V q$, where q is the feature map input. W_1, W_2 are weights from fully-connected layers, and all the W are weight parameters to learn.

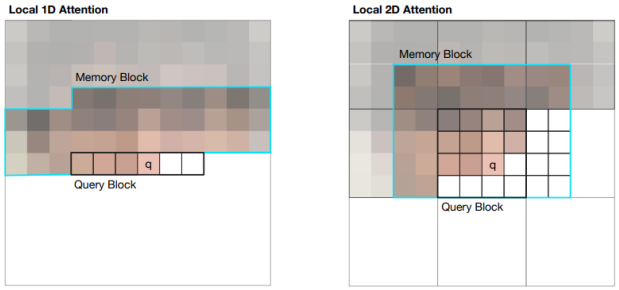

Figure 5. Demonstration of local 1D and 2D attention with query and memory blocks, image from [8]

Before settling on a final model, we experimented with having a single self-attention block per acoustic model, and having an attention block followed by a BiGRU. The outputs of the three blocks are then combined in a separate BiGRU layer to determine the final frame classification output, as the onset and offset time predictions may be useful in predicting frame classification. All three prediction outputs are then fed to time-distributed fully connected layers to give the proper output shape. The outputs are thus dimensions $T \times K$, where T is the number of time frames and K is the number of pitches.

3.4. Inference

During inference, we input the converted audio file to get the predicted frame labels, onset regression, and offset regression. We use the same algorithm as introduced in [7] to transform these output values to actual frame predictions and MIDI notes, where we have settable $\theta_{on}, \theta_{off}, \theta_{fr}$ thresholds to respectively control how confident the model's onset, offset, and frame predictions need to be to actually predict a note. The outputted MIDI notes can then be easily used to replay the predicted audio, or be transcribed to other notation such as sheet music.

4. Experiments/Results

4.1. Preprocessing

Before we ran our experiments, we preprocessed the MAESTRO and MusicNet into Mel representation and into CFP representation. For CFP, the audio files were initially sampled at 44.1 kHz, and were split into 5-second segments, then processed into CFP following [15], with a frame length of 0.02 seconds, with a total of $F = 384$ frequency bins. The final shape of each input was thus $T \times F \times 2$, where

$T = 256$ is the number of frames. For Mel, we used 5-second segments, with a frame length of 0.02 seconds, and following [7], used 229 frequency bins, with a final output shape of $257 \times 229 \times 1$, where the extra frame comes from the 'center=True' argument used during mel extraction. For MusicNet, there were 59891 training segments and 715 test segments. For MAESTRO, there were 288396 training segments, 34691 validation segments, and 36458 test segments.

4.2. Model Search

For our experiments, we first searched for the best model on the MAESTRO dataset. We experimented with 5 models based on the piano transcription model [7] which vary only on the final layers of the acoustic models and on the input type. In all models, for each acoustic model, there are 4 convolutional blocks, where each block consists of 2 convolution layers with kernel sizes 3 x 3. Batch normalization and ReLU are applied after each linear convolutional layer. The four convolutional blocks have average pooling, with output feature maps of 48, 64, 92, and 128 respectively. In each acoustic model, after the convolution layers, we used a combination of self-attention blocks and BiGRU layers:

- single attention block with CFP input (CFP-Att)
- single attention block followed by a BiGRU with CFP input (CFP-Att-GRU)
- two BiGRU with CFP (CFP-GRU)
- two BiGRU with Mel (Mel-GRU)
- single attention block with Mel inputs (Mel-Att)

All BiGRU blocks have 2 layers and have hidden sizes 256. We consider Mel-GRU to be the baseline model because it is practically identical to the model described in [7], just without a velocity prediction module. All attention blocks were multi-headed with 8 heads and had l_q = 128 and l_m = 128. We trained each model on the MAESTRO dataset for 25000 iterations. For both model search and final training, we used a batch size of 12 and an Adam optimizer with a learning rate of 0.0005, with the learning rate being reduced by a factor of 0.9 every 5000 iterations. The results after 25000 iterations are in Table 1 and the training and validation curves for all the models are given in figure 5, 6, 7, 8 and 9. Note that ap stands for average precision and the losses use a log scale while the average precision uses a linear scale. We used frame average precision and onset offset mean squared error as metrics for our model search, following [7].

Contrary to our initial hypothesis, the CFP models trained significantly more slowly and with worse results. For example, CFP-GRU took 4 hours and 17 minutes to run 25000 iterations while MEL-GRU only took 3 hours and 9 minutes. We even tried training CFP-Att to 80000 iterations, but the average precision score stayed lower than 0.6.

Model	Frame ap	Onset MAE	Offset MAE
CFP-Att	0.5239	0.1081	0.1183
CFP-Att-GRU	0.5171	0.1075	0.105
CFP-GRU	0.5265	0.1213	0.1072
Mel-GRU	0.7801	0.1081	0.1238
Mel-Att	0.7488	0.1422	0.1547

Table 1. Validation results after 25000 iterations (ap means average precision, MAE means mean squared error)

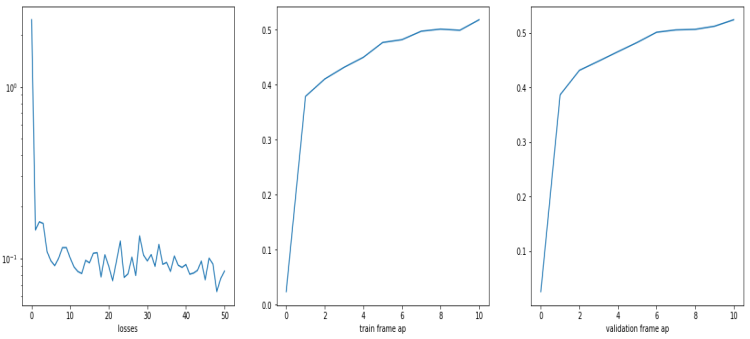

Figure 6. CFP-Att

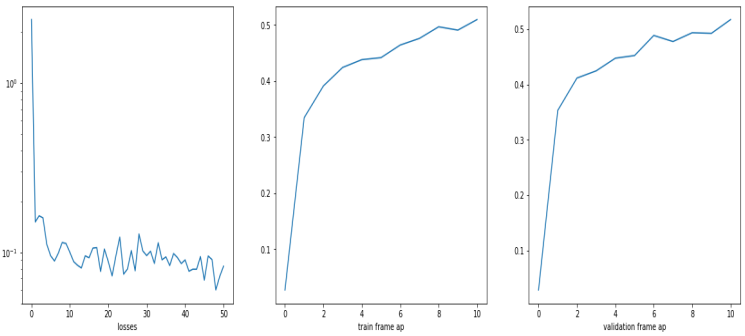

Figure 7. CFP-Att-GRU

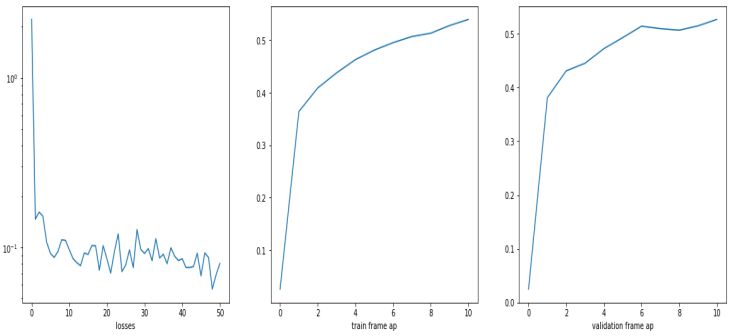

Figure 8. CFP-GRU

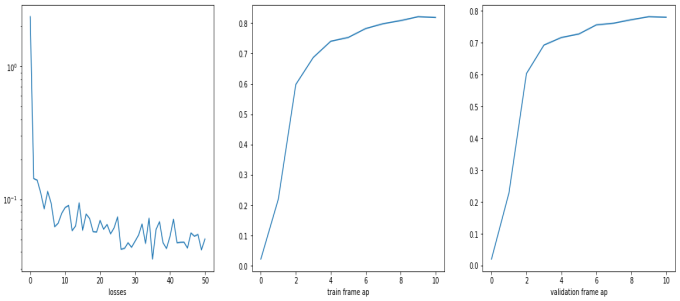

Figure 9. Mel-GRU

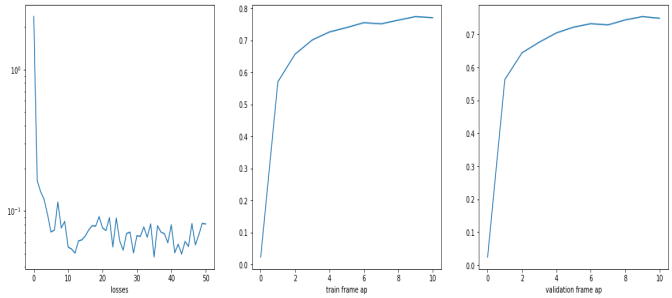

Figure 10. Mel-Att

All three CFP models performed very similarly, with single attention barely performing better than attention-gru. When training with Mel, we ultimately decided to use single-attention as it was computationally the faster to train as well. As shown above, the Mel models performed significantly better in the model search, and we thus decided to use Mel-Att as our final model to continue training.

4.3. Final training

We further trained Mel-single attention and the Mel-Gru model as a baseline on both MAESTRO and MusicNet. For MAESTRO, we trained for a total of 125000 iterations at batch size 12, which is the equivalent to around 5 epochs. For MusicNet, we trained for a total of 100000 iterations at batch size 12, which is equivalent to around 20 epochs due to being a much smaller dataset compared to MAESTRO. The loss curves are given in figure 10, 11, 12, 13. All our code for CFP models is here `https://github.com/jacqueschen1/piano_transcription`, and for Mel models is here `https://github.com/jacqueschen1/MusicTranscription`.

4.4. Evaluation

Once training was complete on both models, we ran the inference function as described earlier on the test sets of MAESTRO and MusicNet to get frame-level predictions and note-level MIDI predictions, using $\theta_{on} = 0.1$, $\theta_{off} =$

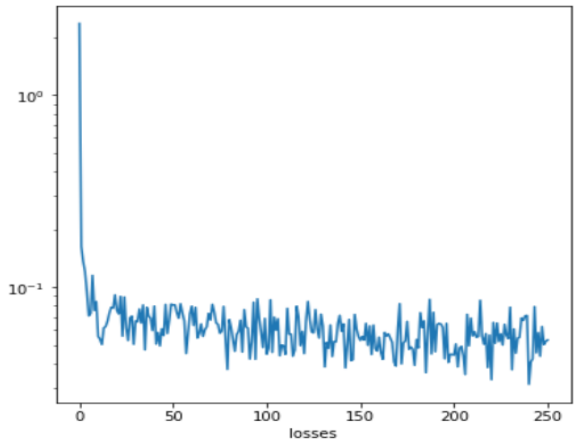

Figure 11. Training loss of Mel-Attention on MAESTRO dataset

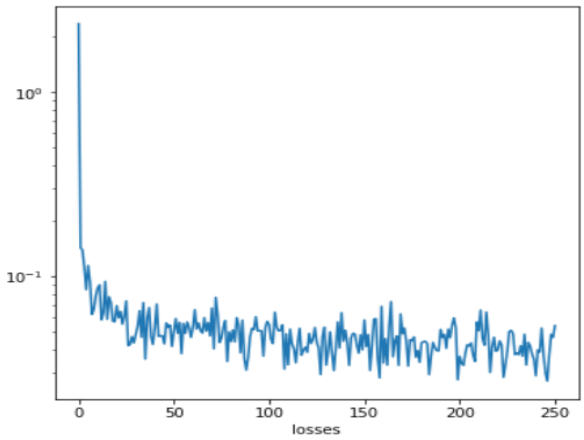

Figure 12. Training loss of Mel-GRU on MAESTRO dataset

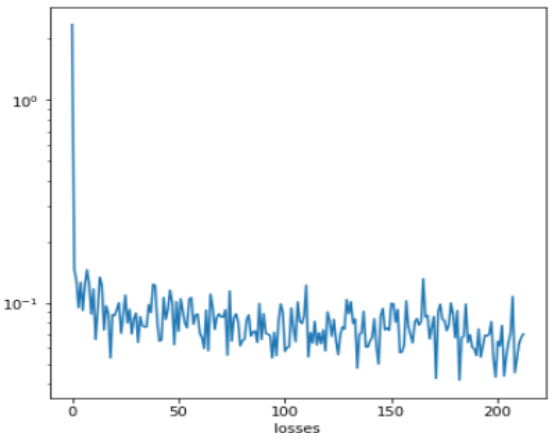

Figure 13. Training loss of Mel-Attention on MusicNet dataset

Model Type	Dataset	Iterations	Frame P (%)	Frame R (%)	Frame F1 (%)	Note P (%)	Note R (%)	Note F1 (%)
Mel-Att	MAESTRO	250000	80.72	68.97	72.77	88.58	76.30	81.77
Mel-GRU	MAESTRO	250000	86.55	61.48	70.06	86.63	83.04	84.77
Mel-Att	MusicNet	200000	68.83	81.84	74.37	89.09	87.83	88.33
Mel-GRU	MusicNet	200000	72.55	80.48	76.14	84.62	92.06	88.09

Table 2. Frame and Note precision, recall, and F1 scores with 500ms onset/offset note tolerance

Model Type	Dataset	Iterations	Note P (%)	Note R (%)	Note F1 (%)
Mel-Att	MAESTRO	250000	45.94	40.14	42.74
Mel-GRU	MAESTRO	250000	45.55	44.05	44.78
Mel-Att	MusicNet	200000	42.70	42.31	42.46
Mel-GRU	MusicNet	200000	46.15	49.47	47.71

Table 3. Note precision, recall, and F1 scores with 50ms onset/offset note tolerance

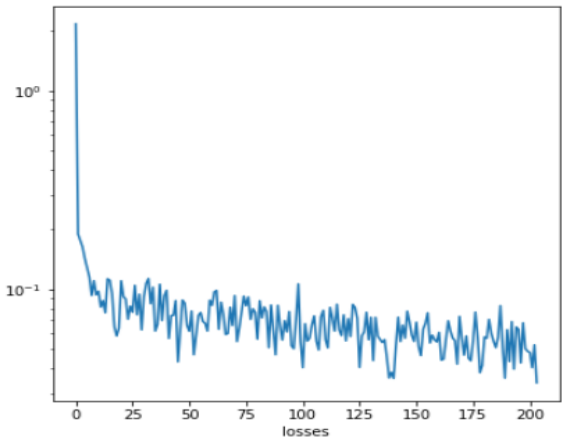

Figure 14. Training loss of Mel-GRU on MusicNet dataset

0.1, and $\theta_{fr} = 0.3$. To evaluate our model, we used frame-level and note-level F1 score, a commonly used metric for this task [7] [14] [15] [3]. For frame-level F1, you simply treat every potential time-step and note combination as a classification problem, with the label 1 being that note should be played at that time-step, and 0 being it should not. For note-level F1, we used a metric implementation developed by [9] specifically for this task, which takes in the note predictions and note targets, along with an onset and offset tolerance time, which determines how off the onset/offset times of the predictions can be to the target while still being labelled correct. We measured F1 scores with a tolerance of both 500ms and 50ms.

As shown from the tables above, the baseline models and the attention models achieved similar scores on both note and frame-level F1. The similarity in scores on both MusicNet and MAESTRO shows that both models were able to successfully transcribe multi-instrumental music. Interestingly, the attention models seemed to achieve better frame recall and worse frame precision, while also achieving worse note recall and better note precision. Both models achieved much lower note-level F1 scores when the onset/offset tolerance was set to 50ms, though the baselines did have slight higher scores in both datasets, as opposed when using 500ms, when the attention model had slightly higher note scores in MusicNet. This suggests that while they may be sufficiently strong at predicting the correct notes, they may need to be further trained or trained with different hyperparameters to achieve more accurate note prediction times.

5. Discussion

Although we were able to get our final attention model to be comparable to the baseline model (Mel-GRU), there is still a lot of work that needs to be done on it. One issue that came up during training was the memory usage of attention. We trained our data on a 11 GB GPU and we found that we would run out of memory if we used a batch size of 12. We overcame this issue by accumulating gradients, which means we ran two iterations with batch size 6 and stepped every 2nd iteration. This issue was more pronounced when we tried using two attention blocks (not listed in the section 5), which we could only run with a batch size 4. Another issue was that we could not increase our memory and query length hyper-parameters in the self-attention blocks due to GPU constraints, which may have also improved performance.

One difficulty that we underestimated was how much more time we needed to run experiments such as hyperparameter search and training the model. We were unable to run as many experiments as we had hoped due to time constraints so we ended up not training the models for as long

as in the papers. For example, [7] trained their piano transcription model on the MAESTRO dataset for 4 days while we could only manage about 1 day. Further exploration into model designs that could capitalize better on CFP inputs could also be another exploration point. Given that we only had a couple weeks to do all of our experiments, it's not surprising that our attention model did not surpass the baseline piano transcription model, but if we had more time for optimization, we could potentially have improved drastically.

One thing we didn't end up having enough time to do was training our models on datasets that included more instruments and genres of music. We ended up only training our models on classical music datasets that included piano and chamber music. In the future we would like to train our models on datasets that include guitar, drums, and other genres of music besides classical. A challenge for this is that music datasets of varying genres with accurate note labels are difficult to find, and the data representations of each dataset can often vary quite drastically, leading to increased time needed to preprocess. For example, MAESTRO labels were provided in raw MIDI files while MusicNet's were provided in CSV files, requiring completely different preprocessing. In general, the preprocessing and handling of these large music datasets into representations like CFP and Mel was a larger challenge than expected, and in the future we would definitely tackle that with more care and thought and have systems in place to more efficiently process the data.

Another idea that we didn't end up exploring was developing methods and techniques to make the model more robust to lower fidelity recordings. The datasets we trained on were all high fidelity clear recordings but a lot of music is recorded at a lower fidelity. We tried to do inference on a noisy recording of a song and compared it to the same song but played at a higher fidelity and the models performance on the lower fidelity song was quite poor and only managed to predict a few of the notes, while the model created a reasonable transcription for the higher fidelity recording.

6. Conclusion

We experimented with incorporating Combined Frequency and Periodicity (CFPs) inputs and log mel inputs as well as attention into the piano transcription model described by [7]. We found that when we use log mel inputs with a single attention block, we were able to get comparable performance with the baseline model. The ideas explored in this project and the preliminary results could be developed further in the future to develop a robust model to transcribe multi instrumental music and could be useful for other researchers developing attention models for other applications.

References

[1] E. Benetos, S. Dixon, Z. Duan, and S. Ewert. Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30, 2019.

[2] Sebastian Böck and Markus Schedl. Polyphonic piano note transcription with recurrent neural networks. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 121–124. IEEE, 2012.

[3] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription. *arXiv preprint arXiv:1710.11153*, 2017.

[4] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the maestro dataset. *arXiv preprint arXiv:1810.12247*, 2018.

[5] Rainer Kelz, Sebastian Böck, and Cierhard Widnaer. Multitask learning for polyphonic piano transcription, a case study. In *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)*, pages 85–91. IEEE, 2019.

[6] Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. On the potential of simple framewise approaches to piano transcription. *arXiv preprint arXiv:1612.05153*, 2016.

[7] Qiuqiang Kong, Bochen Li, Xuchen Song, Yuan Wan, and Yuxuan Wang. High-resolution piano transcription with pedals by regressing onsets and offsets times. *arXiv preprint arXiv:2010.01815*, 2020.

[8] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR, 2018.

[9] Colin Raffel, Brian McFee, Eric J. Humphrey, J. Salamon, Oriol Nieto, Dawen Liang, and D. Ellis. Mir_eval: A transparent implementation of common mir metrics. In *ISMIR*, 2014.

[10] Leland Roberts. Understanding the mel spectrogram, 2020.

[11] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939, 2016.

[12] John Thickstun, Zaid Harchaoui, and Sham Kakade. Learning features of music from scratch, 2017.

[13] Y. Wu, B. Chen, and L. Su. Automatic music transcription leveraging generalized cepstral features and deep learning. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 401–405, 2018.

[14] Yu-Te Wu, Berlin Chen, and Li Su. Polyphonic music transcription with semantic segmentation. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 166–170. IEEE, 2019.

[15] Yu-Te Wu, Berlin Chen, and Li Su. Multi-instrument automatic music transcription with self-attention-based instance segmentation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2796–2809, 2020.