ECE477
Computer Audition

Real-Time Piano Pitch Transcription

Ko Muramatsu

Purpose and Motivation
- Constructing Interactive Model with Polyphonic Instrument

- Piano — 88 keys (MIDI numbers 21-107), Polyphonic instrument
  - The piano is often associated with electronics (e.g., synthesizer), automated performance (e.g., piano rolls), having a close interactive nature through the keyboard interface
    - *Voyager*, George Lewis: https://www.youtube.com/watch?v=o9UsLbsdA6s
    - *Deus Cantando*, Peter Ablinger: https://www.youtube.com/watch?v=Wpt3lmSFW3k
- However, the piano's wide pitch range and complex polyphonic gestures present significant challenges for real-time performance processing.
- Project Goal
  - To develop a robust real-time pitch transcription model of piano, specifically for live performance
  - Integrating the model into creative musical context, such as live piano performances with semi-automated virtual instruments (as if the pianist has a conversation with the virtual instruments)

Voyager, George Lewis

Deus Cantando, Peter Ablinger

Based Model

Examples from Previous Research on Piano Transcription

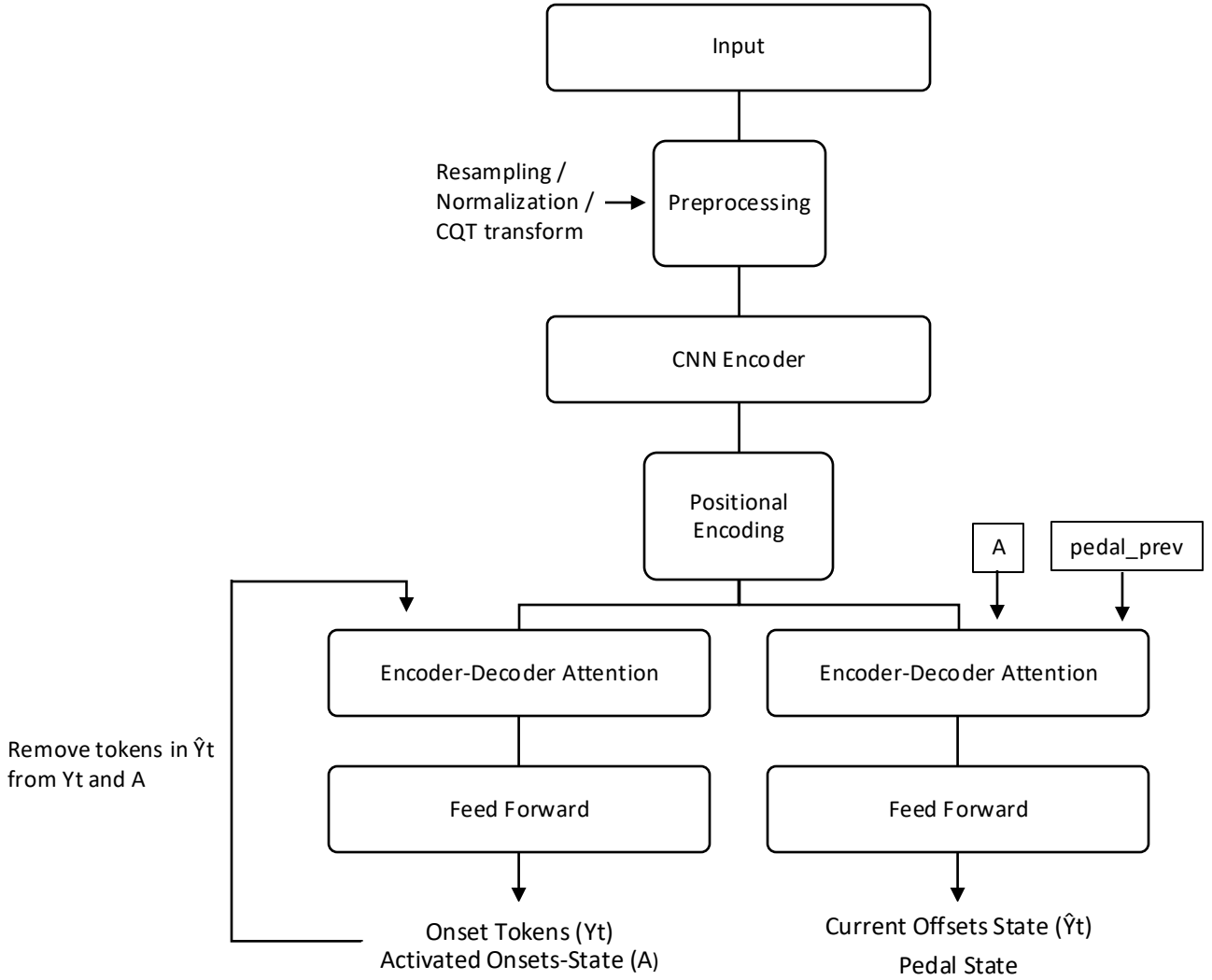
- Automatic Piano Transcription with Hierarchical Frequency-time Transformer (K. Toyama, T. Akama, Y. Ikemiya, Y. Takida, W. Liao, and Y. Mitsufuji), July 2023
  - Utilizes a Transformer with hierarchical layers for onset-offset-velocity detection
  - The first layer has 1D convolutional layer to capture temporal domain information, while the second layer includes Transformer encoder and decoder for precise transcription
- Scoring Time Intervals Using Non-hierarchical Transformer for Automatic Piano Transcription (Y. Yan and Z. Duao), Apr 2024
  - Implements a Semi-Markov Conditional Random Field (semi-CRF) to incorporate time-interval dependencies between events
  - Addresses the potential of semi-CRF. Even with a low time resolution feature map projected from the encoder, the model transcribes the events with high accuracy
- Streaming Piano Transcription Based on Consistent Onset and Offset Decoding with Sustain Pedal Detection (W. Wei, J. Zhao, Y, Wu, K, Yoshii), Nov 2024 in ISMIR 2024
  - Adopts sequence-to-sequence event prediction with two interdependent Transformer decoders: one for onsets and the other for offsets and pedal prediction

> The presented project builds on the latest research. It employs Transformers to leverage the interdependency between encoder and two decoders, predicting onsets, offsets, and pedal states

> Realizes the real-time processing and focuses on minimal latency for the sake of performability

Pipeline

- Dataset used for training: MAESTRO v.3.0.0



- Technical Challenge

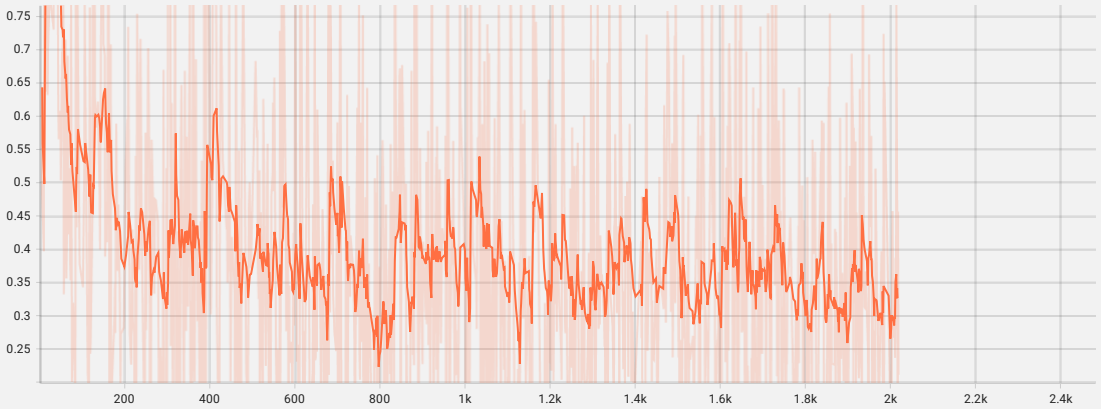
- Handling Loss Functions (Onsets, Offsets, Pedal)
  - Requires fine-tuning of the balance between the three loss functions during training
  - Previous research predicts pedal states in the offset decoder as binary values (on / off)
    - My experimentation showed that predicting pedal states as continuous float values (range: 0.0-1.0) leads to slightly faster and better fit to GT
    - Continuous pedal prediction also helps expressiveness in the realization stage, such as expressing different nuances between half-pedaling and full-pedaling
    - Nevertheless, the loss function of the pedal prediction is unstable, showing a complex relation between pedal and offsets prediction

- Imbalanced Positive/Negative Classes
  - Onset/offset matrices (time_steps, num_keys) are imbalanced in nature
    - False states are dominant over True states
    - Middle pitch ranges are more frequently used than extreme range
      - → Set pos_weight in BCEWithLogitsLoss() to address the issue (but probably it was too much in my experiment:/)

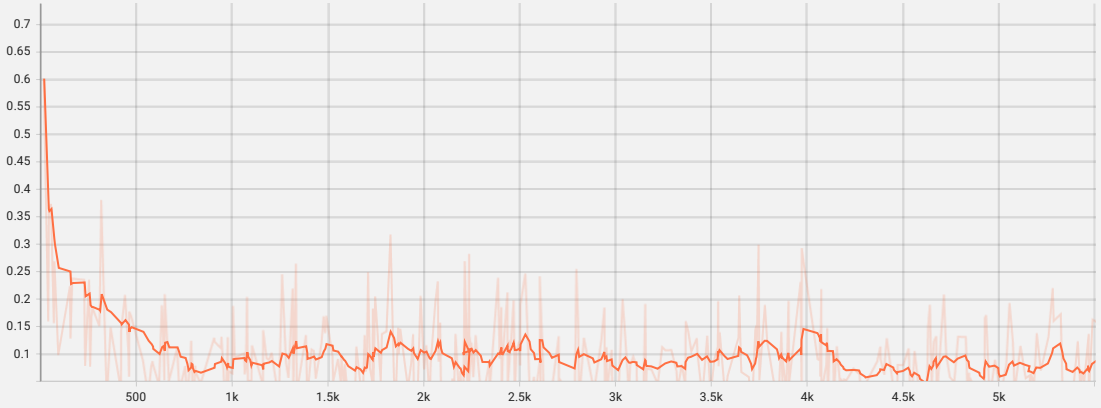
- Time Resolution issue
  - Original model used a CNN encoder with a receptive field of 40 and a hop size of 20 ms, resulting in a time resolution of 800 ms, which is a significant issue for real-time processing
  - Applied overlap factor=0.5 to resolve this, but this ended up having a bad result in evaluation
    - → With M=40 and overlap=0.5 (original model), precision: 0.450, recall: 0.802, F1: 0.613
  - Later started experimenting with reduced receptive field and encoder structure, showing the best result in my experiment
    - → With M=20 and overlap=0.0 (w/o post-processing), precision: 0.464, recall: 0.885, F1: 0.605 (too much weight on predicting True states)
    - → With M=20 and overlap=0.0 (w post-processing), precision: 0.707, recall: 0.796, F1: 0.755

- Observation on logged results

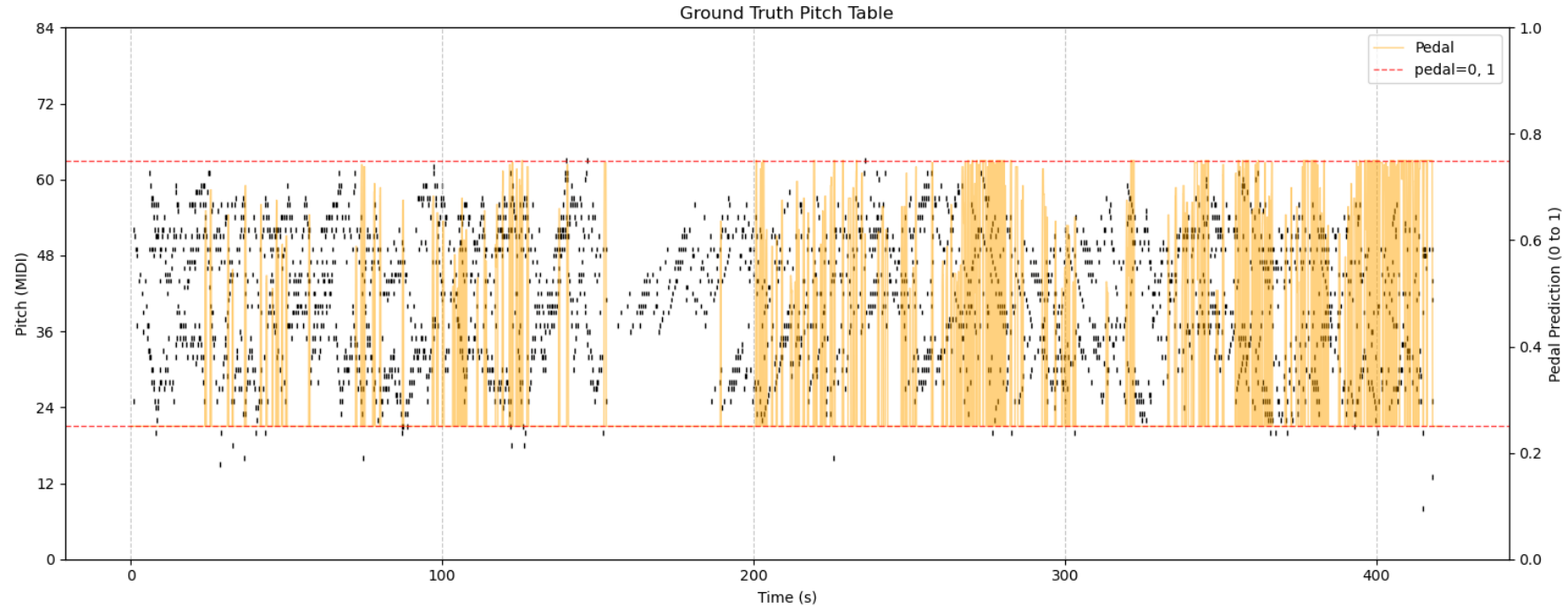
Loss function tracking



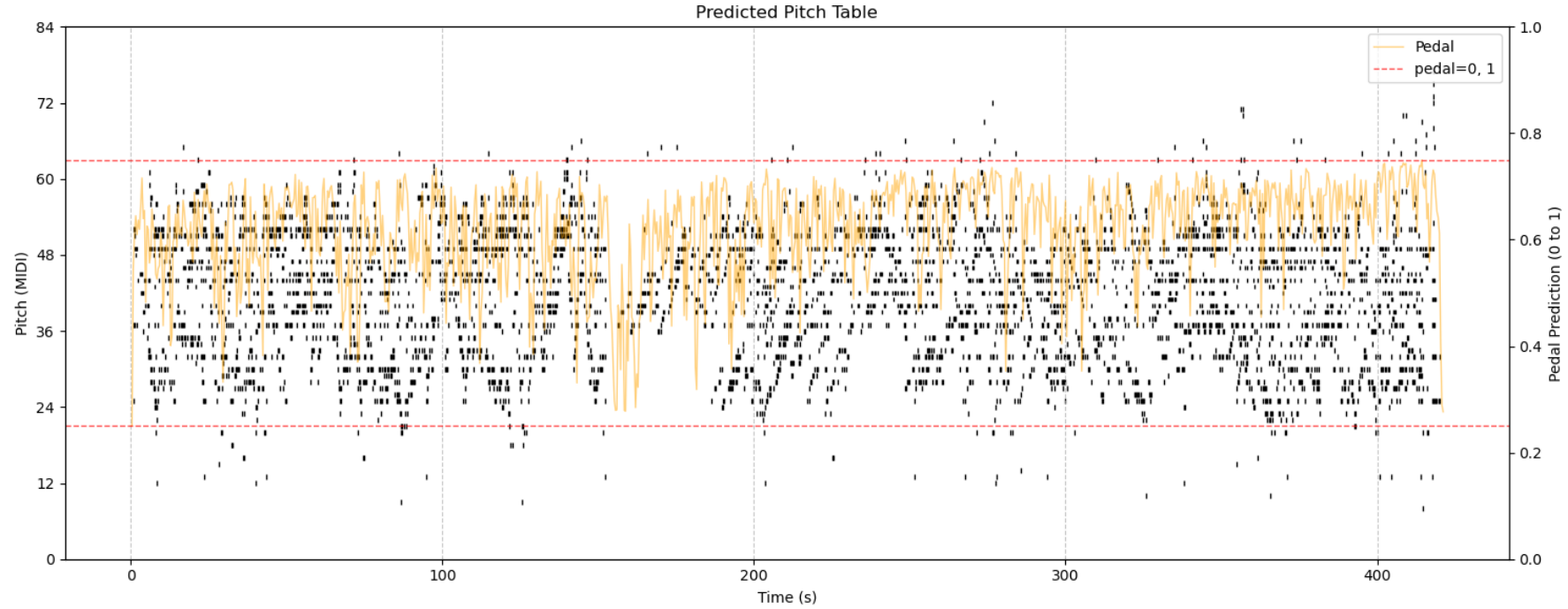
Pedal binary prediction, first 2k steps
(M=40 model)



Pedal as continuous-value prediction, first 5k steps
(M=40 model)



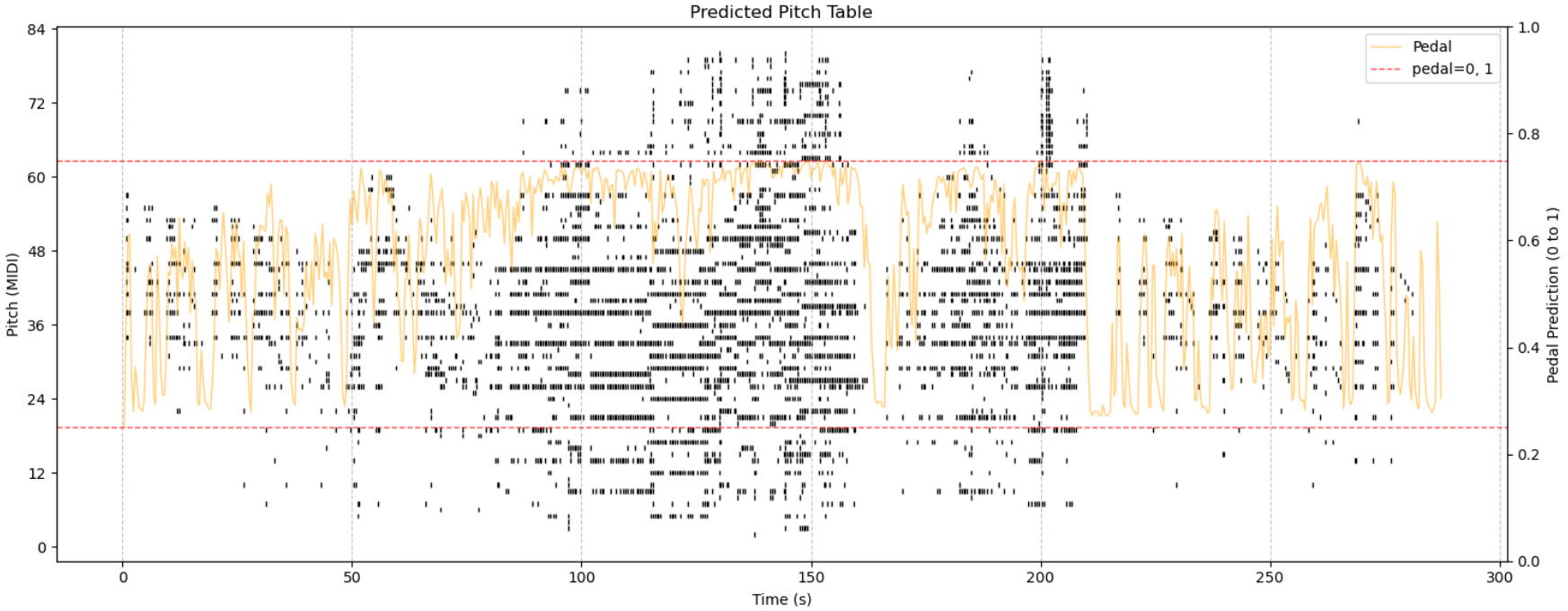
Ground Truth Pitch Table



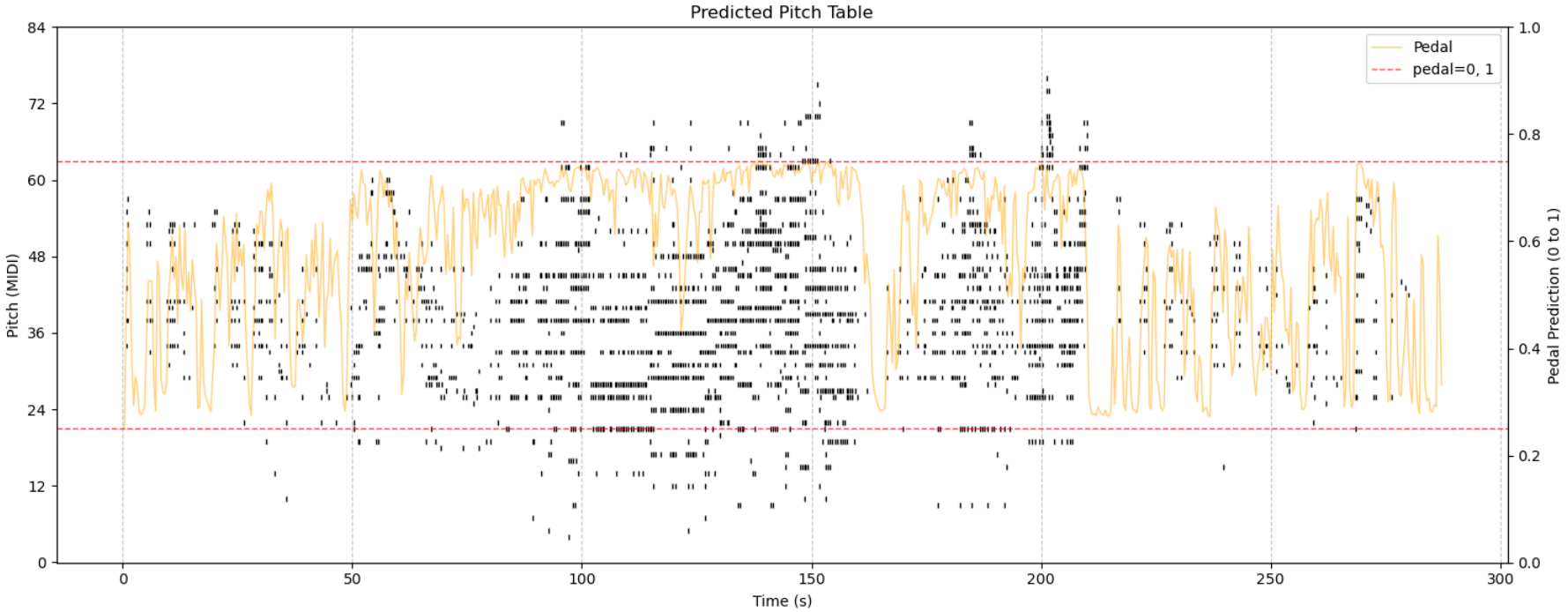
Transcribed Pitch Table by Best Model

- Postprocessed with confidence threshold and offsets compensation
- Prevent predicting notes with lower confidence and durational retention by a lack of corresponding offsets

<u>Prediction of the extreme pitch range is not stable</u>



Transcription by the best model
- without offset post-processing



Transcription by the best model
- with offset post-processing
(by confidence threshold and compensating offsets)

Mainly filter out notes in the extreme pitch ranges

Demo

: Latency of processing the model is appx. 15 ms.
: Predicted notes are triggered by onset slice detection (out of loop from the model)

Left channel:
Live piano →

← Right channel:
Transcribed +
processed sound

https://youtu.be/Baaix6Q3UFM

Room for Improvement

- Regarding the Model:
  - Improve offset prediction accuracy
  - Experiment further with the encoder structure to enhance precision and time resolution
  - Train with longer chunks of samples. It has been trained with appx. 6 seconds samples per batch, but this short duration might have made the training even more imbalanced
  - Continue fine-tuning thresholds and weights
  - Optimize decoders structure for computational efficiency (particularly important for streaming)
  - Any other ideas?

- Regarding Realization / Performance:
  - Reconcile timing mismatches between onset slices (with velocity) and pitch detection (currently detached)
  - Utilize offset prediction effectively (currently it is not correctly handled in the realization)
  - Use stereo mics to balance the spectrum information in high/low pitch register
  - Explore more creative ideas…
    - Experiment with Markov Models and/or HMMs to "predict" upcoming pitches
    - Work with articulations, dynamics, and duration (e.g., determine durations by dynamics, exaggerate long/short duration and strong/soft dynamic contrasts)
    - Explore timbre variation (e.g., synthesized bass sound for left-hand notes, guitar sound for right-hand notes)
    - Introduce intentional delays between real performance and MIDI realization for musical effect, as heard in canon
    - Use pitch values as parameters of audio effect (e.g., spectral reverb on certain pitch-class notes, formants-fixed phase vocoder)
  - To be artistic, creative, smooth, and interesting! Goal is to make it sound "interactive"

Thank you!

Ko Muramatsu

Ph. D. Candidate, Eastman School of Music
Teaching Assistant, Electroacoustic Music Studios @ Eastman (EMuSE)
Technical Director, OSSIA New Music
Phone: 617-651-7336
Email: kmuramat@u.rochester.edu