Empower Music Creation with AI

Hao-Wen Dong

About me

Hi, I'm Herman.
I do Music x AI research.
I love music and movies!

National Taiwan University — 2013 – 2017
B.S. in Electrical Engineering

Academia Sinica — 2017 – 2019
Research Assistant
Music and AI Lab, Research Center for IT Innovation

Yamaha Corporation — Summer 2019
Research Intern
AI Group, Research and Development Division

University of California San Diego — 2019 – 2021
M.S. in Computer Science

Dolby Laboratories — Summer 2021 - present
Deep Learning Audio Intern
Applied AI Team, Advanced Technology Group

University of California San Diego — 2019 – present
Ph.D. in Computer Science (in progress)

Outlines

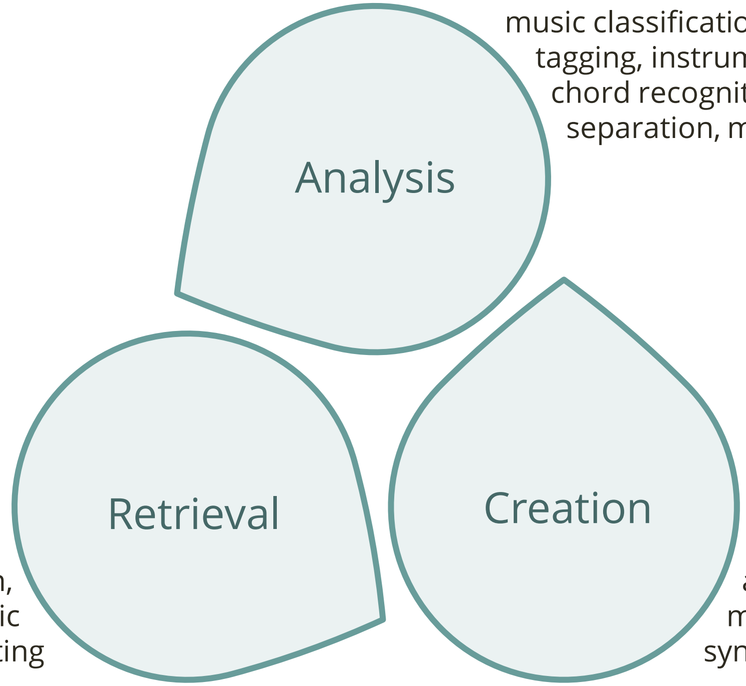
- Music Information Research (MIR)
- MuseGAN – A GAN for music generation (AAAI 2018)
- MusPy – A toolkit for symbolic music generation (ISMIR 2020)
- Arranger – An AI for automatic instrumentation (ISMIR 2021)

Music Information Research

Music Information Research

- Intelligent ways to analyze, retrieve and create music

music classification, music tagging, instrument recognition, chord recognition, source separation, music transcription



music recommendation system, playlist generation, query by humming, music fingerprinting

automatic composition, automatic accompaniment, music style transfer, music synthesis

Yi-Hsuan Yang, "Music Information Research," *Social Networks and Human-Centered Computing (SNHCC)*, Taiwan International Graduate Program (TIGP), lecture notes, April 2018.

Wen-Yi Hsiao

Li-Chia Yang

Yi-Hsuan Yang

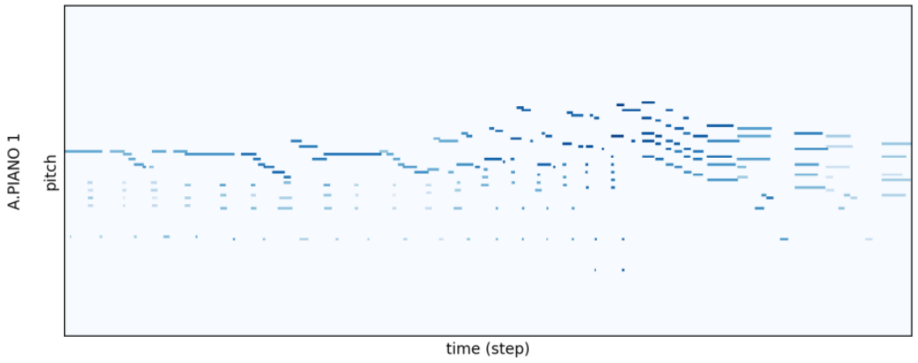
MuseGAN

A GAN for music generation

MuseGAN – Overview

Generate pop music

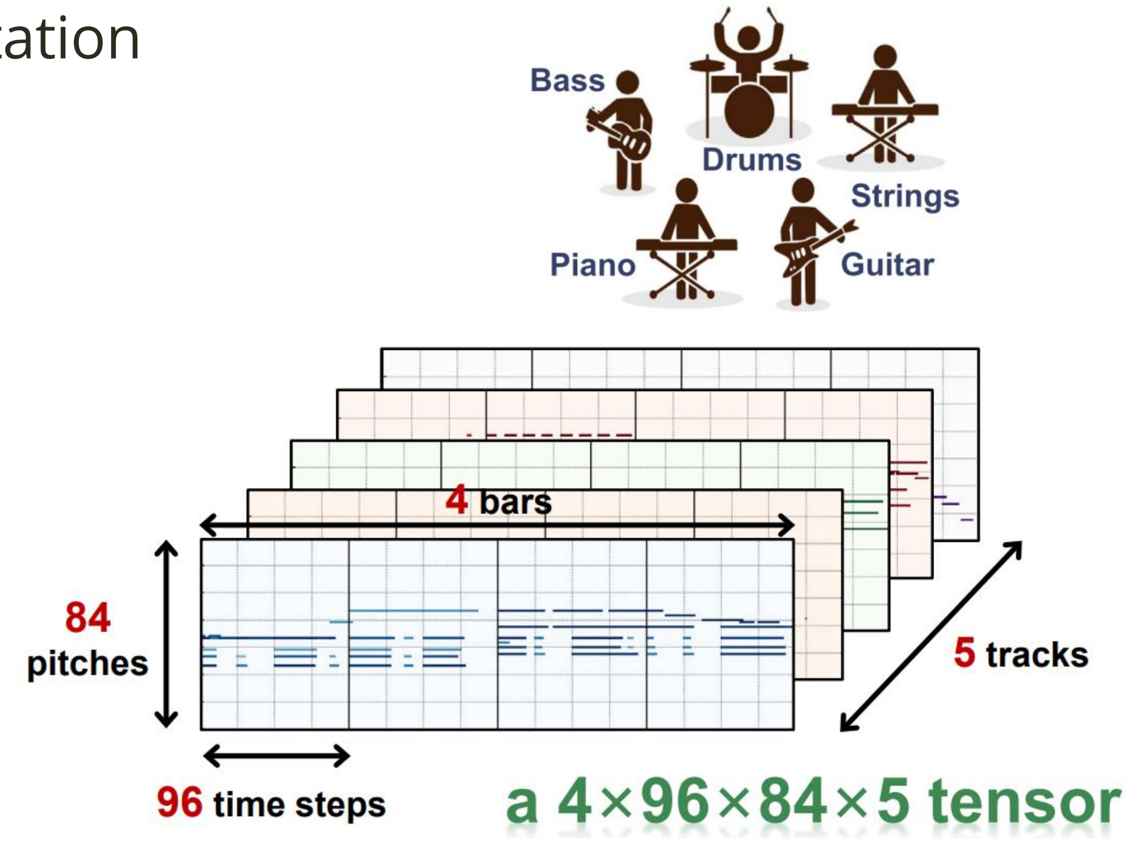
- of multiple tracks
- in a piano-roll format
- using GAN with CNNs



Hao-Wen Dong,* Wen-Yi Hsiao,* Li-Chia Yang, and Yi-Hsuan Yang, "MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment," *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

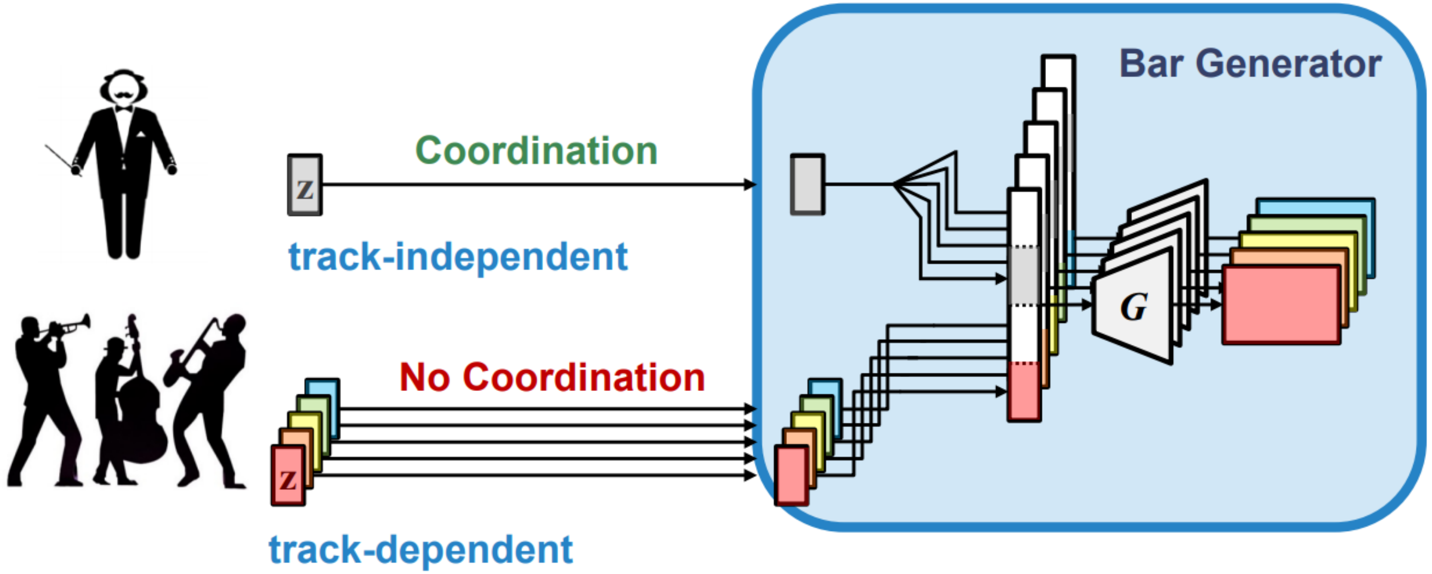
MuseGAN – Data

- Multi-track piano-roll representation
- Lakh Pianoroll Dataset
  - 174,154 multi-track piano-rolls
  - Derived from the Lakh MIDI Dataset
  - Mainly pop songs



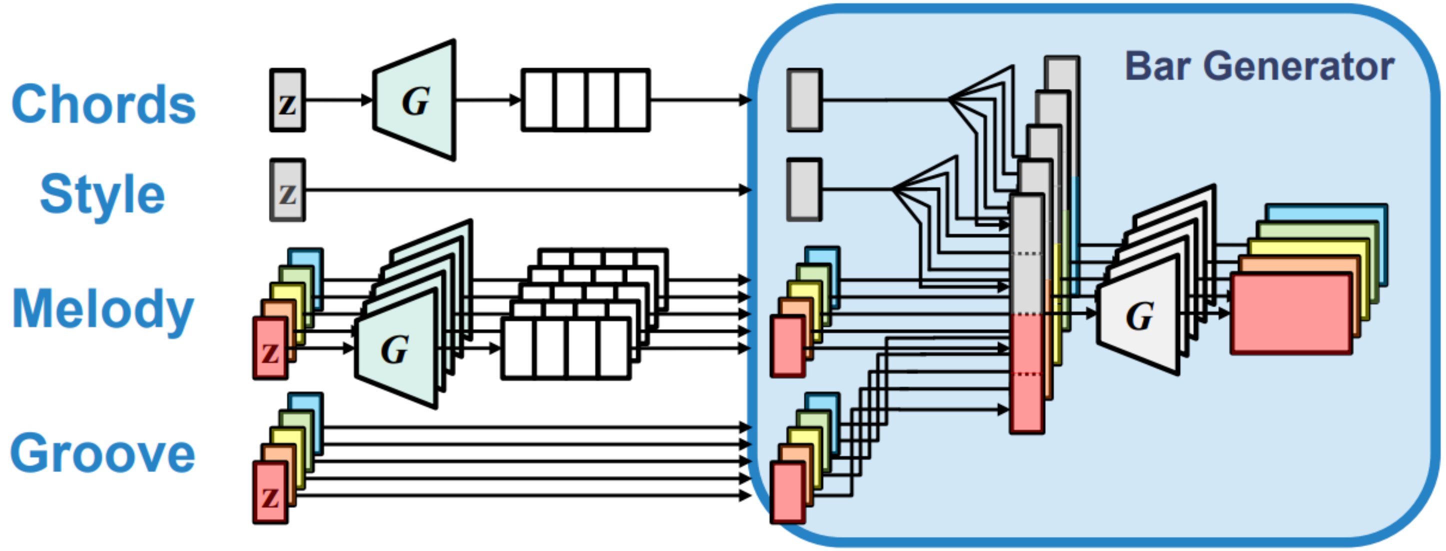
MuseGAN – Model

- Each track takes a shared and a private random vectors as inputs

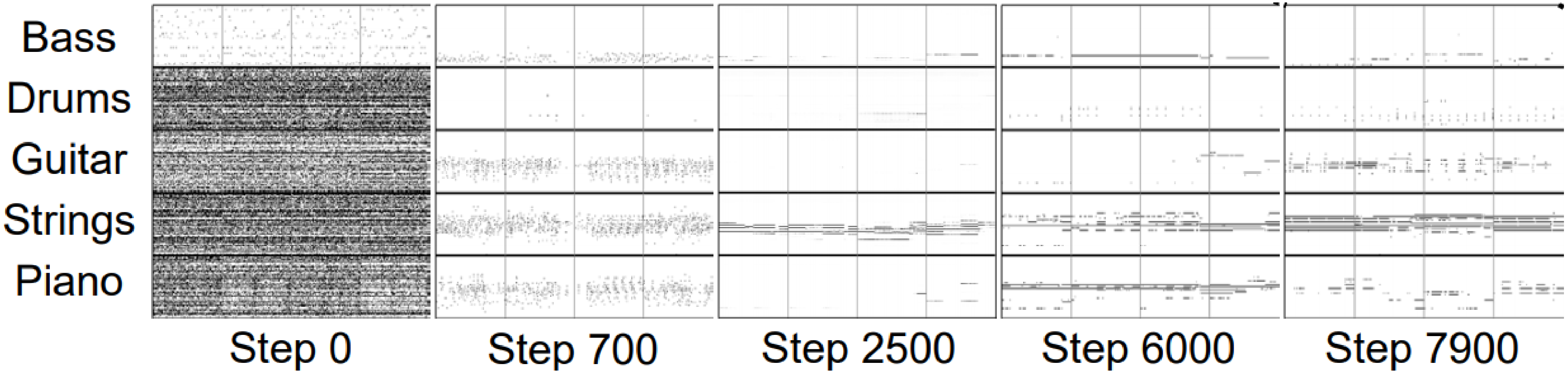


MuseGAN – Model

- Each random vector inputs corresponds to different aspects of music
  - Offer better controllability than one single random vector input



MuseGAN – Results



More examples are available at salu133445.github.io/musegan/ .

MuseGAN – Summary

- First deep learning model for multi-track polyphonic music generation
- Handle track interdependency with shared and private inputs
- Successfully trained on a large dataset of pop music

Ke Chen

Julian McAuley

Taylor Berg-Kirkpatrick

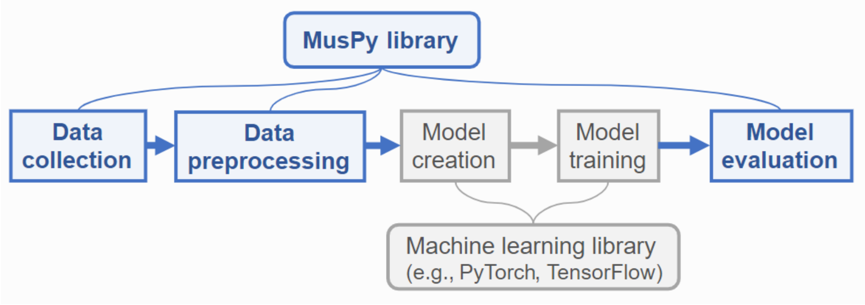
MusPy

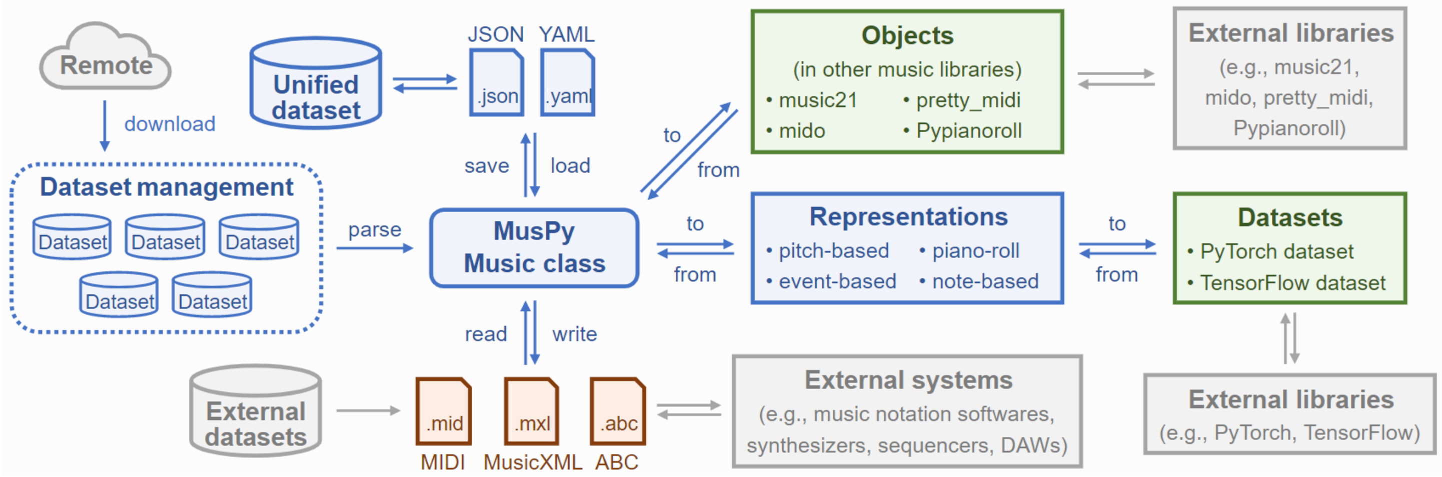
A toolkit for symbolic music generation

MusPy – Overview

A toolkit for symbolic music generation

- Dataset management
- Data I/O
- Data preprocessing
- Model evaluation





Hao-Wen Dong, Ke Chen, Julian McAuley, and Taylor Berg-Kirkpatrick, "MusPy: A Toolkit for Symbolic Music Generation," *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

MusPy – Muspy Music class

- Core class of MusPy
- A universal container for symbolic music
- Serializable to JSON and YAML


```yaml
metadata:
  schema_version: '0.0'
  title: Für Elise
  creators: [Ludwig van Beethoven]
  collection: Example dataset
  source_filename: example.json
resolution: 4
tempos:
  - {time: 0, qpm: 72.0}
key_signatures:
  - {time: 0, root: 9, mode: minor}
time_signatures:
  - {time: 0, numerator: 3, denominator: 8}
downbeats: [4, 16]
lyrics:
  - {time: 0, lyric: Nothing but a lyric}
annotations:
  - {time: 0, annotation: Nothing but an annotation}
tracks:
  - program: 0
    is_drum: false
    name: Melody
    notes:
      - {time: 0, duration: 2, pitch: 76, velocity: 64}
      - {time: 2, duration: 2, pitch: 75, velocity: 64}
      - {time: 4, duration: 2, pitch: 76, velocity: 64}
      - {time: 6, duration: 2, pitch: 75, velocity: 64}
      - {time: 8, duration: 2, pitch: 76, velocity: 64
      - {time: 10, duration: 2, pitch: 71, velocity: 64}
      - {time: 12, duration: 2, pitch: 74, velocity: 64}
      - {time: 14, duration: 2, pitch: 72, velocity: 64}
      - {time: 16, duration: 2, pitch: 69, velocity: 64}
    lyrics:
      - {time: 0, lyric: Nothing but a lyric}
    annotations:
      - {time: 0, annotation: Nothing but an annotation}
```

MusPy – Datasets

- MusPy now supports even more datasets!

Dataset	Format	Hours	Songs	Genre	Melody	Chords	Multitrack
Lakh MIDI Dataset (LMD) [26]	MIDI	>9000	174,533	misc	△	△	△
MAESTRO Dataset [27]	MIDI	201.21	1,282	classical			
Wikifonia Lead Sheet Dataset [28]	MusicXML	198.40	6,405	misc	✓	✓	
Essen Folk Song Database [29]	ABC	56.62	9,034	folk	✓	✓	
NES Music Database [30]	MIDI	46.11	5,278	game	✓		✓
Hymnal Tune Dataset [31]	MIDI	18.74	1,756	hymn	✓		
Hymnal Dataset [31]	MIDI	17.50	1,723	hymn			
music21 Corpus [24]	misc	16.86	613	misc	△		△
Nottingham Database (NMD) [32]	ABC	10.54	1,036	folk	✓	✓	
music21 JSBach Corpus [24]	MusicXML	3.46	410	classical			✓
JSBach Chorale Dataset [11]	MIDI	3.21	382	classical			✓

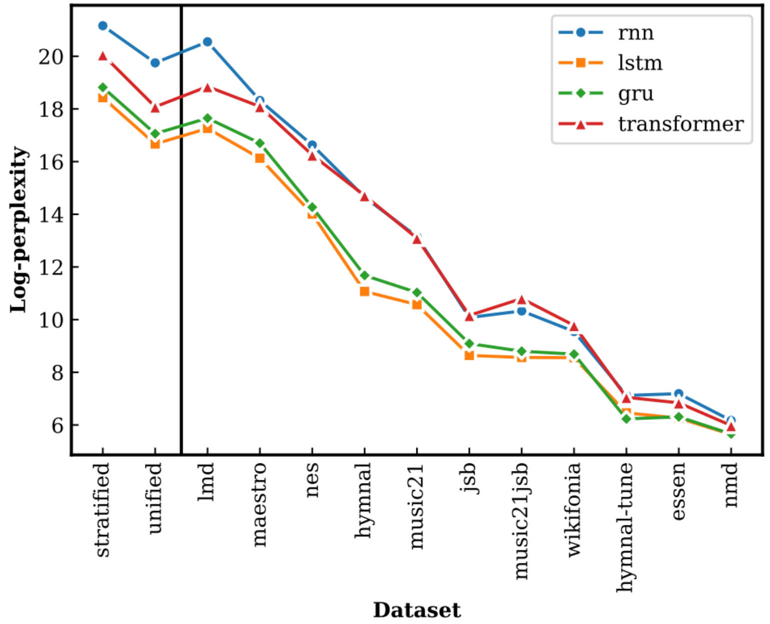
MusPy – Experiments

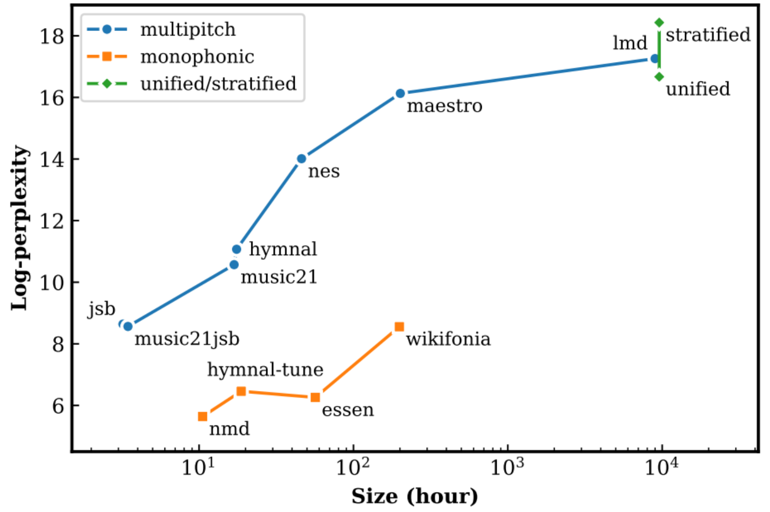
Settings

- Implemented four deep sequential models
  - RNN, LSTM, GRU and Transformer's encoder
- Used a MIDI-like event representation
- Measured the perplexity of 1000 test samples

Results

- All models have similar tendencies
- Perplexity is positively correlated to dataset size
  - Within each group (multipitch vs monophonic)





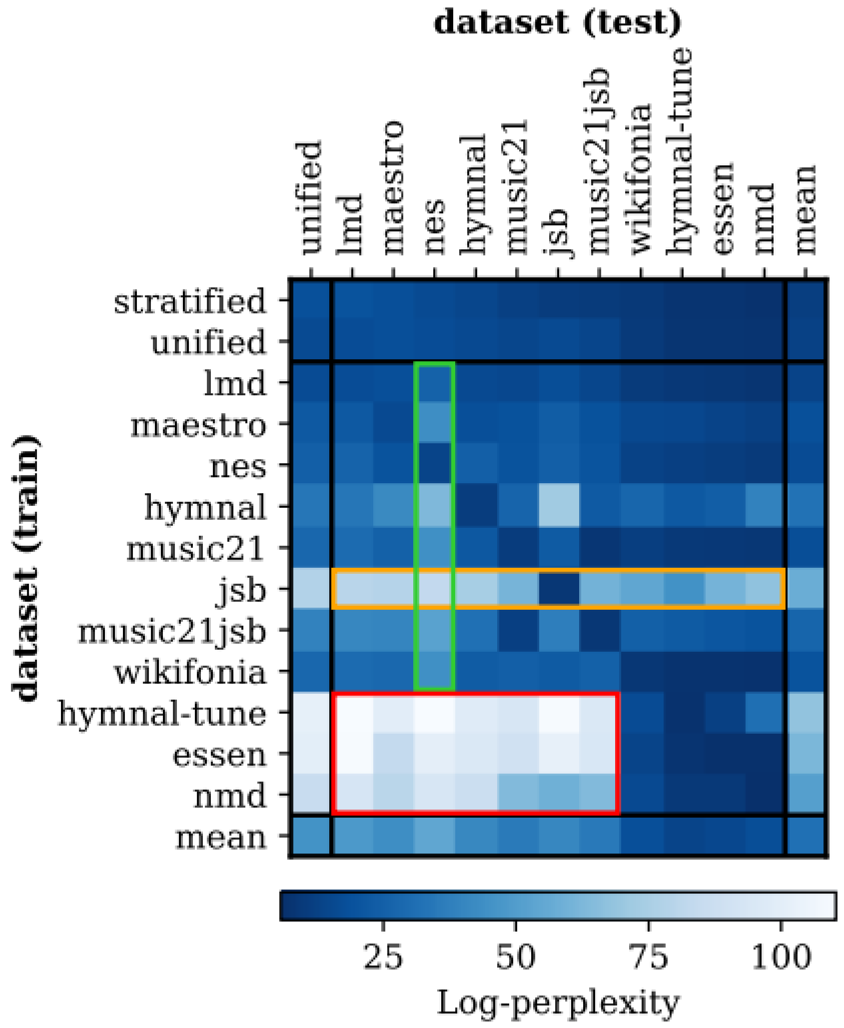
MusPy – Experiments

Settings

- Train a model on some dataset
- Test it on all different datasets

Results

- Cross-dataset generalizability is asymmetric
- A model trained on multi-pitch dataset generalizes well to monophonic dataset, but not the other way around
- The model trained on the combined dataset yields lower perplexity on each dataset



MusPy – Summary

- An open source toolkit for developing a music generation system
- Conducted experiments using MusPy to analyze the relative diversities of different datasets and cross-dataset generalizabilities
- Showed that combining heterogeneous datasets could help improve generalizability of a music generation model

Chris Donahue

Taylor Berg-Kirkpatrick

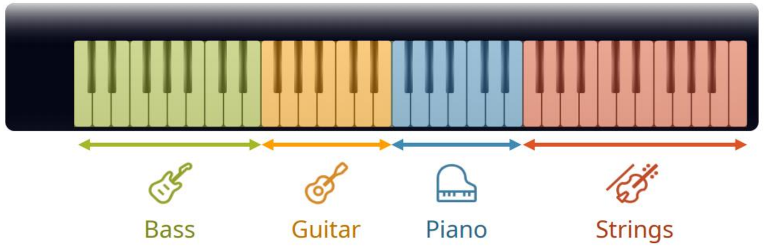
Julian McAuley

Arranger

An AI for automatic instrumentation

Arranger – Overview

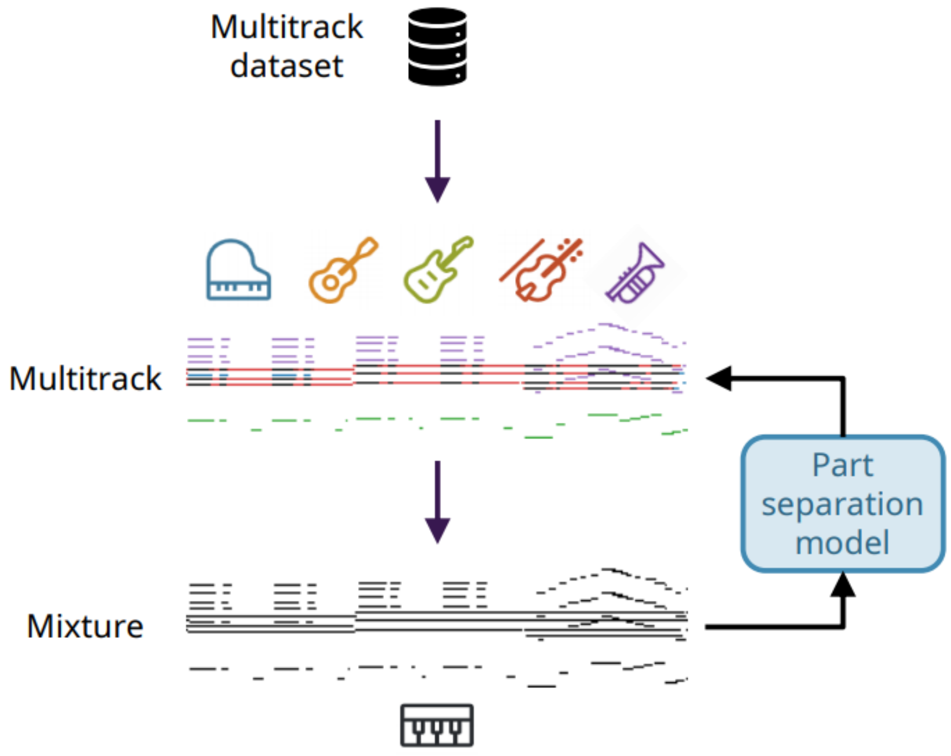
- **Goal**—Dynamically assign instruments to notes in solo music



Hao-Wen Dong, Chris Donahue, Taylor Berg-Kirkpatrick and Julian McAuley, “Towards Automatic Instrumentation by Learning to Separate Parts in Symbolic Multitrack Music,” Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR), in press, 2021.

Arranger – Overview

- Acquire paired data of solo music and its instrumentation
  - Downmix multitracks into single-track mixtures
- Train a part separation model
  - Learn to infer the part label for each note in a mixture
- Approach automatic instrumentation
  - Treat input from a keyboard player as a downmixed mixture
  - Separate out the relevant parts



Arranger – Data

- Four datasets of diverse genres and ensembles

Dataset	Hours	Files	Notes	Parts	Ensemble	Most common label
Bach chorales [31]	3.23	409	96.6K	4	soprano, alto, tenor, bass	bass (27.05%)
String quartets [32]	6.31	57	226K	4	first violin, second violin, viola, cello	first violin (38.72%)
Game music [33]	45.05	4.61K	2.46M	3	pulse wave I, pulse wave II, triangle wave	pulse wave II (39.35%)
Pop music [34]	1.02K	16.2K	63.6M	5	piano, guitar, bass, strings, brass	guitar (42.50%)

Arranger – Models & input features

Models

- Deep sequential models
  - Online LSTM
  - Offline BiLSTM
- Baseline models
  - Zone-based algorithm
  - Closest-pitch algorithm
  - Multilayer perceptron (MLP)

Input features

- time—onset time (in time step)
- pitch—pitch as a MIDI note number
- duration—note length (in time step)
- frequency—frequency of the pitch (in Hz)
- beat—onset time (in beat)
- position—position within a beat (in time step)

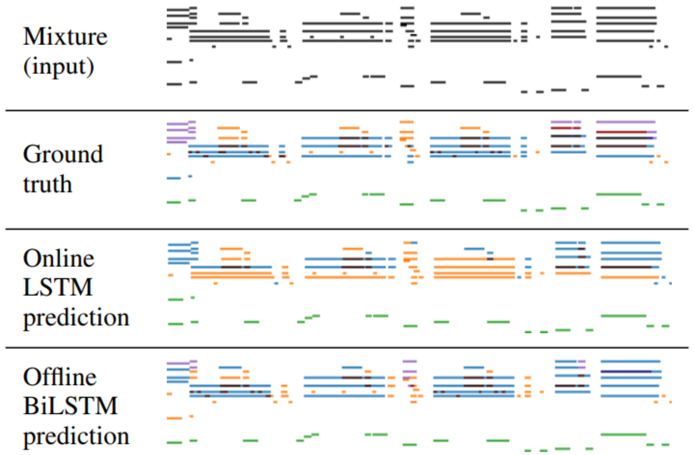
Arranger – Quantitative results

- Proposed models outperform baseline models
- BiLSTM outperforms LSTM
- LSTM models outperform their Transformer couterparts

Model	Bach	String	Game	Pop
Online models				
Zone-based	73.14	58.85	43.67	57.07
MLP [9]	81.63	29.85	43.08*	33.50*
LSTM	**93.02**	**67.43**	**50.22**	**74.14**
Transformer-Dec	91.51	57.03	45.82	62.14
Zone-based (oracle)	78.33	66.89	79.54*	†
MLP [9] (oracle)	97.59	58.16	65.30	44.62
Offline models				
BiLSTM	**97.13**	**74.38**	**52.93**	**77.23**
Transformer-Enc	96.81	58.86	49.14	66.57
Online models (+entry hints)				
Closest-pitch	68.87	50.69	57.14	47.45
Closest-pitch (mono)	89.76	42.82	49.91	32.28
LSTM	**92.70**	**62.64**	**62.11**	**74.19**
Transformer-Dec	91.17	62.12	56.73	67.19
Offline models (+entry hints)				
BiLSTM	**97.39**	**71.51**	**64.79**	**75.59**
Transformer-Enc	93.81	56.72	54.67	67.23

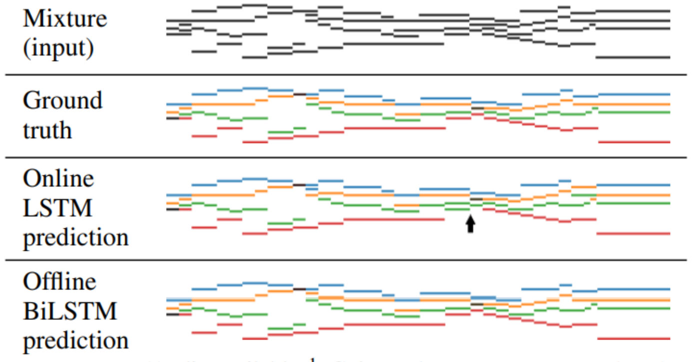
Arranger – Qualitative results

Pop music



(Audio available. [1] Colors: piano, guitar, bass, strings, brass.)

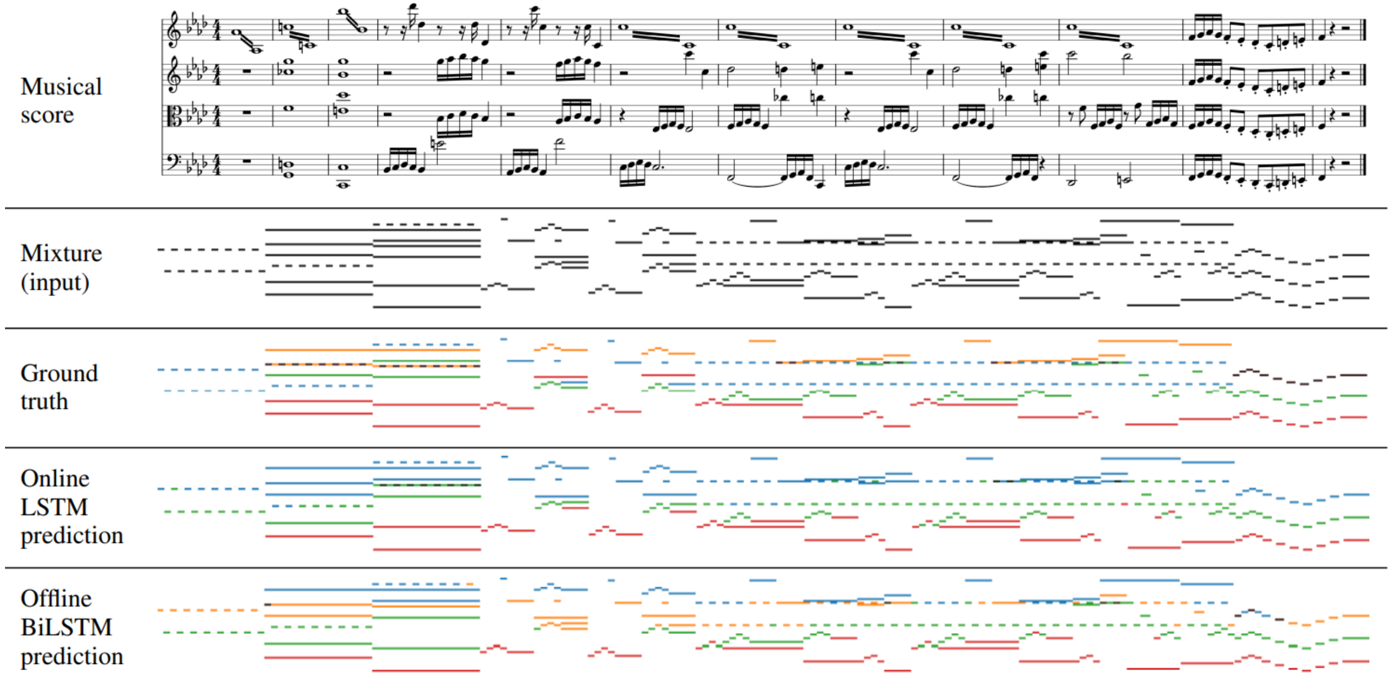
Bach chorales



(Audio available. [1] Colors: piano, soprano, tenor, bass.)

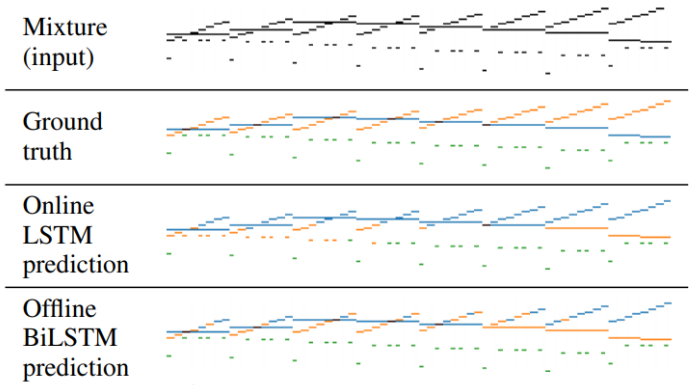
These examples are all hard cases!

String quartets



(Audio available. [1] Colors: first violin, second violin, viola, cello.)

Game music



(Audio available. [1] Colors: pulse wave I, pulse wave II, triangle wave.)

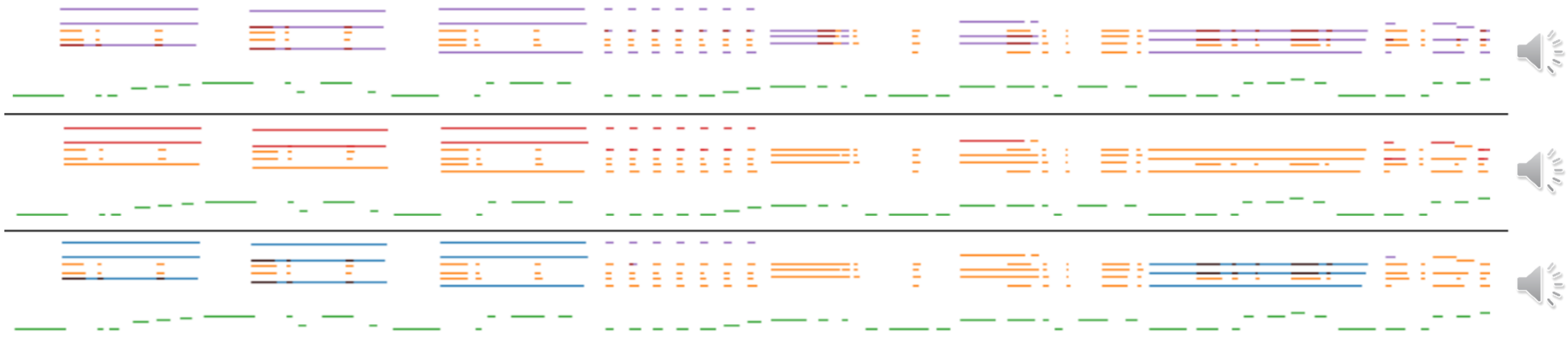
Arranger – Demo

- The proposed models can produce alternative convincing instrumentations for an existing arrangement

Original

LSTM without entry hints

BiLSTM with entry hints



(Audio available. [1] Colors: piano, guitar, bass, strings, brass.)

More examples are available at salu133445.github.io/arranger/ .

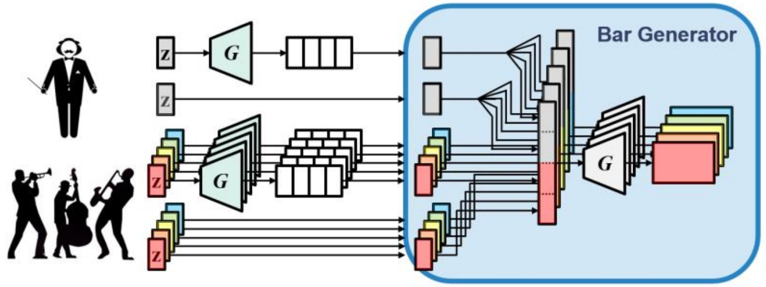
Arranger – Summary

- Proposed a new task of part separation
- Showed that our proposed models outperform various baselines
- Presented promising results for applying a part separation model to automatic instrumentation

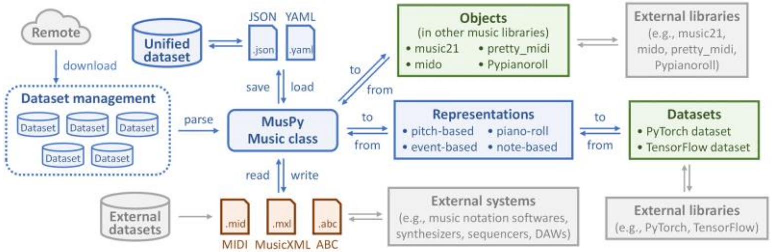
Conclusion

Music x AI

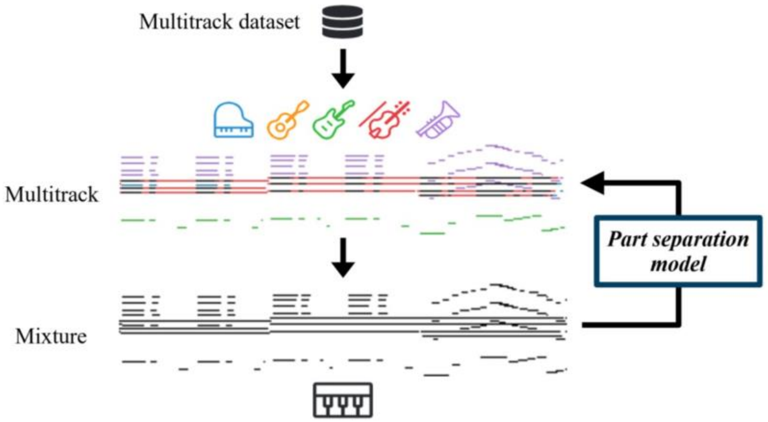
MuseGAN
A GAN for music generation



MusPy
A toolkit for symbolic music generation



Arranger
An AI for automatic instrumentation



To be continued...

Learn more about my projects at
salu133445.github.io .

Thank you!