

2024 edition

Deep Learning for Music Analysis and Generation

Music Classification and Transcription

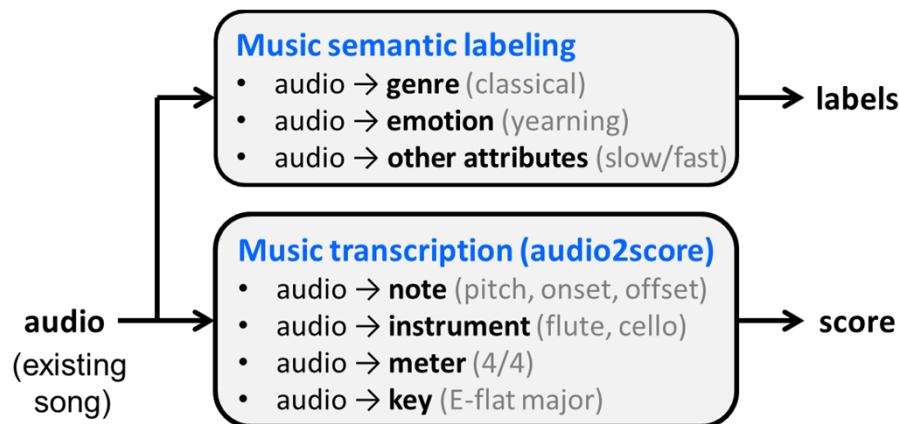
(audio → labels or scores)



Yi-Hsuan Yang Ph.D.
yhyangtw@ntu.edu.tw

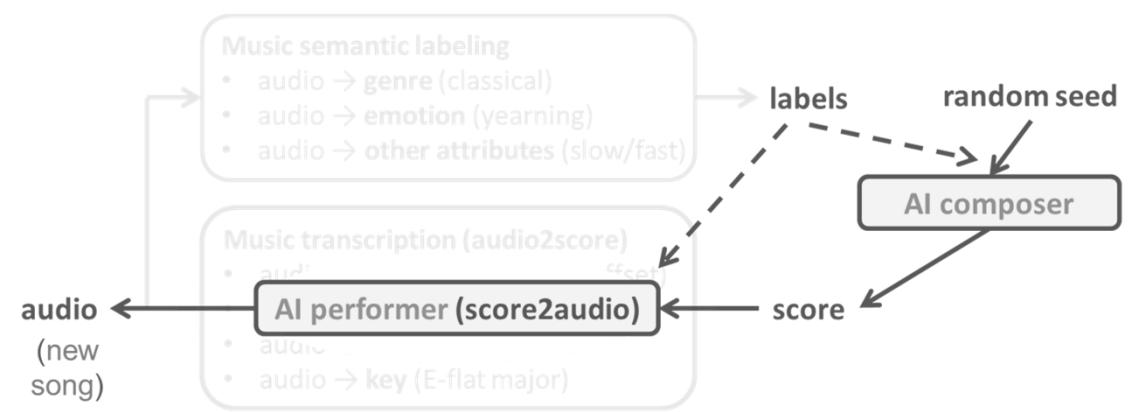
Music AI; or *Music Information Research (MIR)*

- **Music analysis**



- music understanding
- music search
- music recommendation

- **Music generation**



- MIDI generation
- audio generation
- MIDI-to-audio generation

Reference 1: ISMIR 2021 Tutorial

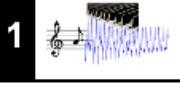
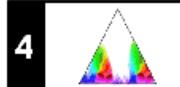
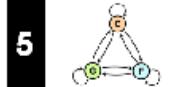
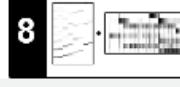
<https://music-classification.github.io/tutorial/landing-page.html>

Music Classification: Beyond Supervised Learning, Towards Real-world Applications

This is a [web book](#) written for a tutorial session of the 22nd International Society for Music Information Retrieval Conference, Nov 8-12, 2021 in an online format. The [ISMIR conference](#) is the world's leading research forum on processing, searching, organising and accessing music-related data.

Reference 2: FMP Notebook

<https://www.audiolabs-erlangen.de/resources/MIR/FMP/C1/C1.html>

Part	Title	Notions, Techniques & Algorithms	HTML	IPYNB
B	 Basics	Basic information on Python, Jupyter notebooks, Anaconda package management system, Python environments, visualizations, and other topics	[html]	[ipynb]
0	 Overview	Overview of the notebooks (https://www.audiolabs-erlangen.de/FMP)	[html]	[ipynb]
1	 Music Representations	Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre	[html]	[ipynb]
2	 Fourier Analysis of Signals	Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT	[html]	[ipynb]
3	 Music Synchronization	Chroma feature, dynamic programming, dynamic time warping (DTW), alignment, user interface	[html]	[ipynb]
4	 Music Structure Analysis	Similarity matrix, repetition, thumbnail, homogeneity, novelty, evaluation, precision, recall, F-measure, visualization, scape plot	[html]	[ipynb]
5	 Chord Recognition	Harmony, music theory, chords, scales, templates, hidden Markov model (HMM), evaluation	[html]	[ipynb]
6	 Tempo and Beat Tracking	Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation	[html]	[ipynb]
7	 Content-Based Audio Retrieval	Identification, fingerprint, indexing, inverted list, matching, version, cover song	[html]	[ipynb]
8	 Musically Informed Audio Decomposition	Harmonic/percussive separation, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF)	[html]	[ipynb]

Reference 3: ISMIR 2018 & 2021 Tutorials

<https://rachelbittner.weebly.com/other-resources.html>

Tutorials

Programming MIR Baselines from Scratch: Three Case Studies

2021

International Society for Music Information Retrieval (ISMIR) conference

- Part 1: Transcription with NMF (Ethan Manilow)
- Part 2: Pitch Tracking with pytorch (**Rachel Bittner**)
- Part 3: Instrument Classification with OpenL3 & Tensorflow (Mark Cartwright)



See the recording here.

Fundamental Frequency Estimation in Music

2018

International Society for Music Information Retrieval (ISMIR) conference

- Part 1: Pitch (Alain de Cheveigné)
- Part 2: Polyphonic fundamental frequency estimation (**Rachel Bittner**)
- Part 3: Applications (Johana Devaney)

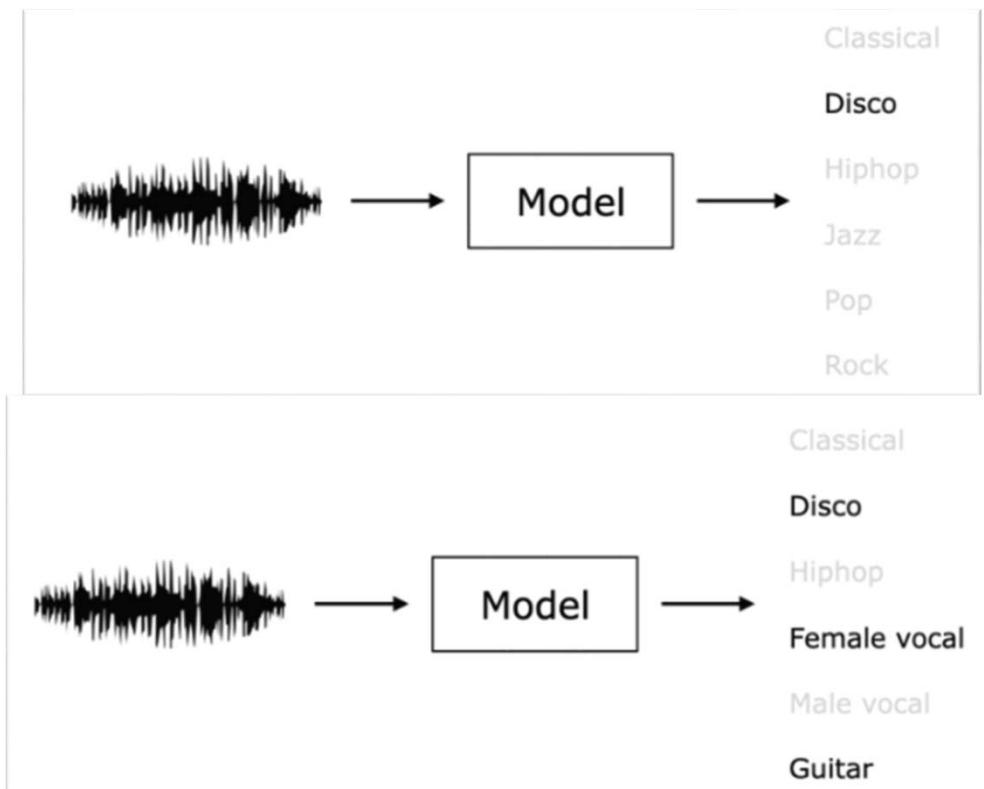
Outline

- **Music classification**
 - **Basics: Tasks, datasets, and evaluation**
 - Supervised models
 - Self-supervised models
- Music transcription
- HW1

Different Classification Tasks

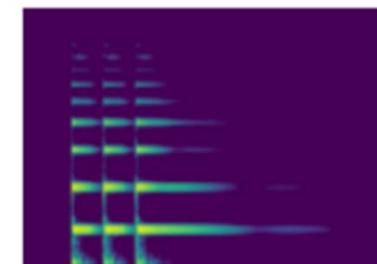
https://music-classification.github.io/tutorial/part1_intro/what-is-music-classification.html

- Single-label vs multi-label

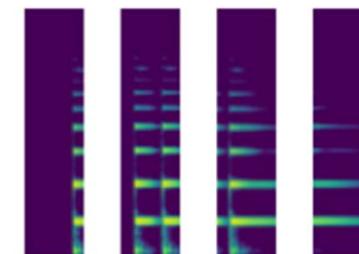


- Song-level vs instance-level

- instance/chunk/clip/segment (various names)



Song-level



Instance-level

Different Classification Tasks

https://music-classification.github.io/tutorial/part1_intro/what-is-music-classification.html

- The most explored music classification tasks in MIR
 - Genre classification [TC02]
 - Mood classification [KSM+10]
 - Instrument identification [HBDP03]
 - Music tagging [Lam08]
- Many others
 - singer/composer classification
 - technique classification
 - audio event detection

Genre/Style Classification

- A conventional category that identifies some pieces of music as belonging to a shared tradition or set of conventions
- Evolve over time

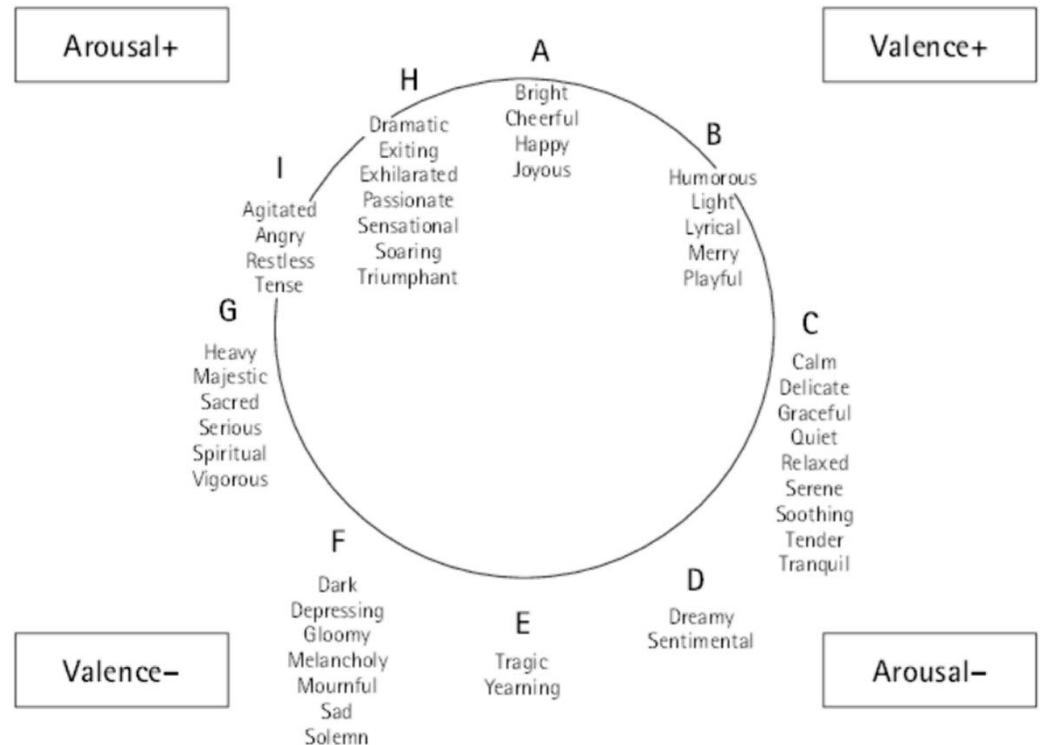


©2003. Translation and redrawing by Frank's Reel Reviews. All rights reserved. www.franksreelreviews.com

Emotion/Mood Classification/Regression

https://github.com/juansgomez87/datasets_emotion

- Perceived vs. felt emotion
- Song-level vs. instance-level
 - music emotion variation detection
- Classification vs. regression
 - **arousal**: energy or neuro-physiological stimulation level
 - **valence**: pleasantness or positive/negative affective states
 - popular taxonomy: 4Qs of the valence/arousal plane
- Inherently subjective

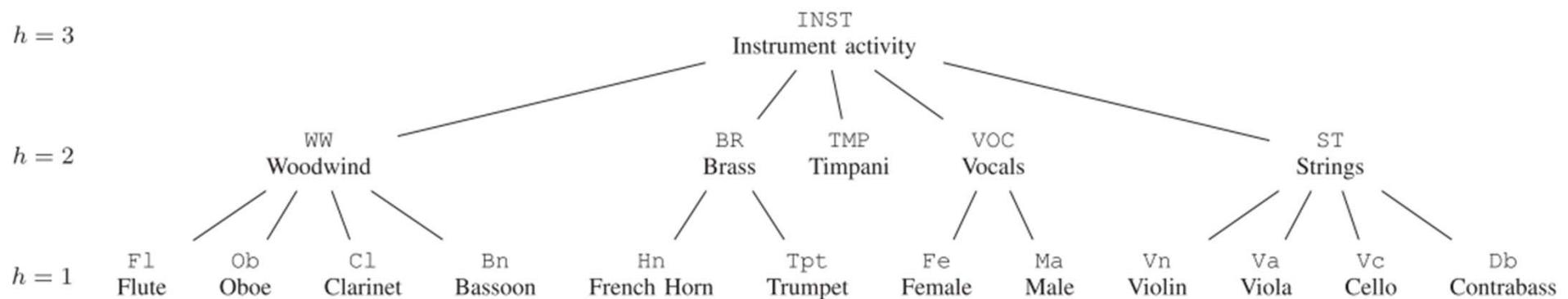
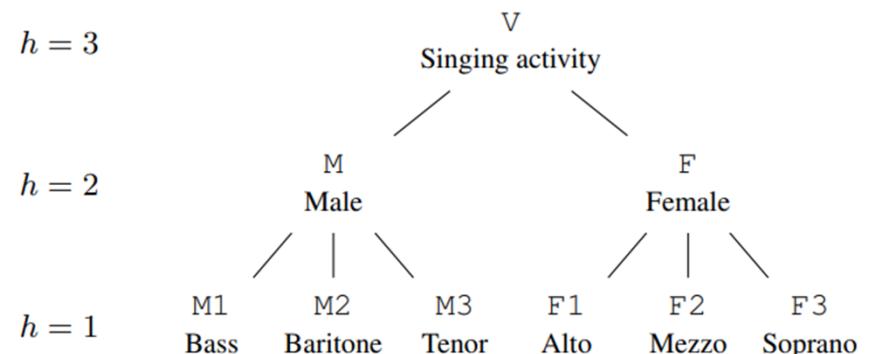


Geneva Emotional Music Scale (GEMS)

<https://musemap.org/resources/gems>

Instrument Classification/Detection

- Song-level vs. instance-level
 - singing activity detection
 - instrument activity detection
- Hierarchical taxonomy



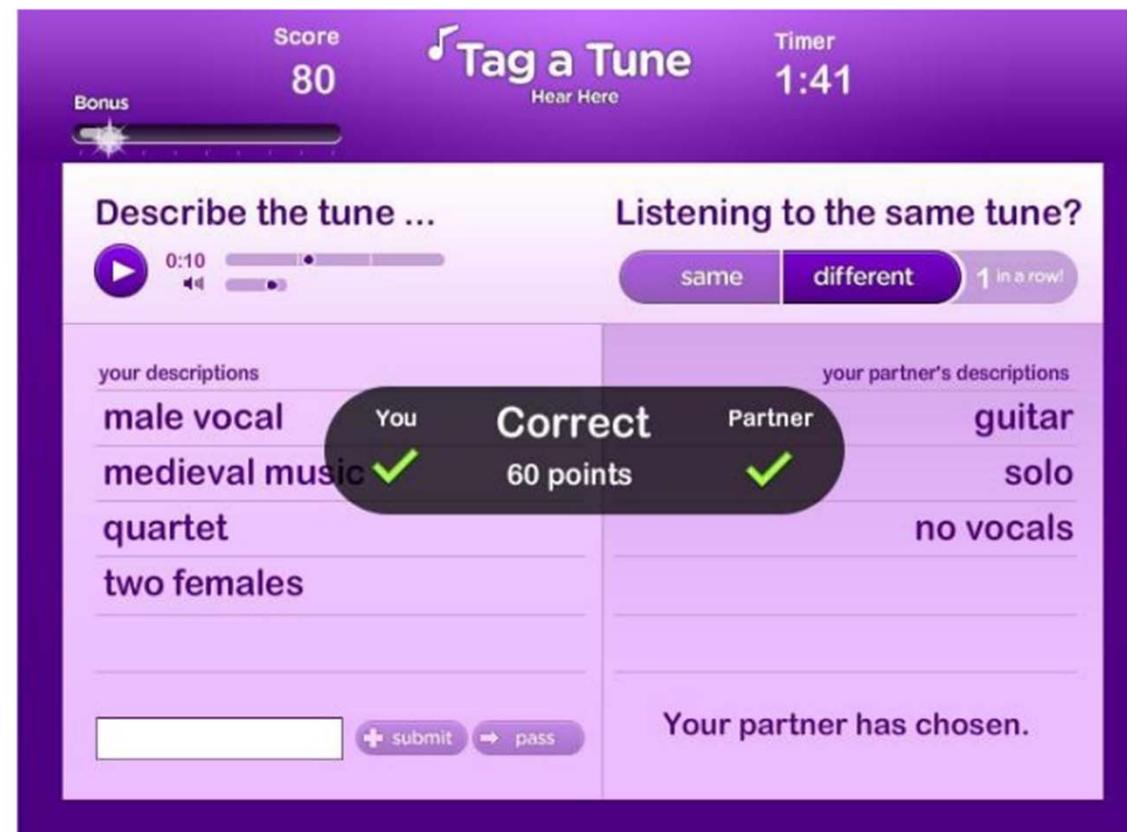
Ref1: Krause et al., “Hierarchical classification of singing activity, gender, and type in complex music recordings,” ICASSP 2022

Ref2: Krause et al., “Hierarchical classification for instrument activity detection in orchestral music recordings,” TASLP 2023

Music Tagging

MagnaTagATune (<https://mirg.city.ac.uk/codeapps/the-magnatagatune-dataset>)

- Top 50 by categories ([source](#))
 - **genre:** classical, techno, electronic, rock, indian, opera, pop, classic, new age, dance, country, metal
 - **instrument:** guitar, strings, drums, piano, violin, vocal, synth, female, male, singing, vocals, no vocals, harpsichord, flute, no vocal, sitar, man, choir, voice, male voice, female vocal, harp, cello, female voice, choral
 - **mood:** slow, fast, ambient, loud, quiet, soft, weird
 - **etc:** beat, solo, beats



Ref: Law et al., "Evaluation of algorithms using games: the case of music annotation," ISMIR 2009

Technique Classification

- Electric guitar
 - bend, vibrato, hammer-on, pull-off, slide
 - <https://zenodo.org/record/1414806>
- Singing voice
 - breathy, vibrato, vocal fry, etc
 - <https://zenodo.org/record/1193957>

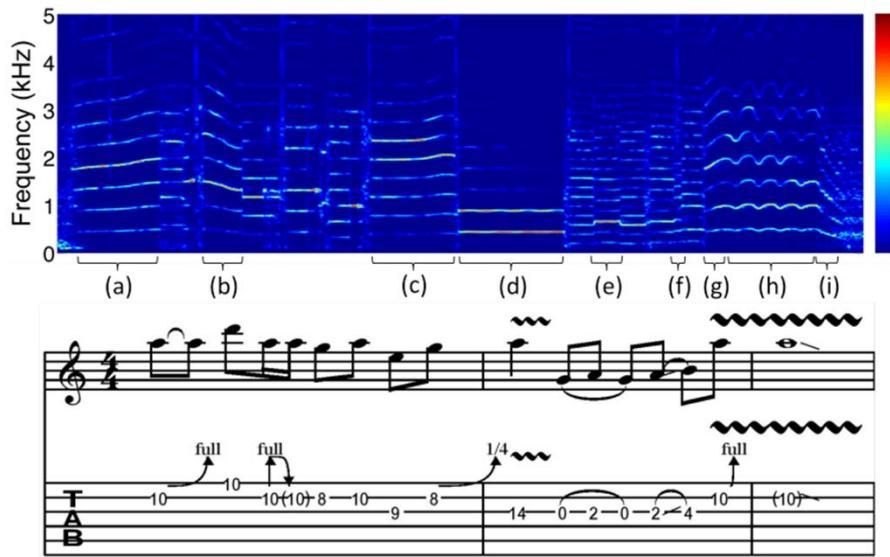
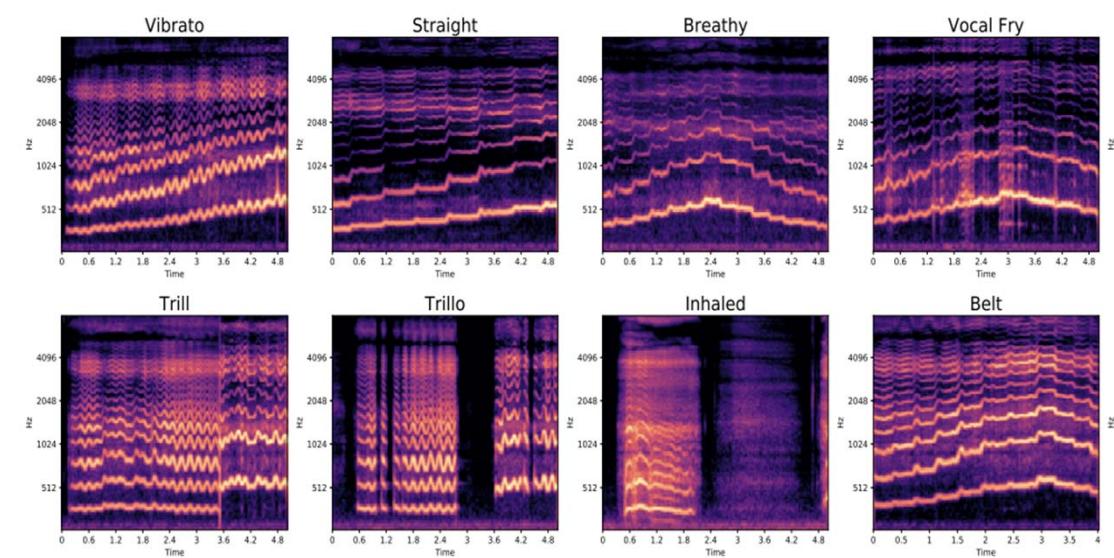
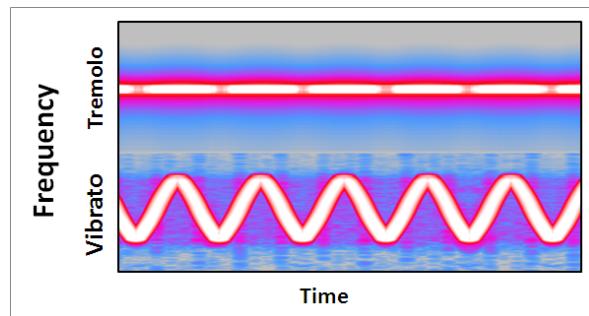


Figure 1. The spectrogram and tablature of a guitar phrase that contains the following techniques: bend (a, b, c, g), vibrato (d, h), hammer-on & pull-off (e) and slide (f, i).



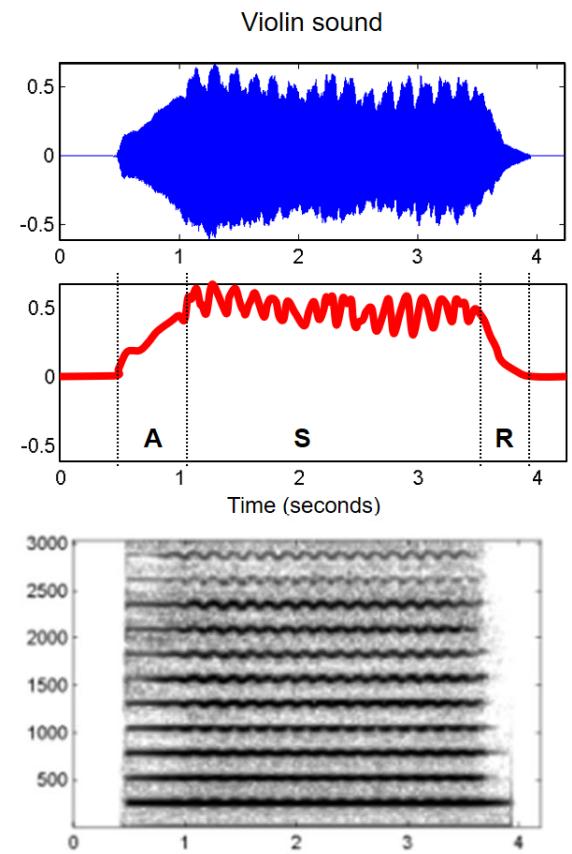
Technique Classification: Vibrato and Tremolo

- **Tremolo:** periodic variations in amplitude (amplitude modulations)
- **Vibrato:** periodic variations in frequency (frequency modulations)
 - Wind and bowed instruments generally use vibratos with an extent of less than half a *semitone* either side
 - Tremolo and vibrato do not necessarily evoke a perceived change in loudness or pitch of the tone



<https://en.wikipedia.org/wiki/Vibrato>

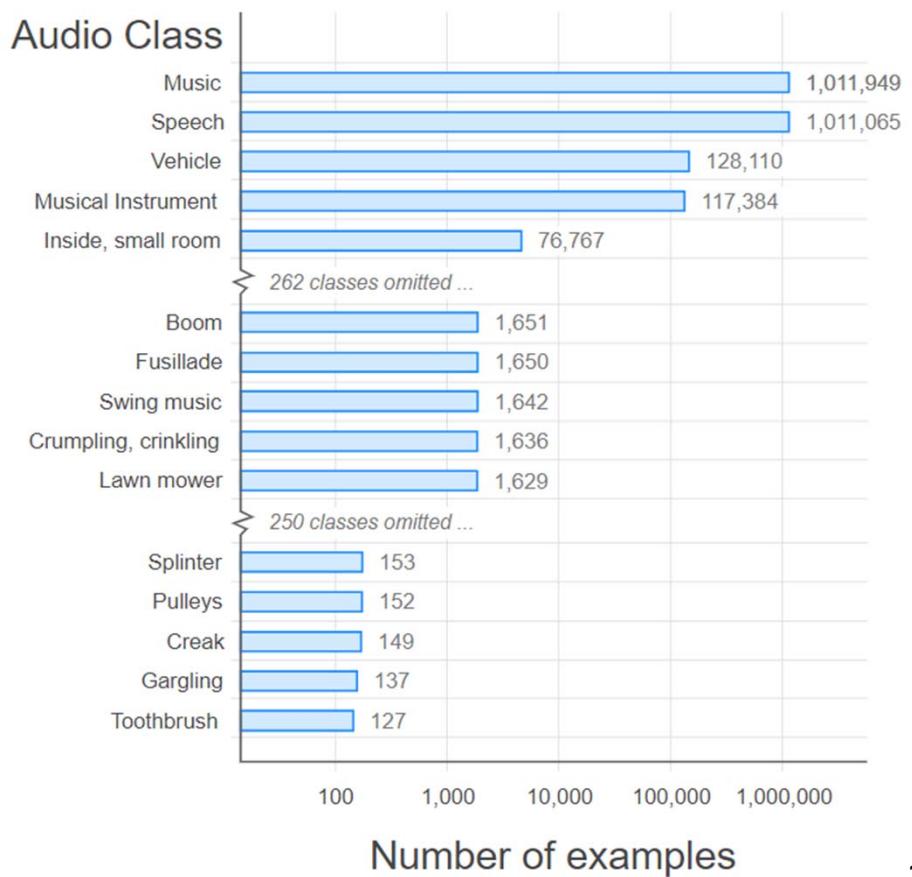
Ref: Sundberg, "Acoustic and psychoacoustic aspects of vocal vibrato," 1994



Audio Event Detection

Audio Set (<http://research.google.com/audioset/>)

- **527** audio classes
 - Over 2M audio clips from YouTube
 - Each 10 second
- Widely used benchmark for **audio classification and audio captioning**
- Can be useful for sound design



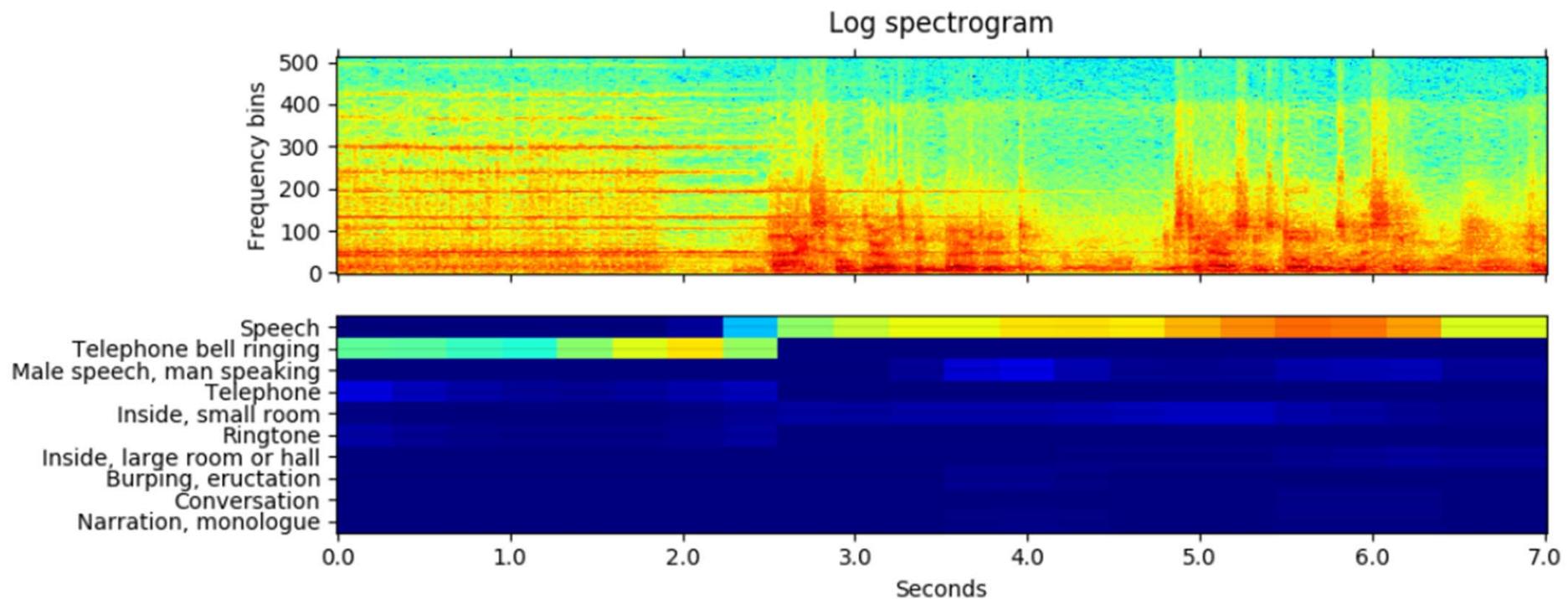
Ref: Gemmeke et al., “Audio Set: An ontology and human-labeled dataset for audio events,” ICASSP 2017

Number of examples

15

Audio Event Detection

https://github.com/qiuqiangkong/audioset_tagging_cnn



- Useful for
 - Music/singing/speech detection; instrument activity detection

Copyright-free Music Audio Datasets

https://music-classification.github.io/tutorial/part2_basics/dataset.html

- Free Music Archive (FMA)
 - <https://freemusicarchive.org/>
 - <https://github.com/mdeff/fma>
- Jamendo dataset
 - <https://www.jamendo.com/>
 - <https://github.com/MTG/mtg-jamendo-dataset>
- FreeSound
 - <https://freesound.org/>
 - <https://labs.freesound.org/datasets/>

MARBLE Benchmark

Music Audio Representation Benchmark for universaL Evaluation

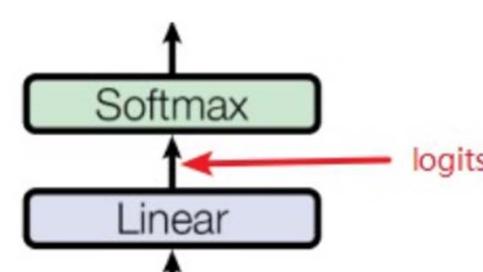
<https://marble-bm.shef.ac.uk/>

Dataset	MTT		GS	GTZAN	GTZAN	EMO		Nsynth	NSynth	VocalSet	VocalSet
Task	Tagging		Key	Genre	Rhythm	Emotion		Instrument	Pitch	Tech	Singer
Metric	ROC	AP	Acc Refined	Acc	F1 ^{beat}	R2 ^V	R2 ^A	Acc	Acc	Acc	Acc
<u>MAP-MERT-v0-95M</u>	90.7	38.2	64.1	74.8	<u>88.3</u>	52.9	69.9	70.4	92.3	73.6	77.0
<u>MAP-MERT-v0-95M-public</u>	90.7	38.4	<u>67.3</u>	72.8	88.1	59.1	72.8	70.4	92.3	75.6	78.0
<u>MAP-MERT-v1-95M</u>	91.0	39.3	63.5	74.8	<u>88.3</u>	55.5	<u>76.3</u>	70.7	92.6	74.2	83.7
<u>MAP-MERT-v1-330M</u>	91.1	39.5	61.7	77.6	87.9	59.0	75.8	72.6	<u>94.4</u>	<u>76.9</u>	87.1
<u>MAP-Music2Vec</u>	90.0	36.2	50.6	74.1	68.2	52.1	71.0	69.3	93.1	71.1	81.4
<u>MusiCNN</u>	90.3	37.8	14.4	73.5	-	44.0	68.8	72.6	64.1	70.3	57.0
<u>CLMR</u>	89.5	36.0	14.8	65.2	-	44.4	70.3	67.9	47.0	58.1	49.9
<u>Jukebox-5B</u>	<u>91.4</u>	<u>40.6</u>	63.8	<u>77.9</u>	-	57.0	73.0	70.4	91.6	76.7	82.6
<u>MULE</u>	91.2	40.1	64.9	75.5	-	<u>60.7</u>	73.1	<u>74.6</u>	88.5	75.5	<u>87.5</u>

Evaluation Metrics for Music Classification

- The output of DL-based classifiers are usually probabilities

- multi-class classification: **softmax**
- multi-label classification: **sigmoid**



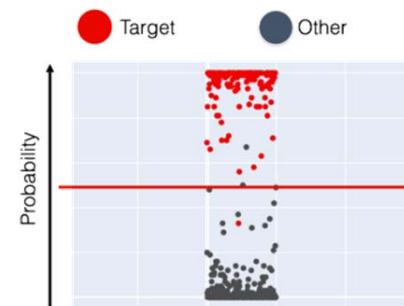
S	T
0.775	1
0.116	0
0.039	0
0.070	0

A double-headed yellow arrow labeled $L_{CE}(S, T)$ connects the two tables.

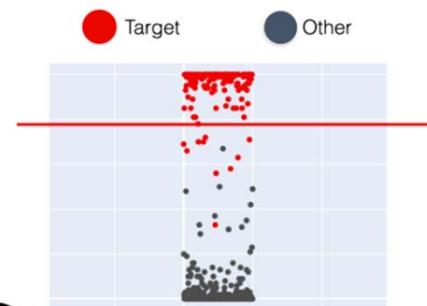
- Probability vs decision

- outputting probabilities is fine at training time
- but, at inference time, need to “**make decisions**”
- usually by thresholding

Probability > 50%



Probability > 80%

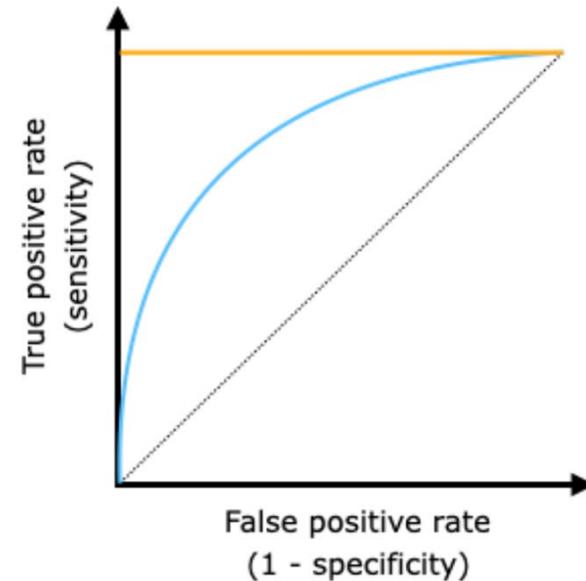
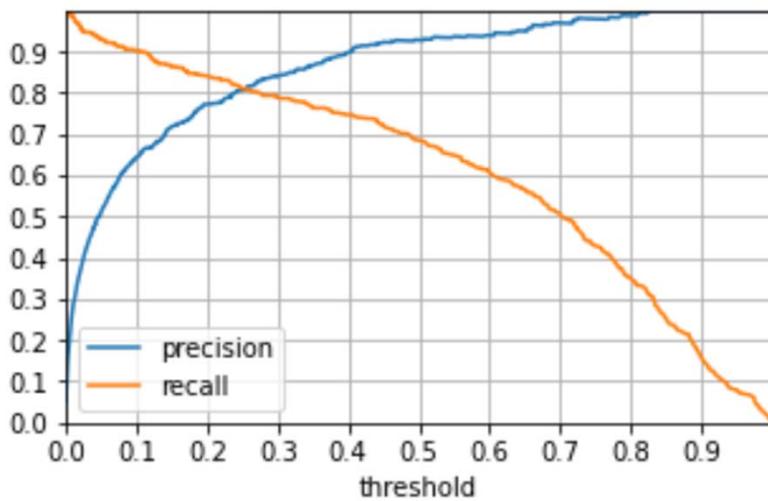


Evaluation Metrics for Music Classification

https://music-classification.github.io/tutorial/part2_basics/evaluation.html

- Classification accuracy (top1, top3)
- Precision, recall, Fscore
 - obtained by varying the threshold
- ROC-AUC
 - “micro” vs “macro” average

average : {'micro', 'macro', 'samples', 'weighted'}



Outline

- **Music classification**
 - Basics: Tasks, datasets, and evaluation
 - **Supervised models**
 - Self-supervised models
- Music transcription
- HW1

Exemplar Models

<https://github.com/minzwon/sota-music-tagging-models>

Available Models

- **FCN** : Automatic Tagging using Deep Convolutional Neural Networks, Choi et al., 2016 [[arxiv](#)]
- **Musicnn** : End-to-end Learning for Music Audio Tagging at Scale, Pons et al., 2018 [[arxiv](#)]
- **Sample-level CNN** : Sample-level Deep Convolutional Neural Networks for Music Auto-tagging Using Raw Waveforms, Lee et al., 2017 [[arxiv](#)]
- **Sample-level CNN + Squeeze-and-excitation** : Sample-level CNN Architectures for Music Auto-tagging Using Raw Waveforms, Kim et al., 2018 [[arxiv](#)]
- **CRNN** : Convolutional Recurrent Neural Networks for Music Classification, Choi et al., 2016 [[arxiv](#)]
- **Self-attention** : Toward Interpretable Music Tagging with Self-Attention, Won et al., 2019 [[arxiv](#)]
- **Harmonic CNN** : Data-Driven Harmonic Filters for Audio Representation Learning, Won et al., 2020 [[pdf](#)]
- **Short-chunk CNN** : Prevalent 3x3 CNN. So-called *vgg*-ish model with a small receptive field.
- **Short-chunk CNN + Residual** : Short-chunk CNN with residual connections.

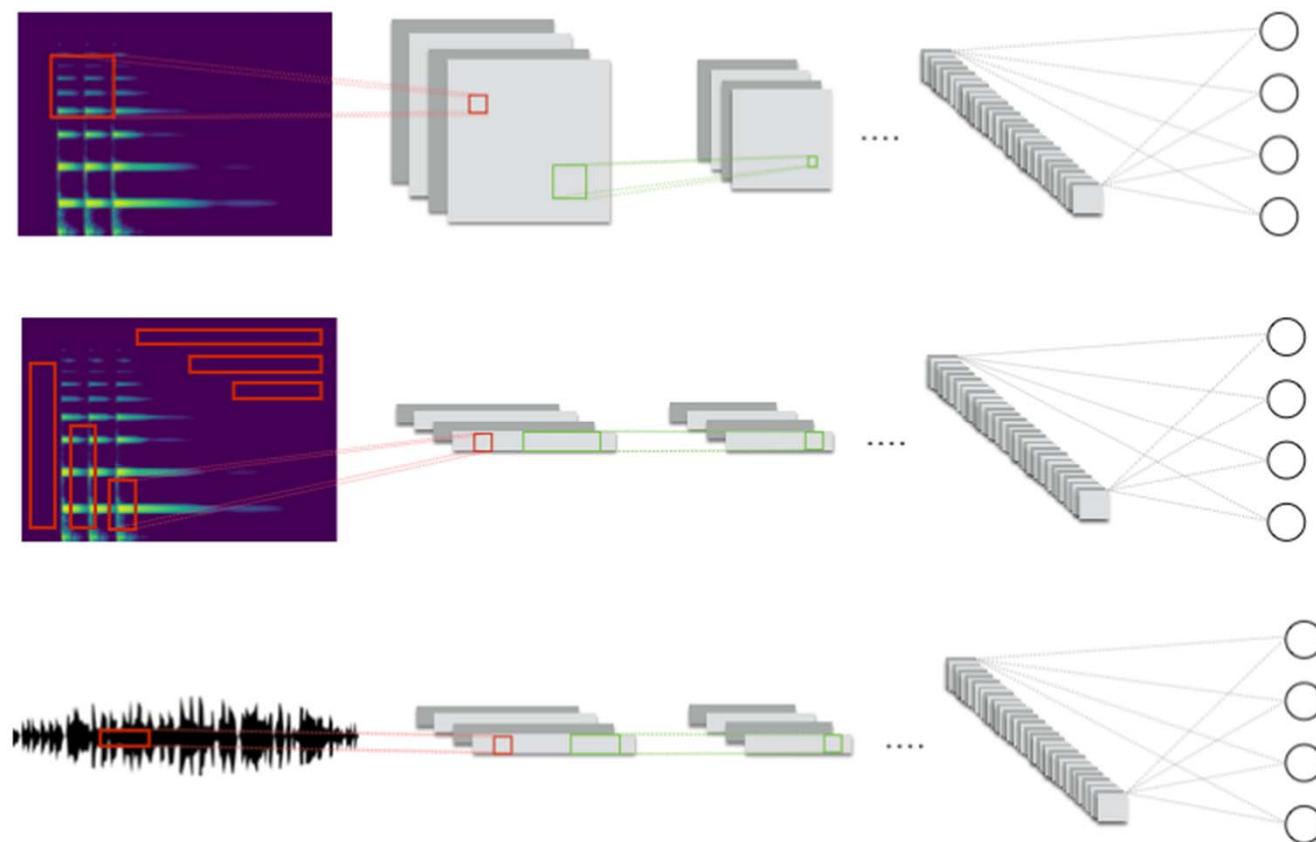
Exemplar Models

https://music-classification.github.io/tutorial/part3_supervised/architectures.html

Model	Preprocessing	Input length	Front end	Back end	Training	Aggregation
FCN	STFT	29.1s	2D CNN	.	song-level	.
VGG-ish / Short-chunk CNN	STFT	3.96s	2D CNN	Global max pooling	instance-level	Average
Harmonic CNN	STFT	5s	2D CNN	Global max pooling	instance-level	Average
MusiCNN	STFT	3s	2D CNN	1D CNN	instance-level	Average
Sample-level CNN	.	3s	1D CNN	1D CNN	instance-level	Average
CRNN	STFT	29.1s	2D CNN	RNN	song-level	.
Music tagging transformer	STFT	5s-30s	2D CNN	Transformer	instance-level	Average

Exemplar Models

https://music-classification.github.io/tutorial/part3_supervised/architectures.html



Audio Data Augmentations

https://music-classification.github.io/tutorial/part3_supervised/architectures.html

Name	Author	Framework	Language	License	Link
Muda	B. McFee et al. (2015)	General Purpose	Python	ISC License	source code
Audio Degradation Toolbox	M. Mauch et al. (2013)	General Purpose	MATLAB	GNU General Public License 2.0	source code
rubberband	-	General Purpose	C++	GNU General Public License (non-commercial)	website , pyrubberband
torchaudio	pytorch.org	PyTorch	Python	BSD 2-Clause "Simplified" License	source code

Audio Degradation Toolbox

<https://github.com/sevagh/audio-degradation-toolbox>

- Exemplar use case: simulate the case of smartphone recording

```
{ "name": "noise", ["snr": 20, "color": "pink"] }
{ "name": "mp3", ["bitrate": 320] }
{ "name": "gain", ["volume": 10.0] }
{ "name": "normalize" }
{ "name": "low_pass", ["cutoff": 1000.0] }
{ "name": "high_pass", ["cutoff": 1000.0] }
{ "name": "trim_millis", ["amount": 100, "offset": 0] }
{ "name": "mix", "path": STRING, ["snr": 20.0] }
{ "name": "speedup", "speed": FLOAT }
{ "name": "resample", "rate": INT }
{ "name": "pitch_shift", "octaves": FLOAT }
{ "name": "dynamic_range_compression", ["threshold": -20.0, "ratio": 4.0, "attack": 5.0, "release": 50.0] }
{ "name": "impulse_response", "path": STRING }
{ "name": "equalizer", "frequency": FLOAT, ["bandwidth": 1.0, "gain": -3.0] }
{ "name": "time_stretch", "factor": FLOAT }
{ "name": "delay", "n_samples": INT }
{ "name": "clipping", ["n_samples": 0, "percent_samples": 0.0] }
{ "name": "wow_flutter", ["intensity": 1.5, "frequency": 0.5, "upsampling_factor": 5.0 ] }
{ "name": "aliasing", ["dest_frequency": 8000.0] }
```

torchaudio_augmentations

https://pytorch.org/audio/stable/tutorials/audio_data_augmentation_tutorial.html

https://music-classification.github.io/tutorial/part3_supervised/tutorial.html

```
from torchaudio_augmentations import (
    RandomResizedCrop,
    RandomApply,
    PolarityInversion,
    Noise,
    Gain,
    HighLowPass,
    Delay,
    PitchShift,
    Reverb,
    Compose,
)
```

pyrubberband

<https://github.com/bmcfee/pyrubberband>

```
>>> import soundfile as sf
>>> import pyrubberband as pyrb
>>> y, sr = sf.read("myfile.wav")
>>> # Play back at double speed
>>> y_stretch = pyrb.time_stretch(y, sr, 2.0)
>>> # Play back two semi-tones higher
>>> y_shift = pyrb.pitch_shift(y, sr, 2)
```

- **Time stretch:** make it faster/slower without changing the pitch
- **Pitch shift**

Sample Code

https://music-classification.github.io/tutorial/part3_supervised/tutorial.html

```
# convolutional layers
self.layer1 = Conv_2d(1, num_channels, pooling=(2, 3))
self.layer2 = Conv_2d(num_channels, num_channels, pooling=(3, 4))
self.layer3 = Conv_2d(num_channels, num_channels * 2, pooling=(2, 5))
self.layer4 = Conv_2d(num_channels * 2, num_channels * 2, pooling=(3, 3))
self.layer5 = Conv_2d(num_channels * 2, num_channels * 4, pooling=(3, 4))

# dense layers
self.dense1 = nn.Linear(num_channels * 4, num_channels * 4)
self.dense_bn = nn.BatchNorm1d(num_channels * 4)
self.dense2 = nn.Linear(num_channels * 4, num_classes)
self.dropout = nn.Dropout(0.5)
self.relu = nn.ReLU()

# reshape and aggregate chunk-level predictions
b, c, t = wav.size()
logits = cnn(wav.view(-1, t))
logits = logits.view(b, c, -1).mean(dim=1)
_, pred = torch.max(logits.data, 1)
```

Exemplar Model: PANNs

https://github.com/qiuqiangkong/audioset_tagging_cnn

- From sound event detection
- Purely convolutional
- Can be used as a pre-trained model
 - Produce audio embeddings that have been used by CLAP (<https://github.com/LAION-AI/CLAP>) in learning audio-text joint embedding space for **text-to-audio** generation

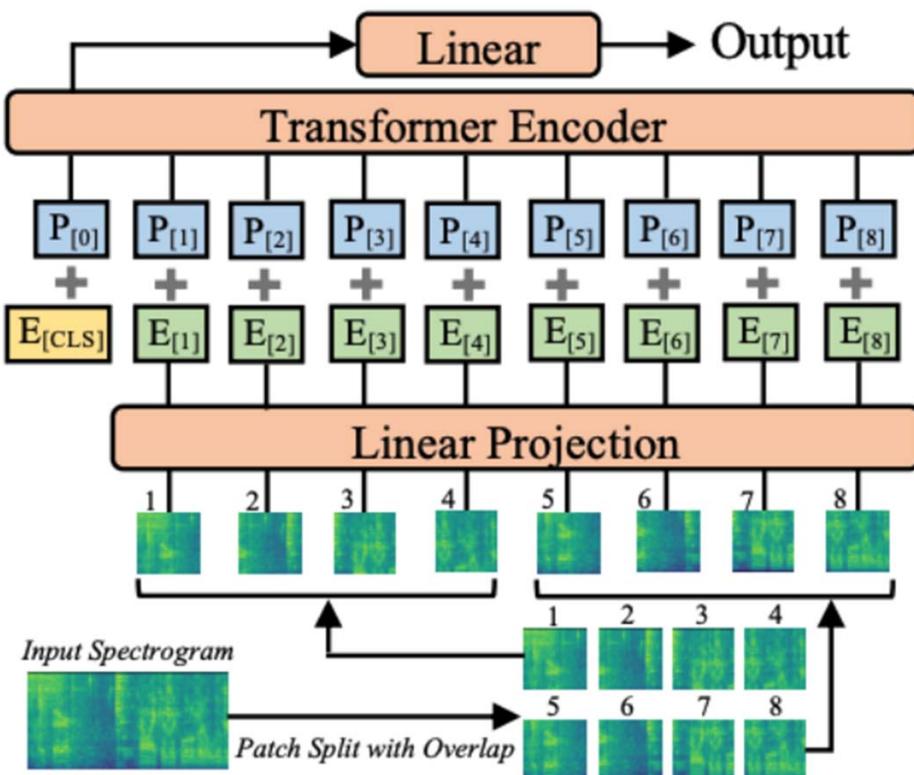
VGGish [1]	CNN6	CNN10	CNN14
Log-mel spectrogram 96 frames \times 64 mel bins	Log-mel spectrogram 1000 frames \times 64 mel bins		
$3 \times 3 @ 64$ ReLU	$5 \times 5 @ 64$ BN, ReLU	$(3 \times 3 @ 64) \times 2$ BN, ReLU	$(3 \times 3 @ 64) \times 2$ BN, ReLU
MP 2×2	Pooling 2×2		
$3 \times 3 @ 128$ ReLU	$5 \times 5 @ 128$ BN, ReLU	$(3 \times 3 @ 128) \times 2$ BN, ReLU	$(3 \times 3 @ 128) \times 2$ BN, ReLU
MP 2×2	Pooling 2×2		
$(3 \times 3 @ 256) \times 2$ ReLU	$5 \times 5 @ 256$ BN, ReLU	$(3 \times 3 @ 256) \times 2$ BN, ReLU	$(3 \times 3 @ 256) \times 2$ BN, ReLU
MP 2×2	Pooling 2×2		
$(3 \times 3 @ 512) \times 2$ ReLU	$5 \times 5 @ 512$ BN, ReLU	$(3 \times 3 @ 512) \times 2$ BN, ReLU	$(3 \times 3 @ 512) \times 2$ BN, ReLU
MP 2×2 Flatten	Global pooling		Pooling 2×2
FC 4096×2 ReLU	FC 512, ReLU		$(3 \times 3 @ 1024) \times 2$ BN, ReLU
FC 527, Sigmoid	FC 527, Sigmoid		Pooling 2×2
	$(3 \times 3 @ 2048) \times 2$ BN, ReLU		Global pooling
	FC 2048, ReLU		FC 2048, ReLU
	FC 527, Sigmoid		FC 527, Sigmoid

Ref: Kong et al., “PANNs: Large-scale pretrained audio neural networks for audio pattern recognition,” arXiv 2020 30

Exemplar Model: Audio Spectrogram Transformer

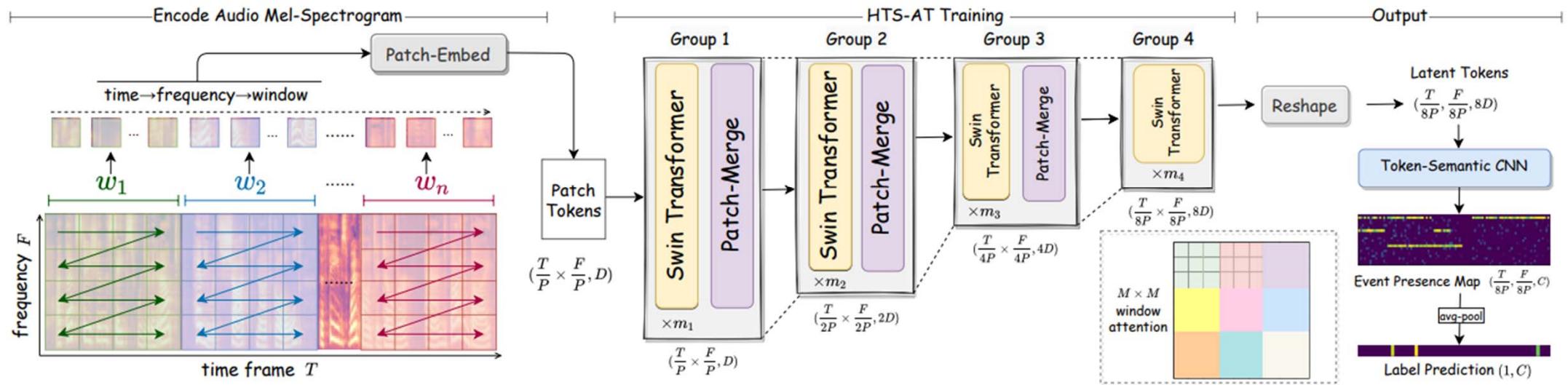
<https://github.com/YuanGongND/ast>

- From sound event detection
- Use Vision **Transformer** (ViT) based architecture
 - The first convolution-free, purely attention-based model for audio classification
- May need larger amount of training data and compute



Ref: Gong et al., “AST: Audio Spectrogram Transformer,” INTERSPEECH 2021

Exemplar Model: HTS-AT



Ref: Chen et al., "HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection," ICASSP 2022

Ref: Wu et al., "Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation," arXiv 2022

Model	AudioCaps (mAP@10)		Cloho (mAP@10)	
	A→T	T→A	A→T	T→A
PANN+CLIP Trans.	4.7	11.7	1.9	4.4
PANN+BERT	34.3	44.3	10.8	17.7
PANN+RoBERTa	37.5	45.3	11.3	18.4
HTSAT+CLIP Trans.	2.4	6.0	1.1	3.2
HTSAT+BERT	43.7	49.2	13.8	20.8
HTSAT+RoBERTa	45.7	51.3	13.8	20.4

Table 2: The text-to-audio retrieval result (mAP@10) of using different audio/text encoder on AudioCaps and Cloho.

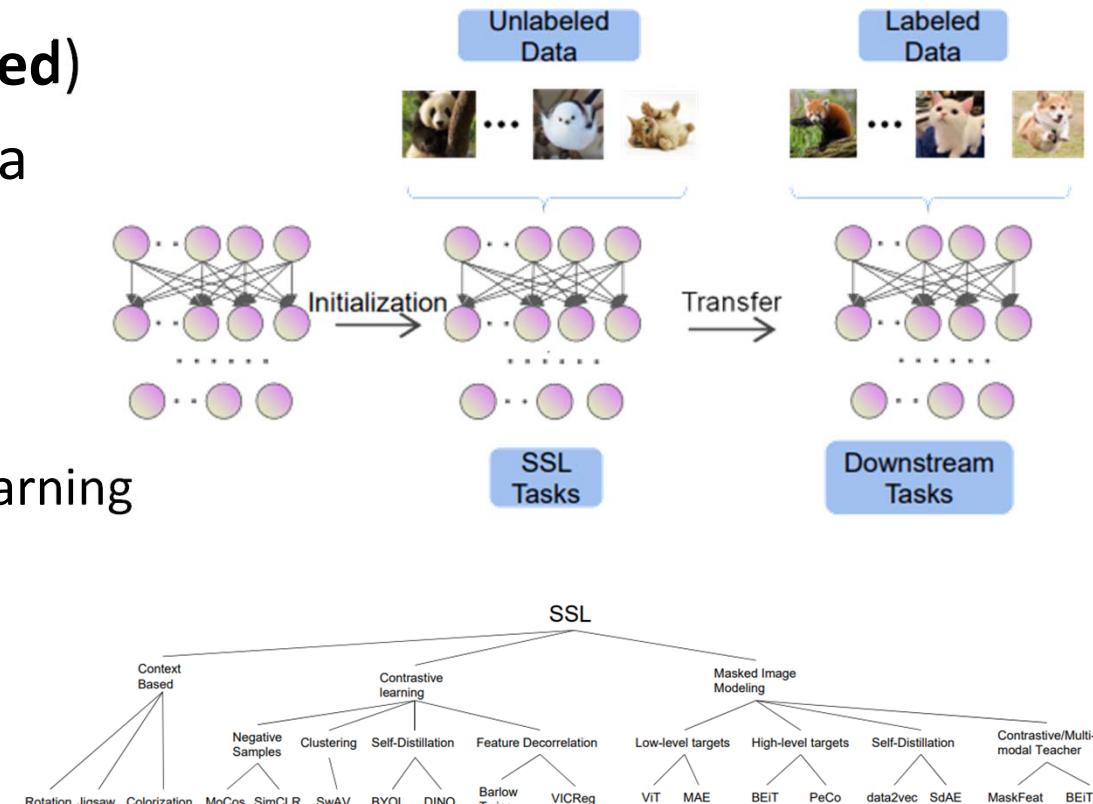
Outline

- **Music classification**
 - Basics: Tasks, datasets, and evaluation
 - Supervised models
 - **Self-supervised models**
- Music transcription
- HW1

Self-Supervised Learning (SSL)

https://music-classification.github.io/tutorial/part5_beyond/introduction.html

- **Do not** need human labels (**unlabeled**)
- Can learn from **large** amount of data
- Can do “**transfer learning**”
 - **Pre-train** the model with SSL
 - **Fine-tune** with a few more layers on downstream tasks using supervised learning
- Famous methods
 - **Contrastive learning** (e.g., SimCLR)
 - **Masked modeling** (e.g., BERT)

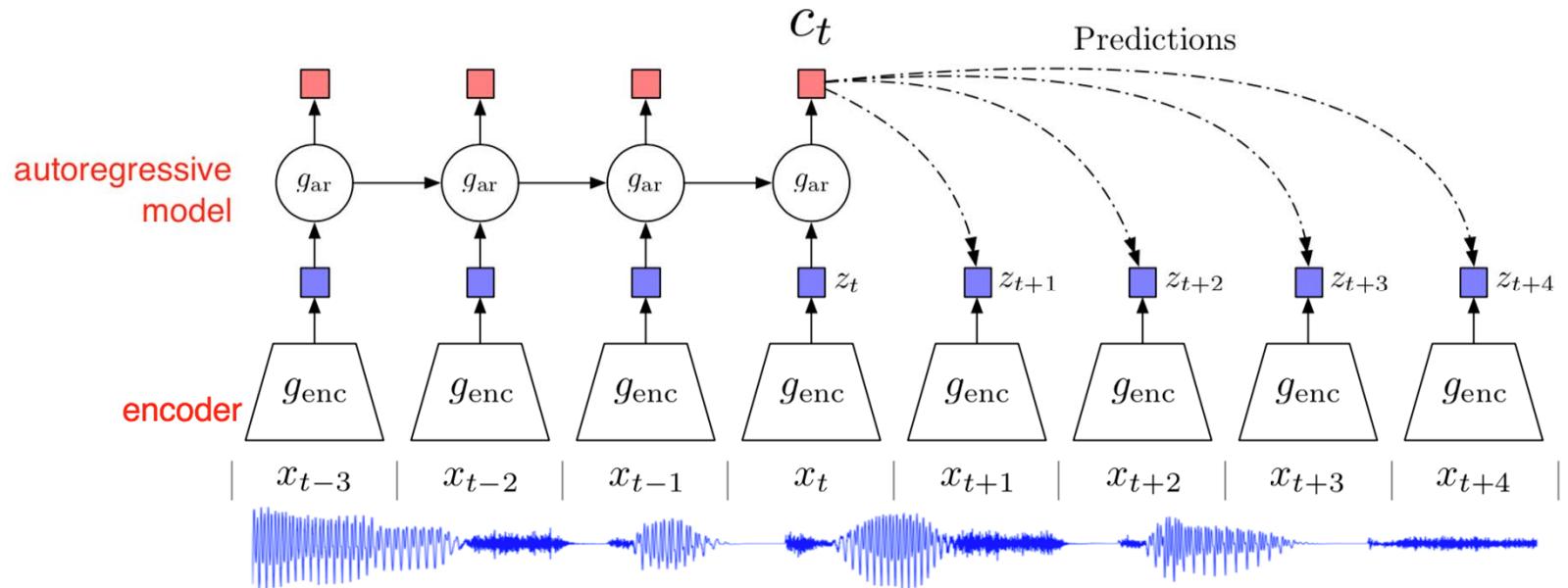


Gui et al., “A survey on self-supervised learning: Algorithms, applications, and future trends,” TPAMI 2024

Contrastive Predictive Coding

https://music-classification.github.io/tutorial/part5_beyond/methods.html

- Learn by predicting the future in a learned latent space

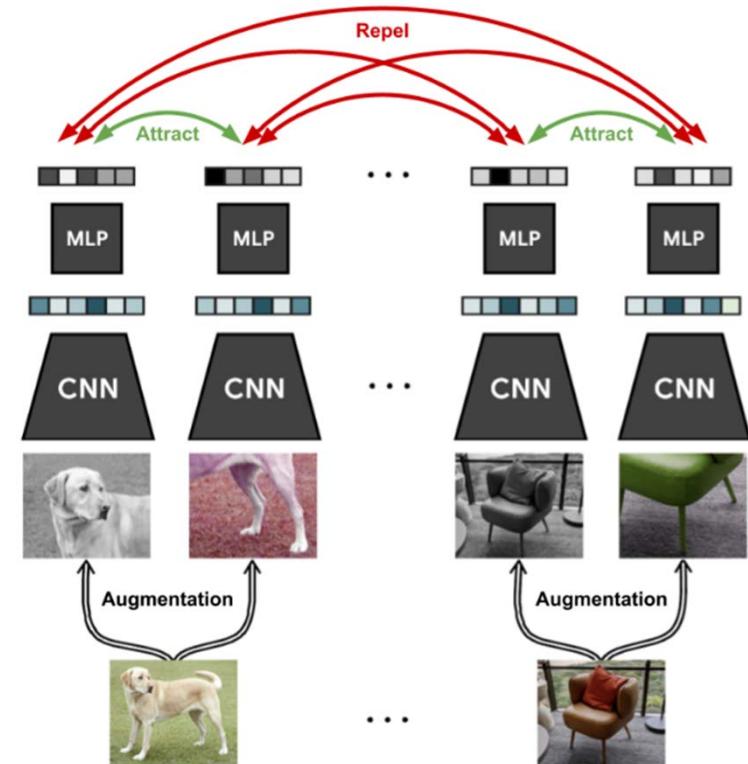


Ref: van den Oord et al., “Representation learning with contrastive predictive coding,” arXiv 2018

SimCLR

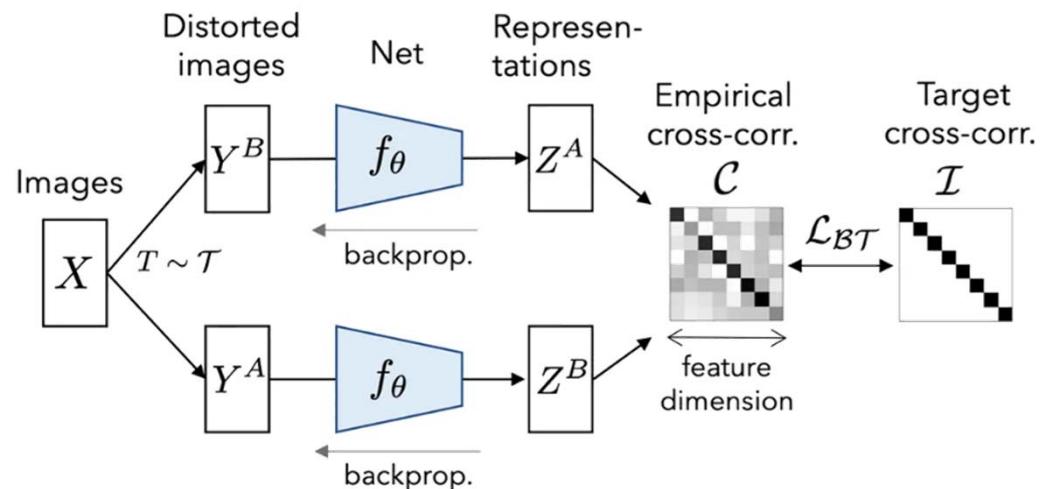
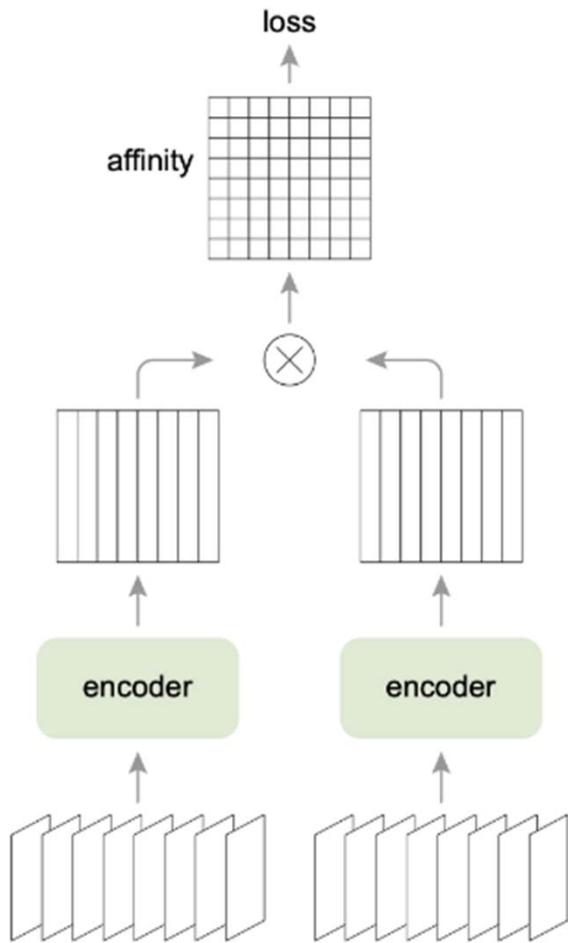
https://music-classification.github.io/tutorial/part5_beyond/methods.html

- “Draw closer” positive data pairs while “push away” negative data pairs
 - **Positive pairs:** different “augmented views” of the same instance (done by *data augmentation*)
 - **Negative pairs:** different instances
- **Metric learning**
 - shared encoder (CNN+MLP), output an *embedding vector*
 - learn a similarity metric discriminatively



Ref: Chen et al., “A simple framework for contrastive learning of visual representations,” ICML 2020

SimCLR

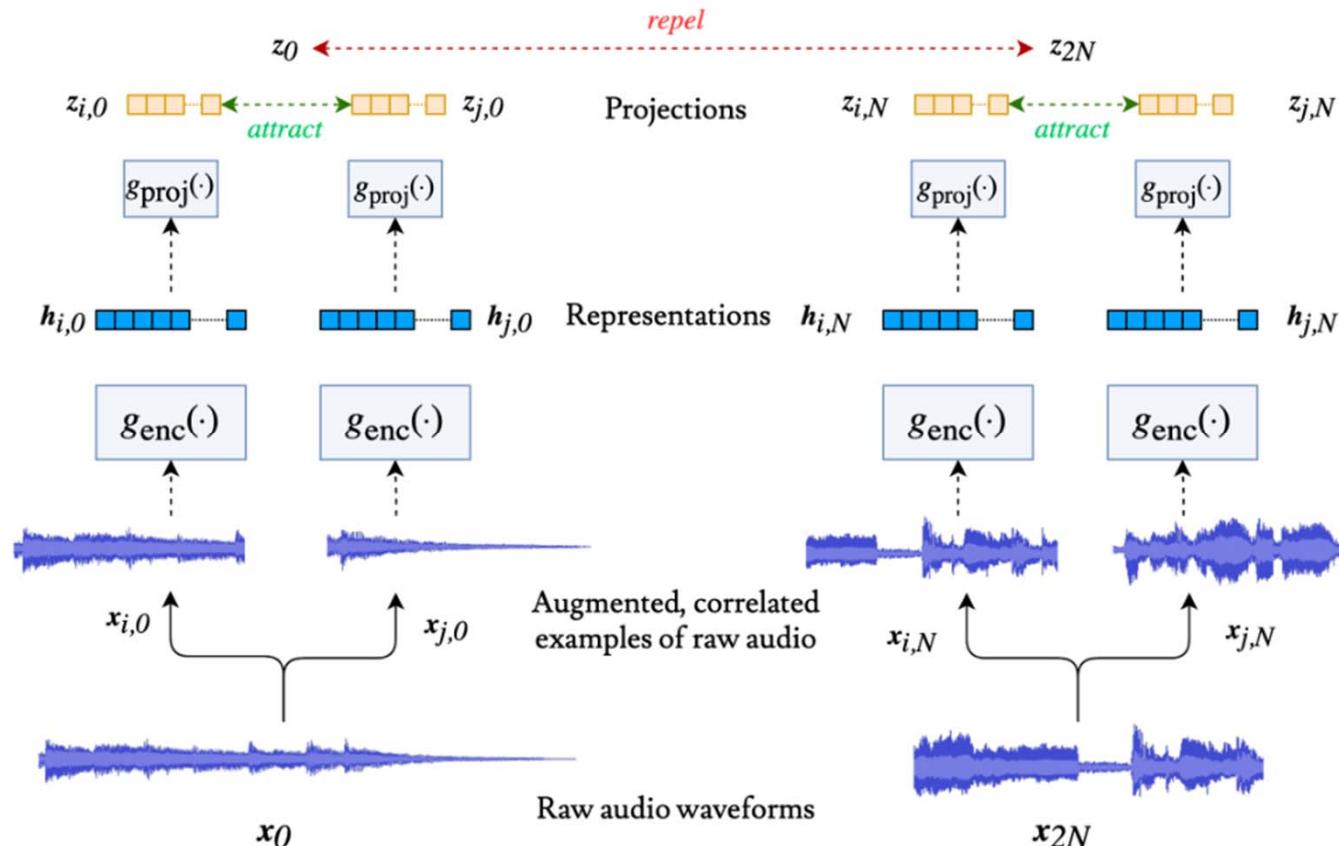


- Training details
 - given a mini-batch of N samples, we will have $2N$ samples after data augmentation
 - given a positive pair, view the other $2(N-1)$ pairs as negative
 - infoNCE loss

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$

Exemplar Model: CLMR

https://music-classification.github.io/tutorial/part5_beyond/self-supervised-learning.html

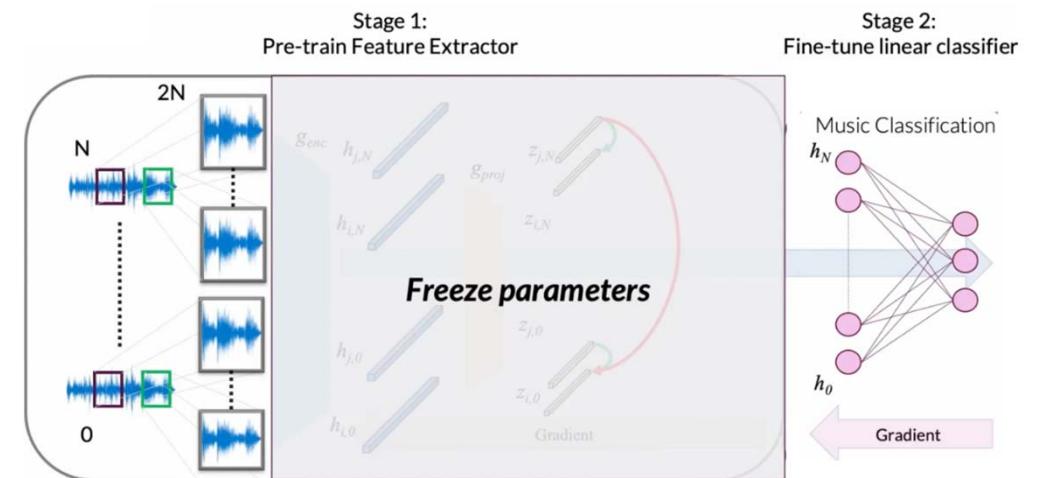
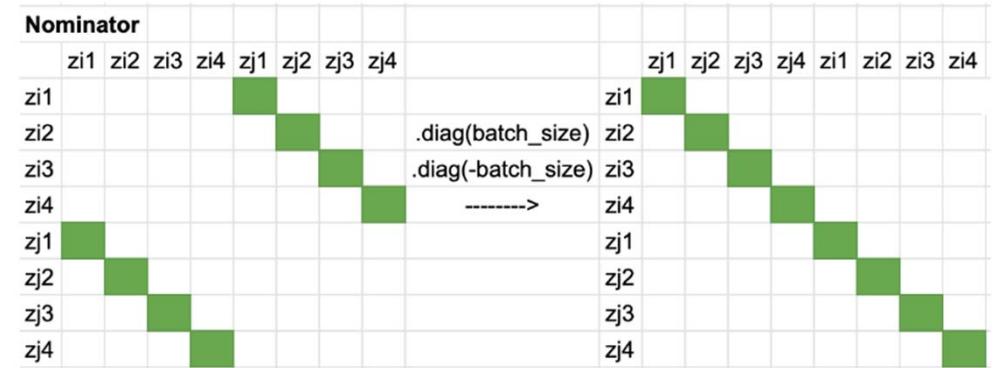


Ref: Spijkervet & Burgoyne, "Contrastive Learning of Musical Representations," ISMIR 2021

Sample Code

https://music-classification.github.io/tutorial/part5_beyond/self-supervised-learning.html

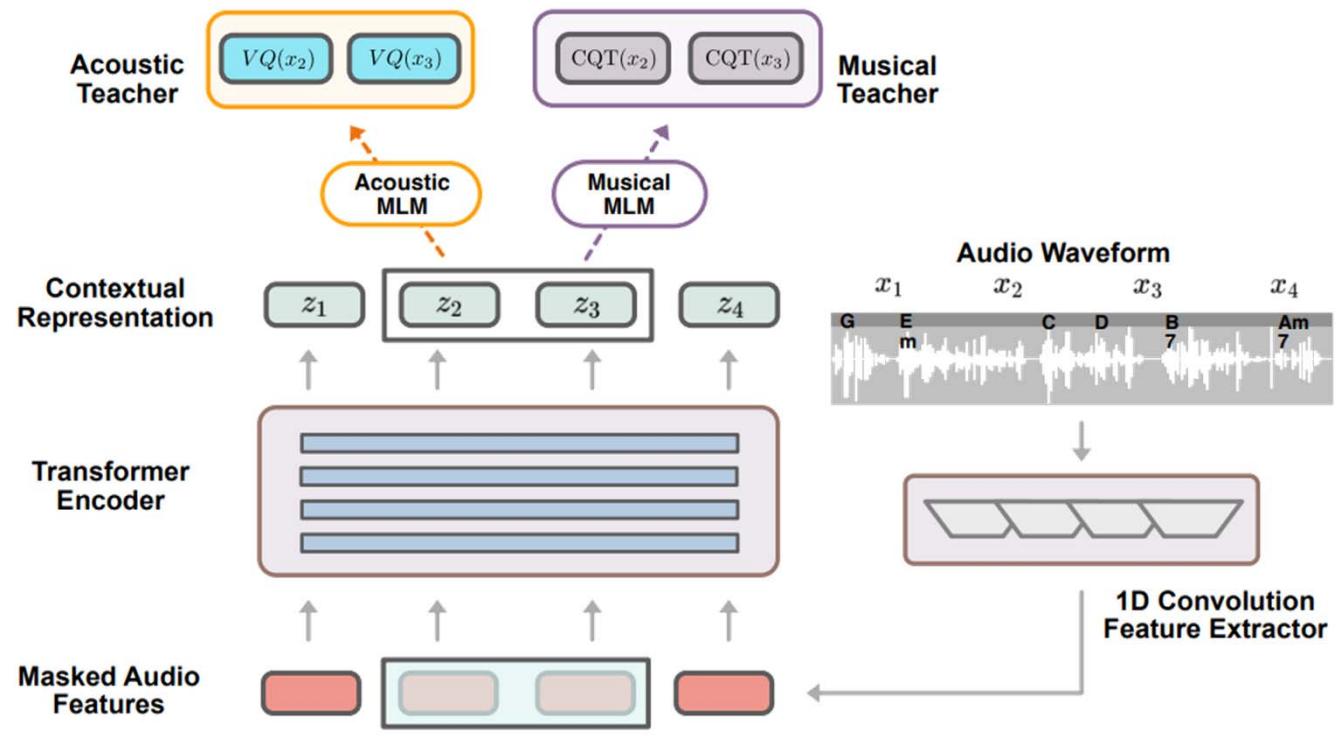
- Use **sample-level CNN** as the feature extractor
 - 59,049 (3^{10}) samples at 22kHz (~2.68 seconds)
- Use SimCLR for learning
 - `torchaudio_augmentations`
- Freeze the pre-trained feature extractor, fine-tune linear classifier
 - supervised SampleCNN: 49.6%
 - SSL CLMR + SampleCNN: 55.2%



Exemplar Model: MERT

<https://github.com/yizhilll/MERT>

- Use **sample-level CNNs**
 - 5-second clips as input
 - 75 embeddings per second
- Masked language modeling
 - acoustic teachers (\mathcal{L}_H)
 - predict HuBERT or Encodenc
 - musical teachers (\mathcal{L}_{CQT})
 - predict CQT
 - do both



Ref: Li et al., "MERT: Acoustic music understanding model with large-scale self-supervised training," ICLR 2024

MERT Outperforms CLMR

<https://marble-bm.shef.ac.uk/>

Dataset	MTT		GS	GTZAN	GTZAN	EMO		Nsynth	NSynth	VocalSet	VocalSet
Task	Tagging		Key	Genre	Rhythm	Emotion		Instrument	Pitch	Tech	Singer
Metric	ROC	AP	Acc Refined	Acc	F1 ^{beat}	R2 ^V	R2 ^A	Acc	Acc	Acc	Acc
<u>MAP-MERT-v0-95M</u>	90.7	38.2	64.1	74.8	<u>88.3</u>	52.9	69.9	70.4	92.3	73.6	77.0
<u>MAP-MERT-v0-95M-public</u>	90.7	38.4	<u>67.3</u>	72.8	88.1	59.1	72.8	70.4	92.3	75.6	78.0
<u>MAP-MERT-v1-95M</u>	91.0	39.3	63.5	74.8	<u>88.3</u>	55.5	<u>76.3</u>	70.7	92.6	74.2	83.7
<u>MAP-MERT-v1-330M</u>	91.1	39.5	61.7	77.6	87.9	59.0	75.8	72.6	<u>94.4</u>	<u>76.9</u>	87.1
<u>MAP-Music2Vec</u>	90.0	36.2	50.6	74.1	68.2	52.1	71.0	69.3	93.1	71.1	81.4
<u>MusiCNN</u>	90.3	37.8	14.4	73.5	-	44.0	68.8	72.6	64.1	70.3	57.0
<u>CLMR</u>	89.5	36.0	14.8	65.2	-	44.4	70.3	67.9	47.0	58.1	49.9
<u>Jukebox-5B</u>	<u>91.4</u>	<u>40.6</u>	63.8	<u>77.9</u>	-	57.0	73.0	70.4	91.6	76.7	82.6
<u>MULE</u>	91.2	40.1	64.9	75.5	-	<u>60.7</u>	73.1	<u>74.6</u>	88.5	75.5	<u>87.5</u>

Outline

- Music classification
- **Music transcription**
- HW1

Reference 3: ISMIR 2018 & 2021 Tutorials

<https://rachelbittner.weebly.com/other-resources.html>

Tutorials

Programming MIR Baselines from Scratch: Three Case Studies

2021

International Society for Music Information Retrieval (ISMIR) conference

- Part 1: Transcription with NMF (Ethan Manilow)
- Part 2: Pitch Tracking with pytorch (**Rachel Bittner**)
- Part 3: Instrument Classification with OpenL3 & Tensorflow (Mark Cartwright)



See the recording here.

Fundamental Frequency Estimation in Music

2018

International Society for Music Information Retrieval (ISMIR) conference

- Part 1: Pitch (Alain de Cheveigné)
- Part 2: Polyphonic fundamental frequency estimation (**Rachel Bittner**)
- Part 3: Applications (Johana Devaney)

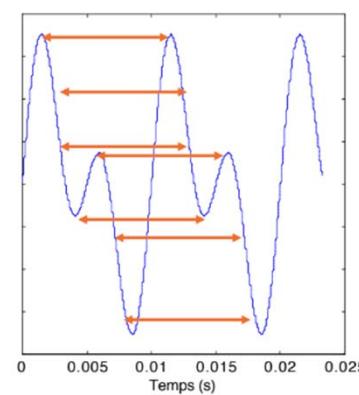
Monophonic F0 Estimation

- Part 1: Pitch (Alain de Cheveigné)

<https://drive.google.com/file/d/1uWCwE03dM0j0Ptp1g5vPsT56nPd0DCW2/view>

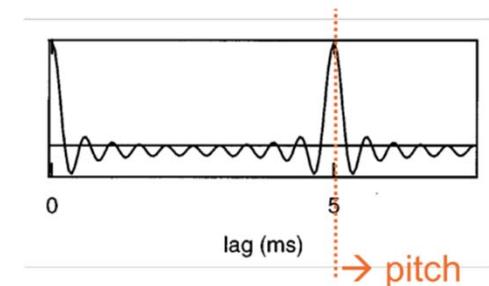
- Pitch: depends on fundamental frequency (F0), not on shape
- Frequency-domain F0 analysis
 - problems: harmonics can be stronger, missing fundamental, etc
- Time-domain F0 analysis
(based on **auto-correlation**)
 - YIN (2002), pYIN (2014)

```
librosa.pyin(y, *, fmin, fmax, sr=22050,
```



$$A_t(\tau) = \sum_{W}^{''shift and compare''} x(t)x(t - \tau)$$

product time lag
integration window

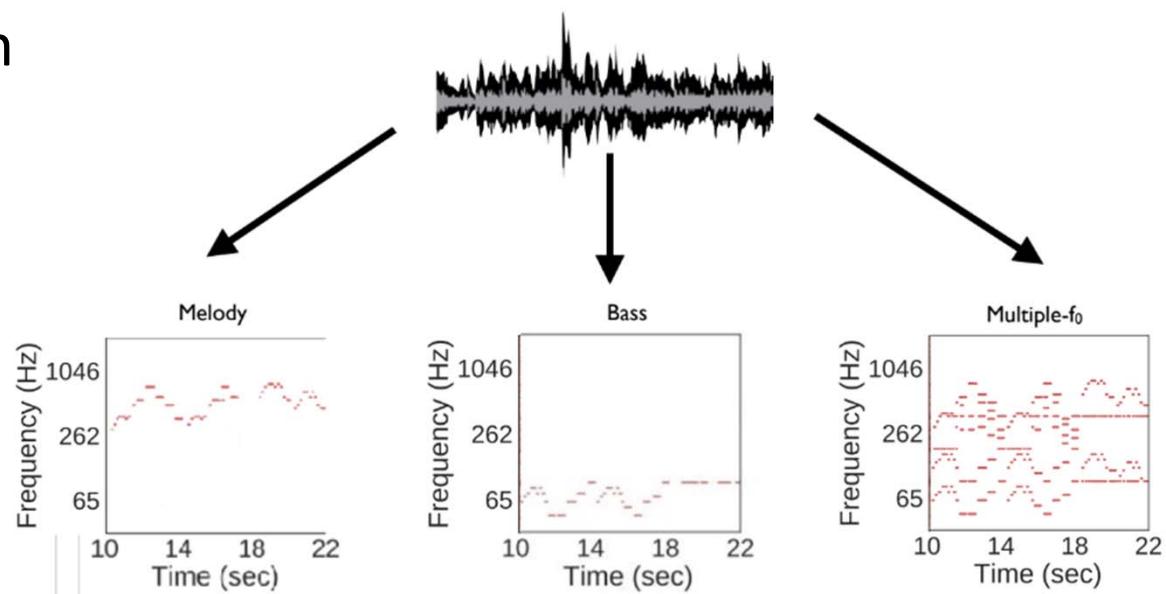


F0 Estimation

- Part 2: (Rachel Bittner)

https://drive.google.com/file/d/1Jmj6tNBGFMIEIEQldgNZ2_YgD17gS6Sv/view

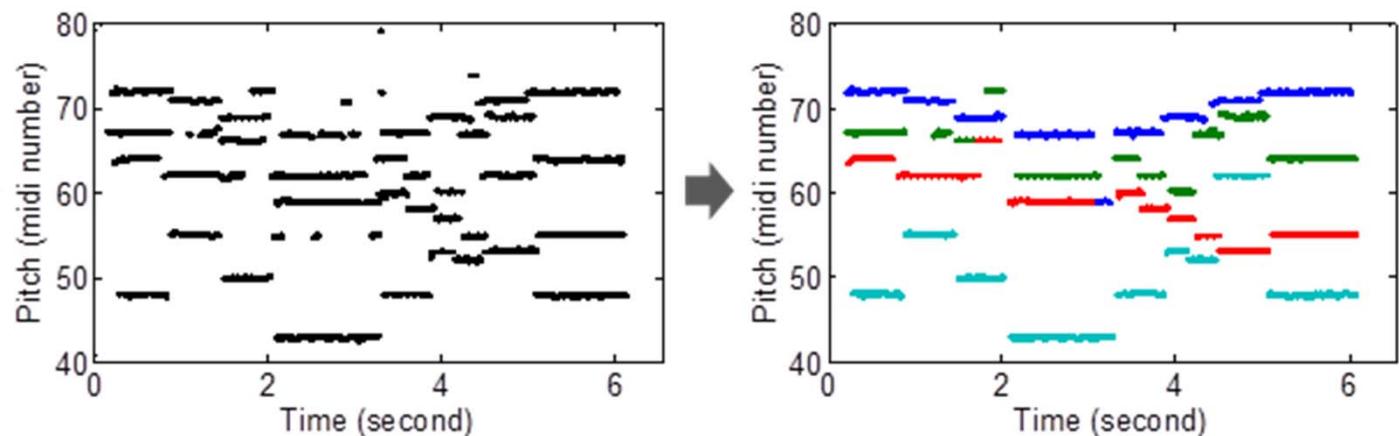
- Monophonic input vs **polyphonic** input
- Single vs **multiple** F0 estimation
- F0 estimation vs **note** estimation



F0 Estimation: Background

https://drive.google.com/file/d/1Jmj6tNBGFMIEIEQldgNZ2_YgD17gS6Sv/view

- Related tasks
 - frame-level f0 estimation
 - note estimation
 - streaming
 - score transcription
- Time resolution
- Frequency resolution
- Voicing



Melody Extraction vs. Note Transcription

- **Melody extraction:** F0 (can reflect *overshoot*, *vibrato*, *glissando*, etc)
- **Note transcription:** Note pitch (quantized in frequency)

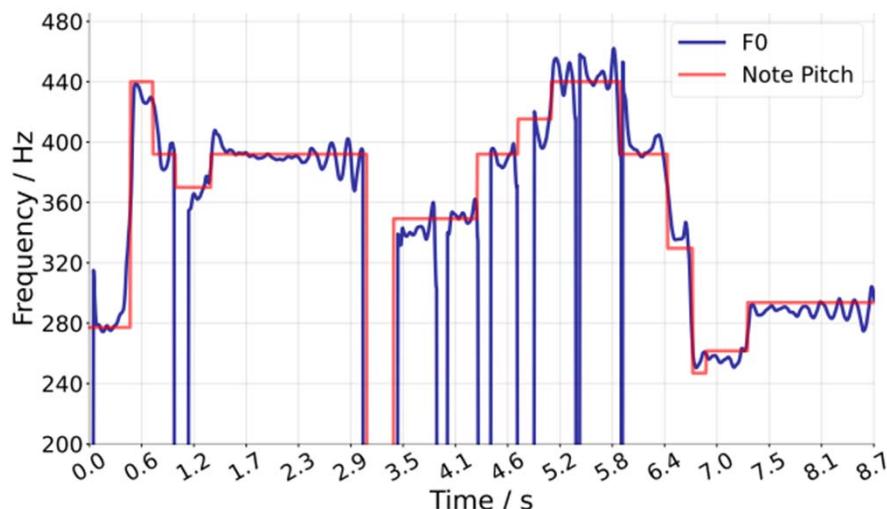


Figure source: M4Singer paper
(<https://openreview.net/pdf?id=qIDmAaG6mP>)

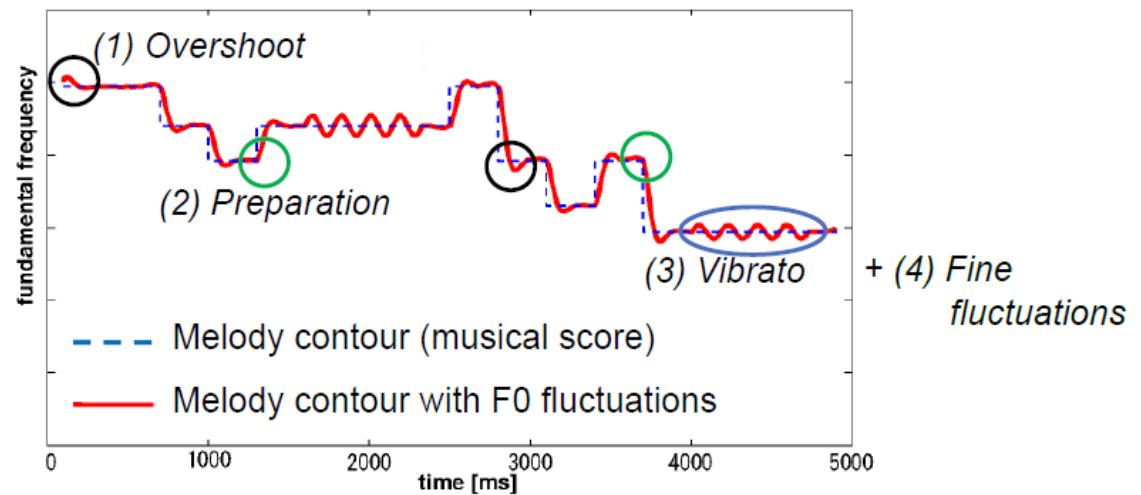
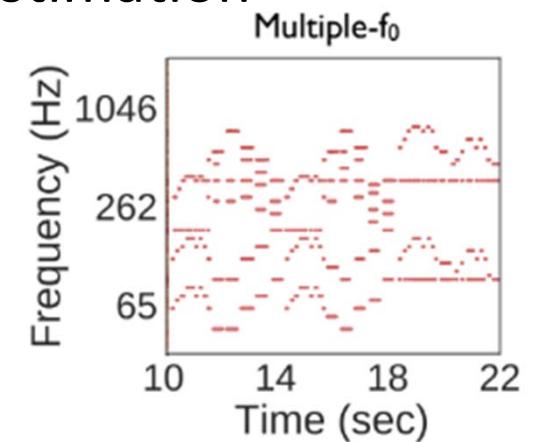
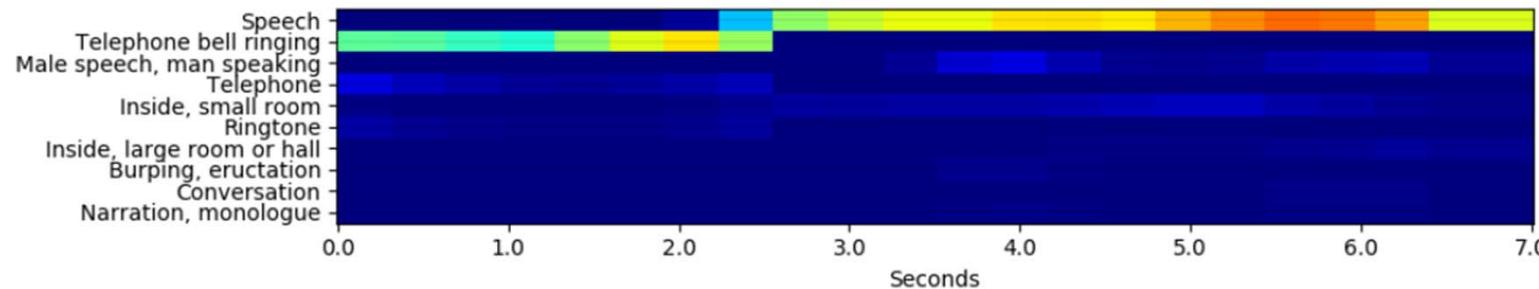


Figure from: Saitou et al, "Speech-to-singing synthesis: converting speaking voices to singing voices by controlling acoustic features unique to singing voices," WASPAA 2007

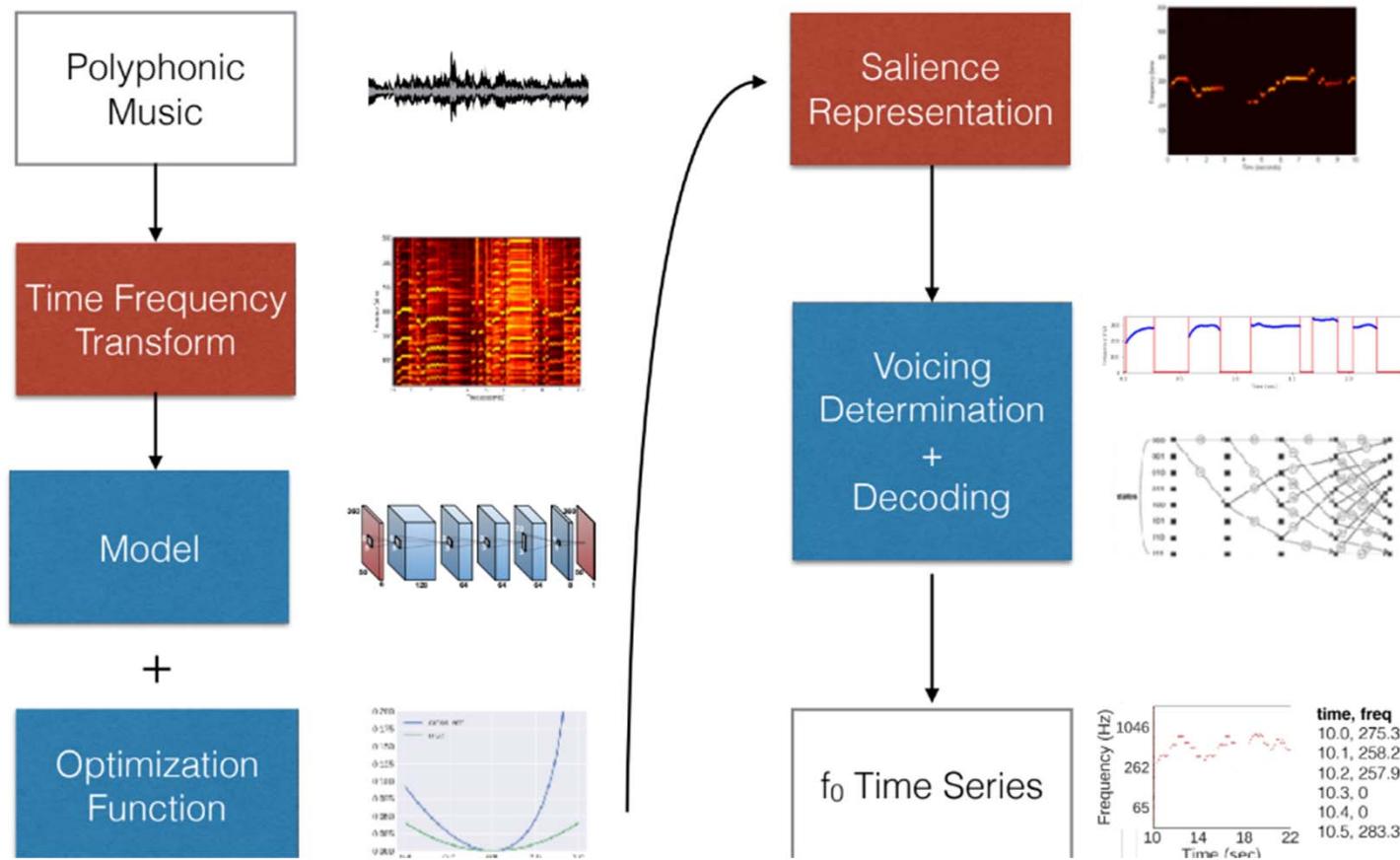
From an ML/DL Viewpoint

- **Per song:** genre classification
 - we can make predictions for each chunk and then aggregate the result over time (e.g., by taking the average logits or by majority voting on the chunk-level decisions)
- **Per short-time chunk:** audio event detection, instrument activity detection
 - e.g., output is a matrix [class x time]
- **Per time-frequency point:** f0 estimation, multi-pitch estimation
 - e.g., output is a matrix [frequency x time]



Polyphonic F0 Estimation: Typical Approach

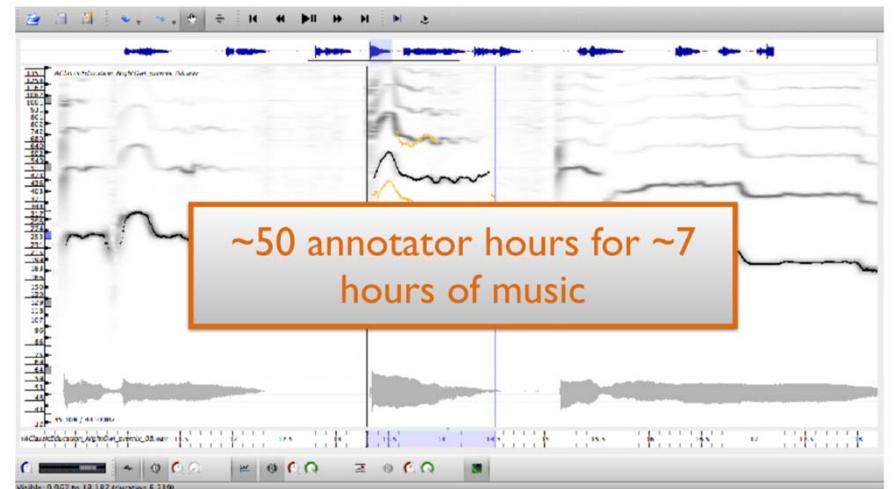
https://drive.google.com/file/d/1Jmj6tNBGFMIEIEQldgNZ2_YgD17gS6Sv/view



Supervised Training

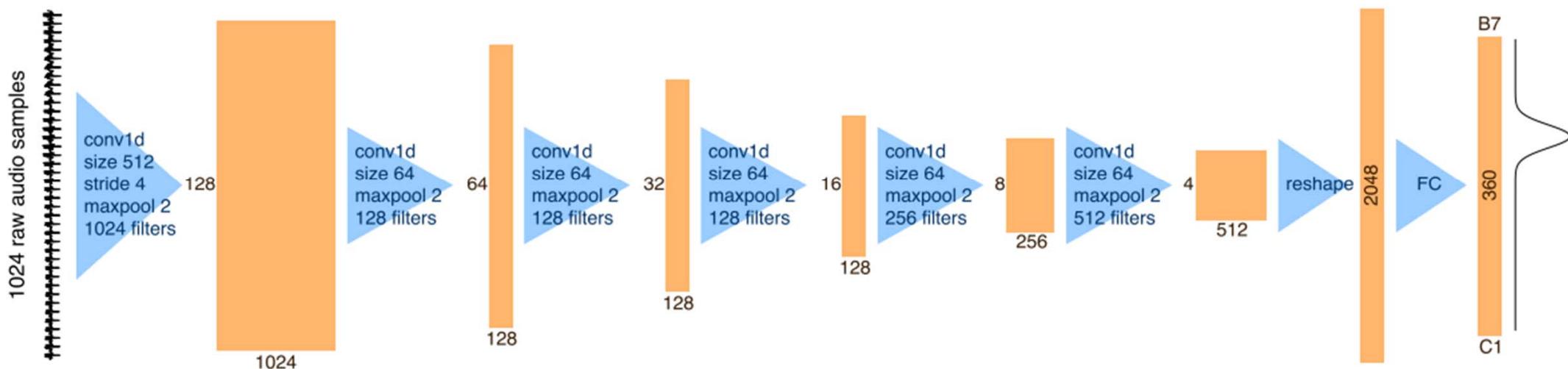
https://drive.google.com/file/d/1Jmj6tNBGFMIEIEQldgNZ2_YgD17gS6Sv/view

- But labeled data is very hard to collect
- Progress therefore a bit slow
 - Single F0 estimation for mono signals is nearly solved
 - Multi-F0 estimation still has room for improvement
 - Note transcription for arbitrary instruments remains hard
 - one exception is piano note transcription, which is nearly solved, partly thanks to high-quality piano synthesizers



Exemplar Model: CREPE (for Monophonic F0 Estimation)

<https://github.com/marl/crepe>

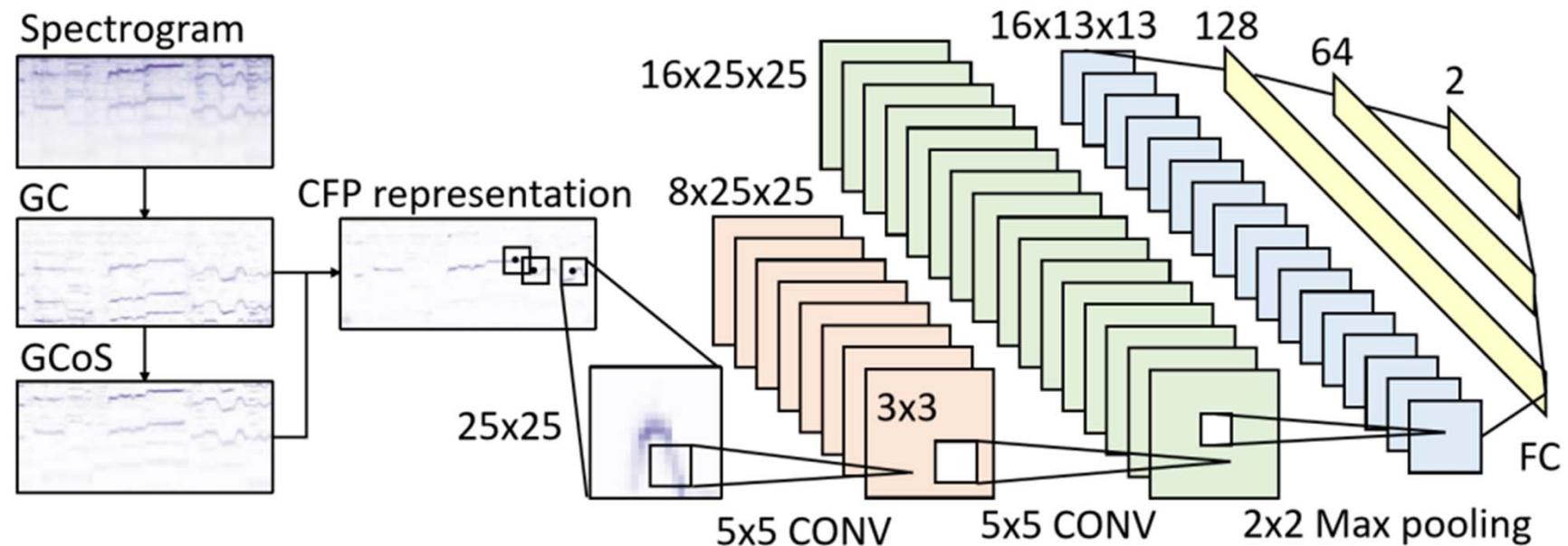


- Make a prediction every 1,024 samples
- Freq resolution: 20 cents (1/5 semitone)
- Outperform DSP-based methods such as pYIN and SWIPE

Dataset	Threshold	CREPE	pYIN	SWIPE
RWC-synth	50 cents	0.999±0.002	0.990±0.006	0.963±0.023
	25 cents	0.999±0.003	0.972±0.012	0.949±0.026
	10 cents	0.995±0.004	0.908±0.032	0.833±0.055

Exemplar Model: ExtPatchCNN (for Monophonic F0 Estimation)

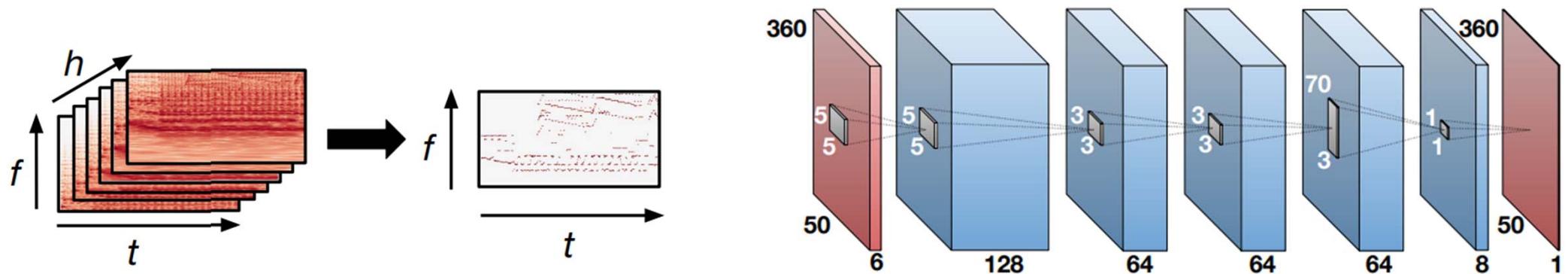
<https://github.com/leo-so/VocalMelodyExtPatchCNN>



- Binary classification: melody vs non-melody

Exemplar Model: DeepSalience (for Polyphonic F0 Estimation)

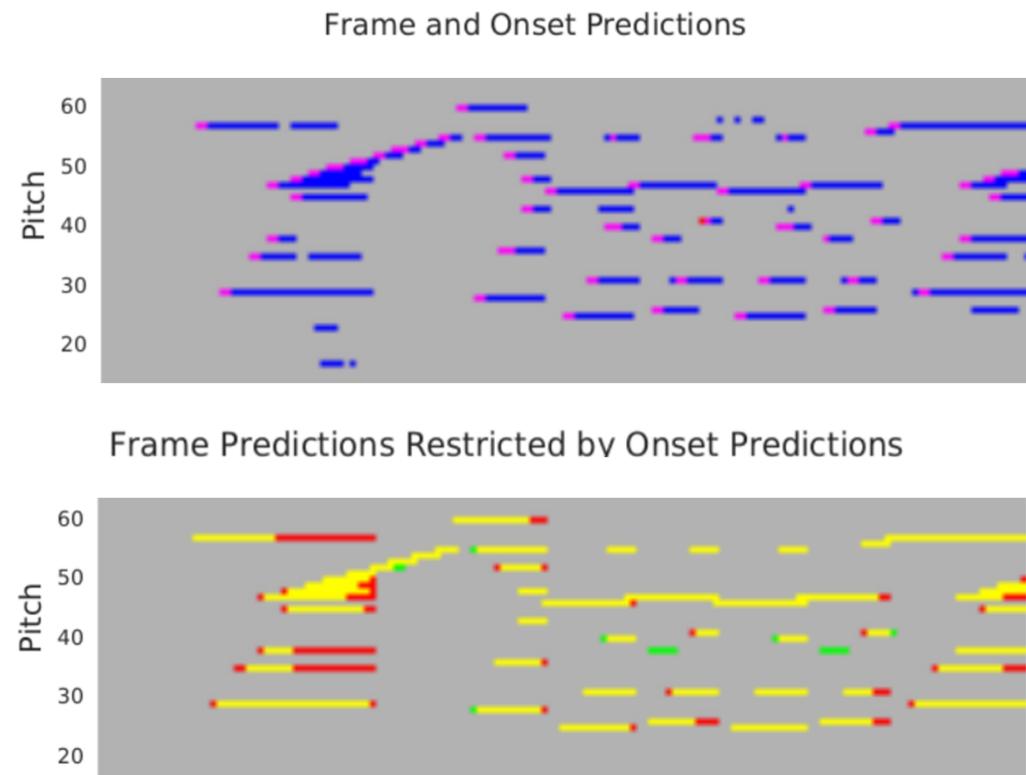
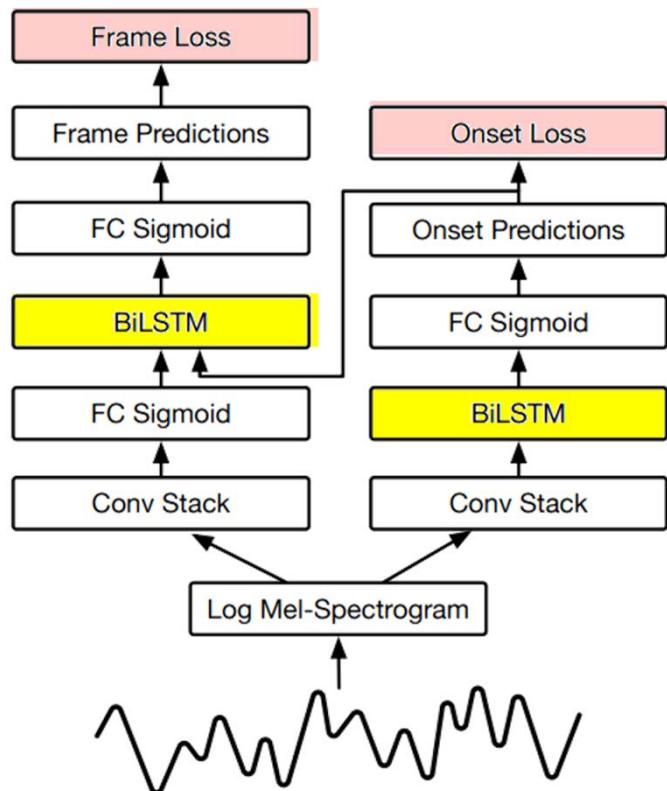
<https://github.com/rabitt/ismir2017-deepsalience>



- Input: harmonic constant-Q transform (HCQT) with 6 channels
- Model: 5-layer CNN (**no strides, no pooling → no shift-invariance**)
 - (5 x 5): 1 semitone in frequency and 50 ms in time
 - (70 x 3): 14 semitones in frequency to capture relationships between frequency content within an octave

Exemplar Model: Onset-and-Frames (for Piano Transcription)

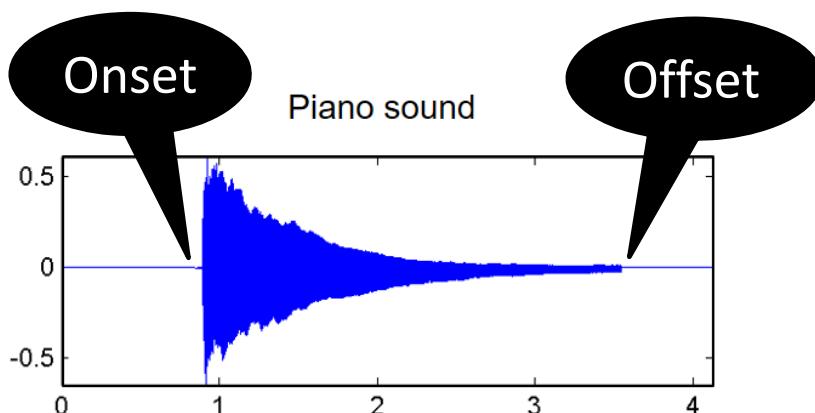
<https://magenta.tensorflow.org/onsets-frames>



Pitch, Onset, Offset, Velocity

<https://magenta.tensorflow.org/datasets/maestro>

“We partnered with organizers of the International **Piano-e-Competition** for the raw data used in this dataset. During each installment of the competition virtuoso pianists perform on **Yamaha Disklaviers** which, in addition to being concert-quality acoustic grand pianos, utilize an integrated high-precision **MIDI capture** and playback system.”



Dynamic's note velocity			
Dynamic	Velocity*	Voice	
ppp	16	Whispering	
pp	33	Almost at a whisper	
p	49	Softer than speaking voice	
mp	64	Speaking voice	
mf	80		
f	96	Louder than speaking	
ff	112	Speaking loud	
fff	126	Yelling	

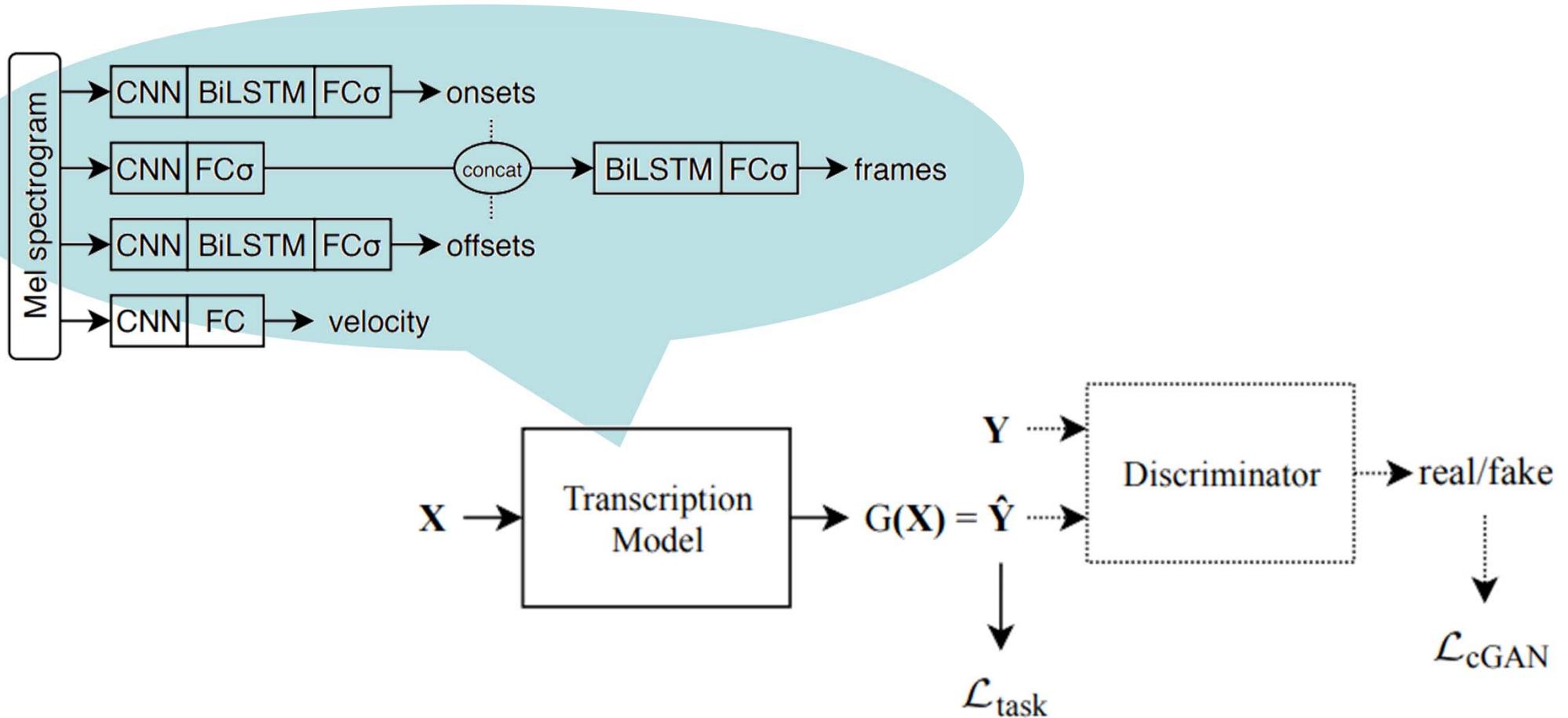


*Note velocity adopted from Logic Pro

<https://freeonlinesheetmusic.wordpress.com/2016/01/10/dynamics/>

Hawthorne et al., “Enabling factorized piano music modeling and generation with the MAESTRO dataset,” ICLR 2019

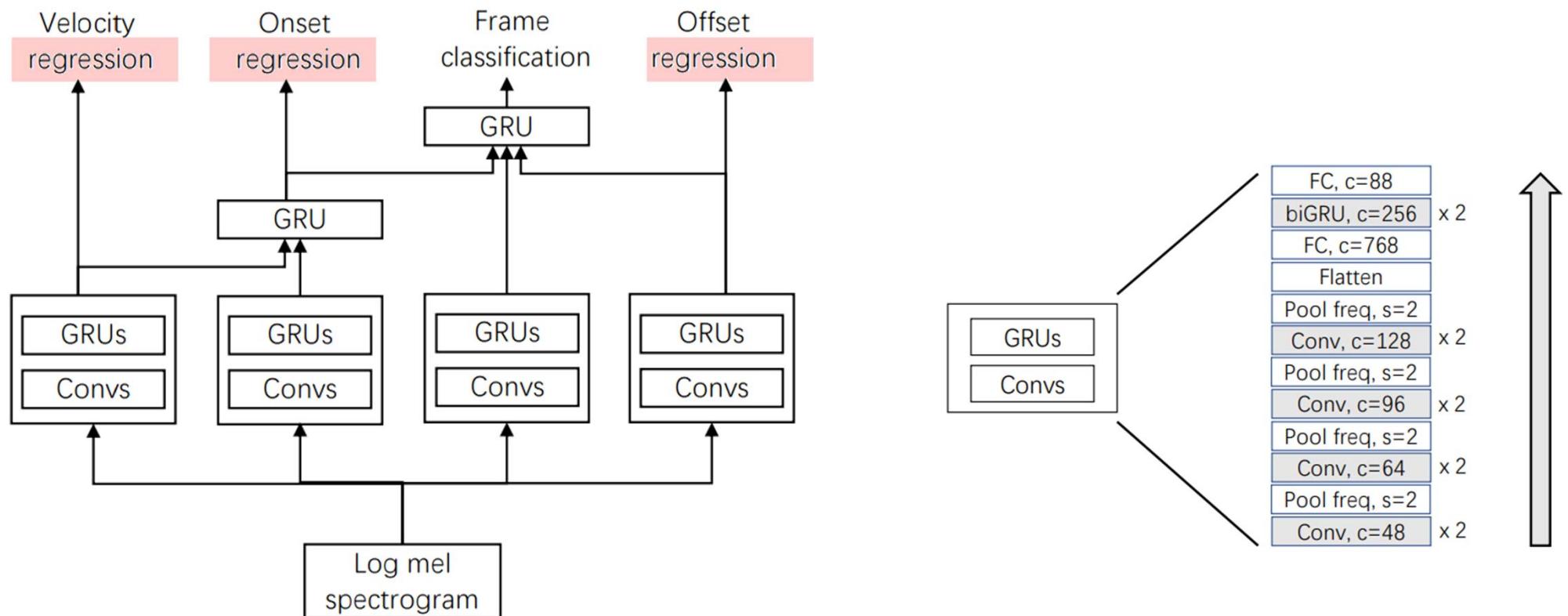
Exemplar Model: Piano Transcription with GAN



Kim & Bello, "Adversarial learning for improved onsets and frames music transcription," ISMIR 2019

Exemplar Model: High-resolution Piano Transcription

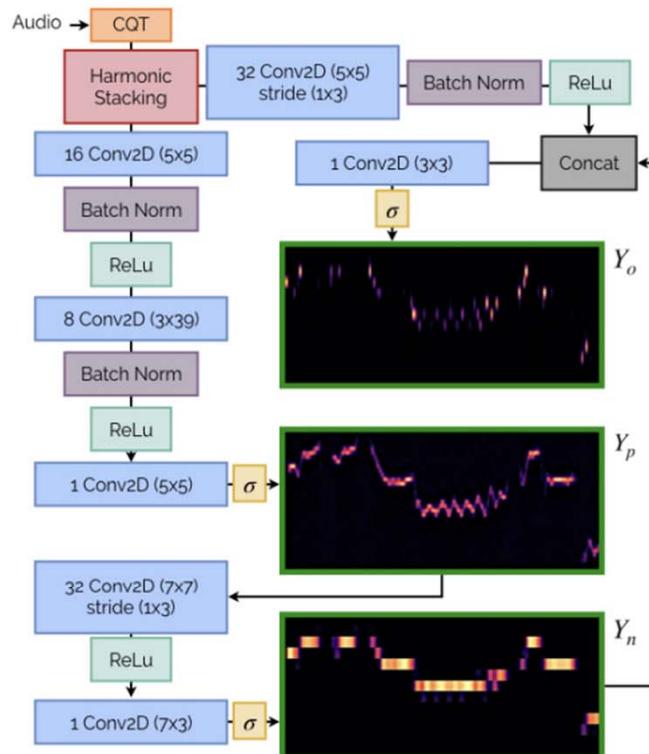
https://github.com/bytedance/piano_transcription



Kong et al., "High-resolution piano transcription with pedals by regressing onsets and offsets times," TASLP 2021

Exemplar Model: Basic Pitch (for Note Transcription)

<https://github.com/spotify/basic-pitch>



- For *instrument-agnostic* note transcription and multi-pitch estimation
- Pure CNN-based for being light-weight

Try Basic Pitch, a free audio-to-MIDI converter with pitch bend detection, built by Spotify.
[Learn more](#) or follow the instructions below.

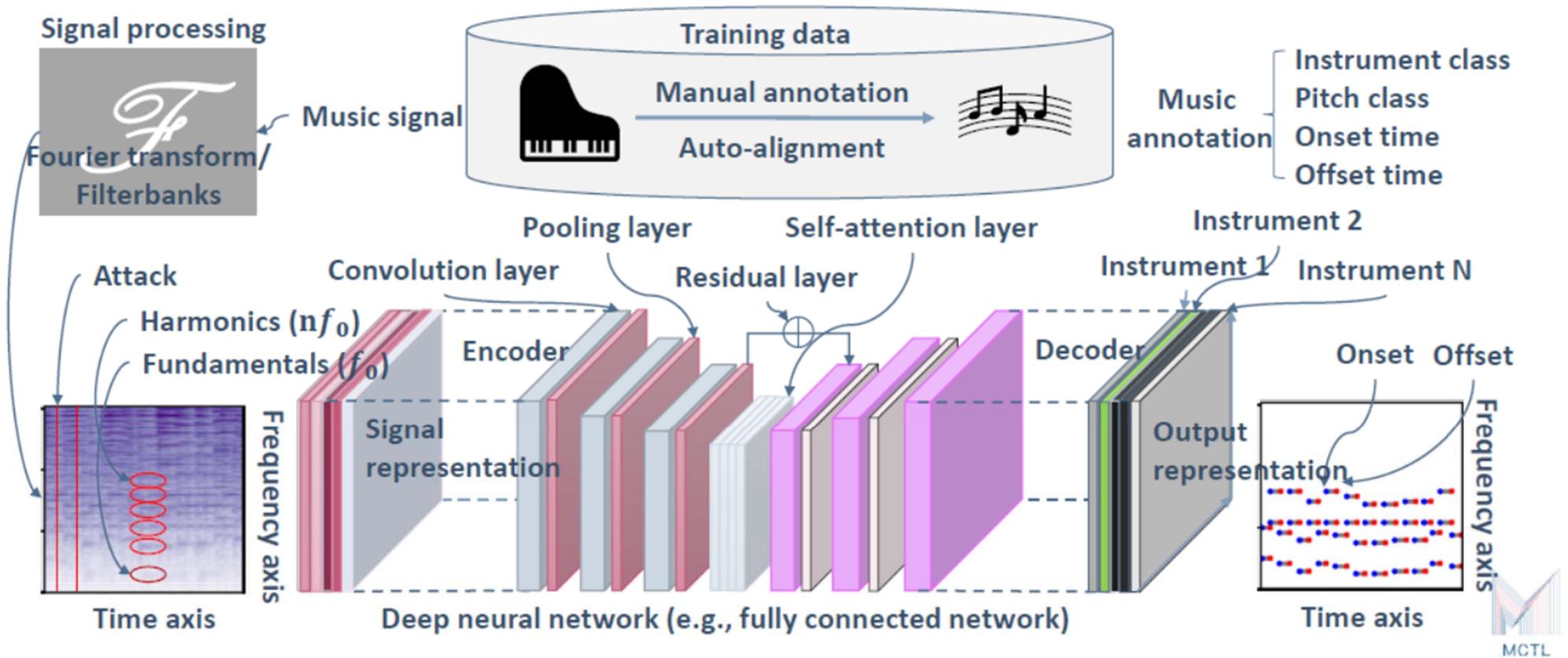
- 1 — Press record and sing a ditty into your computer. Or drop a recording of any single instrument (piano, guitar, xylophone, you name it).
- 2 — Then get a MIDI version back. Just like that.
- 3 — Download the MIDI file to fine tune and make corrections in your favorite digital audio workstation.

Fig. 1. The NMP architecture. The matrix posteriorgram outputs Y_o , Y_p , and Y_n are outlined in green. σ indicates a sigmoid activation.

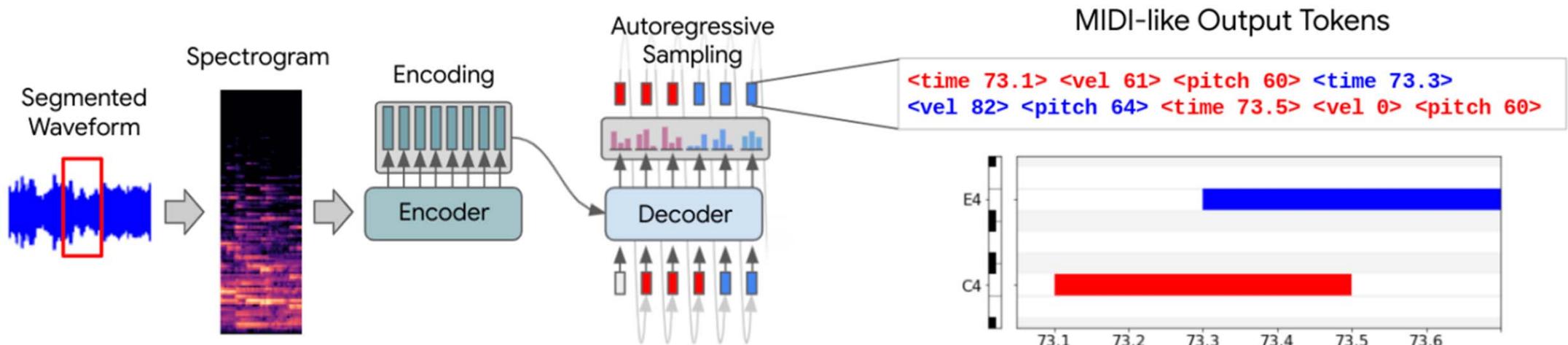
Bittner et al., “A lightweight instrument-agnostic model for polyphonic note transcription and multipitch estimation,” ICASSP 2022

Exemplar Model: Omnidartz (for Multi-Instrument Transcription)

<https://github.com/spotify/basic-pitch>



Exemplar Model: Seq2Seq Transformers (for Piano Transcription)



- Jointly modeling audio features and language-like output dependencies
- Possible to pre-train the decoder

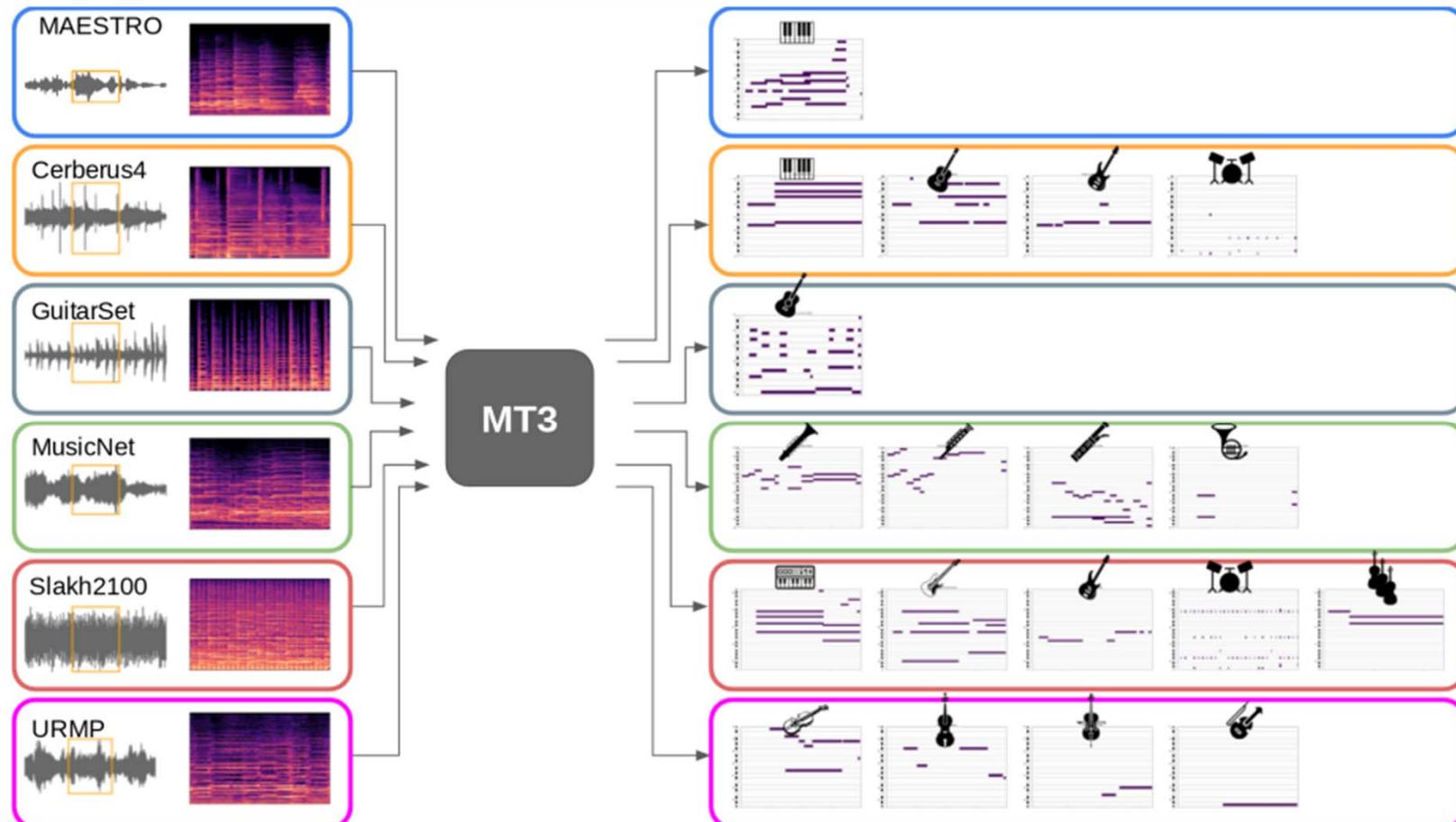
	Model	Onset, Offset, & Velocity F1	Onset & Offset F1	Onset F1
MAESTRO V1.0.0	Transformer (ours)	82.18	83.46	95.95
	Kong et al. 2020 [19]	80.92	82.47	96.72
	Kwon et al. 2020 [21]	–	79.36	94.67
	Kim & Bello 2019 [20]	80.20	81.30	95.60
	Hawthorne et al. 2019 [2]	77.54	80.50	95.32

Evaluation Metrics for Pitch-Related Tasks

https://craffel.github.io/mir_eval/

- **Single- and multi-pitch estimation**
 - correct if within 0.5 semitones of a reference frequency
 - chroma accuracy (ignore octave)
 - voicing measures
- **Piano transcription**
 - onset tolerance window: 50ms
 - offset tolerance window: 20% of note duration
 - onset only vs. onset+offset
- **Chord transcription**
 - root
 - majmin
 - majmin_inv
 - thirds
 - triads
 - tetrads
 - sevenths
 - overseg, underseg, seg

Exemplar Model: MT3 (for Multi-instrument Music Transcription)

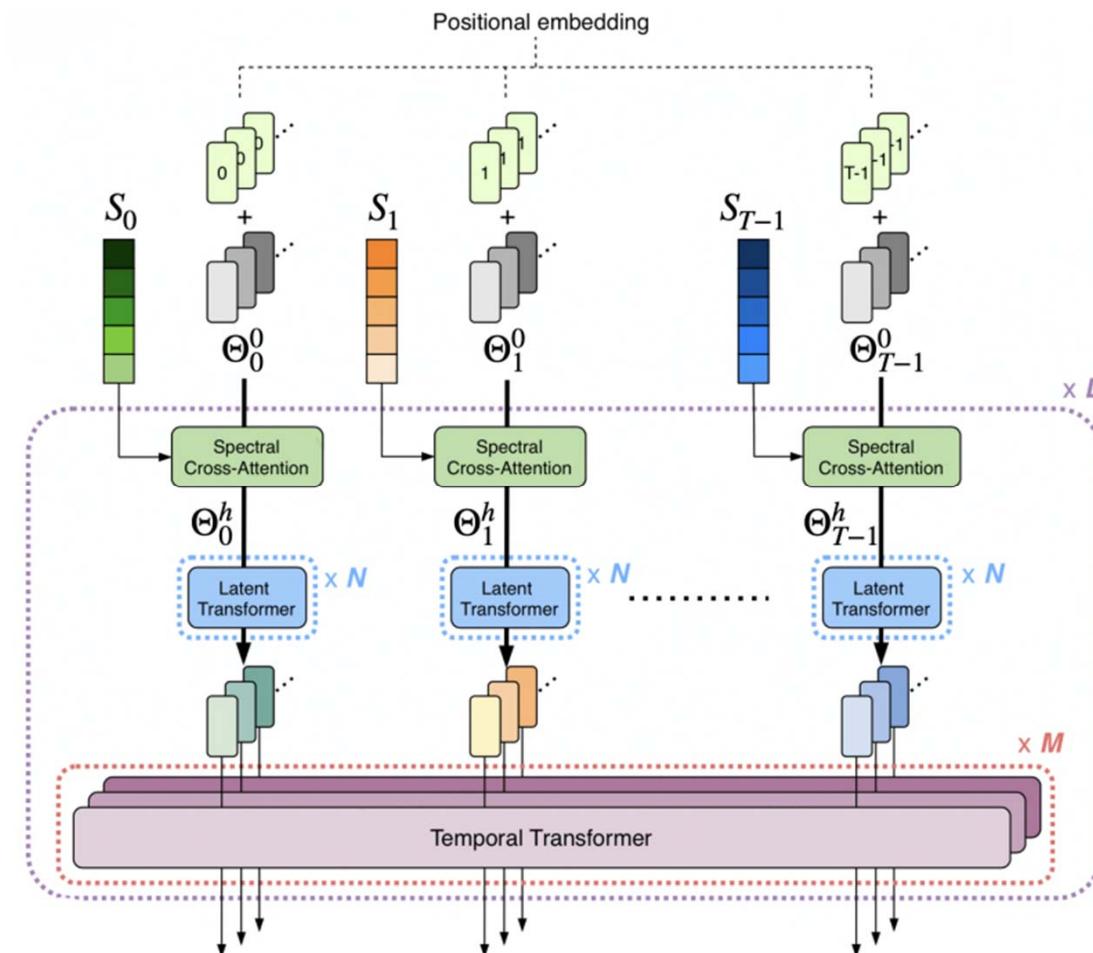


Exemplar Model: Perceiver TF (for Multi-instrument Music Transcription)

- Feedforward Transformer-encoder (not encoder/decoder)
- Use audio-domain loss, rather than MT3's symbolic-domain loss
- Onset-and-frames-like loss, but for $J > 1$ (multiple) instruments

$$\mathcal{L} = \sum_{j=0}^{J-1} (l_{\text{onset}}^j + l_{\text{frame}}^j)$$

- Outperforms MT3

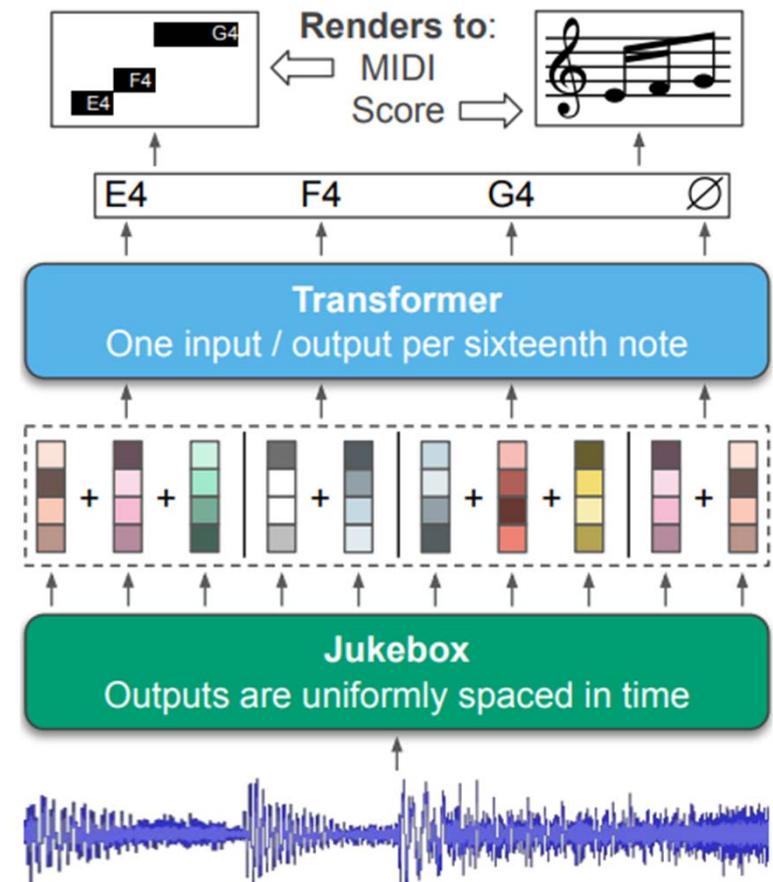


Exemplar Model: Sheet Sage (for Lead Sheet Transcription)

<https://github.com/chrisdonahue/sheetsage>

- Predict both melody and chord
 - Use a large pre-trained model called **Jukebox** as model backbone and then do transfer learning
 - Use **Transformer** to learn the language model (LM) for melody and chords
 - Computationally heavy but pretty accurate
 - Lighter alternatives: BTC
(<https://github.com/jayg996/BTC-ISMIR19>)

Donahue et al., “Melody transcription via generative pre-training,” ISMIR 2022



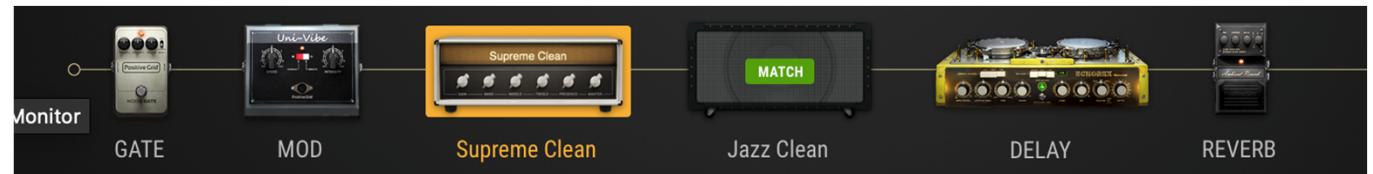
Guitar Transcription: Need to be Invariant to Audio Effects

(Examples provided by **Positive Grid®**)

Dry (single
coil EG)



In the
Clouds



Overdriven
Verb Icon



Lazy
Down



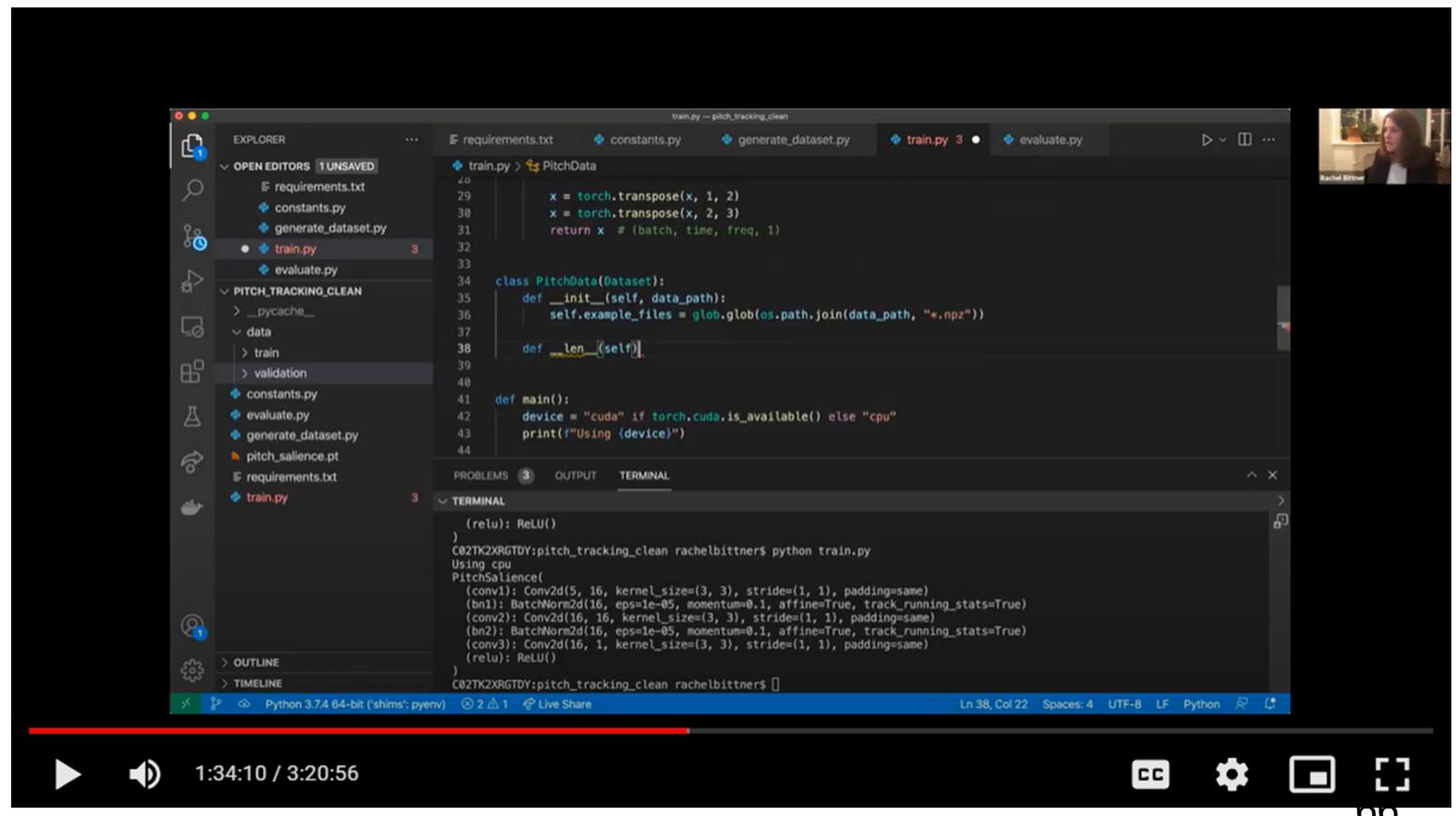
Chen et al., “Towards automatic transcription of polyphonic electric guitar music: A new dataset and a multi-loss transformer model,” ICASSP 2022

ISMIR 2021 Tutorial: Programming MIR Baselines from Scratch - Pitch Tracking

https://github.com/rabitt/ismir-2021-tutorial-case-studies/tree/main/pitch_tracking

- Video online

<https://drive.google.com/file/d/18LNaKy2ymFjEWj19gHy25wgtFV4pdML0/view>



Outline

- Music classification
 - Basics: Tasks, datasets, and evaluation
 - Supervised models
 - Self-supervised models
- Music transcription
- HW1