MIDI-DDSP for timbre transfer

Volodymyr Barannik

Supervisor: Shvai Nadiya Oleksandrivna
NaUKMA, 2023

Introduction to sound synthesis and timbre transfer

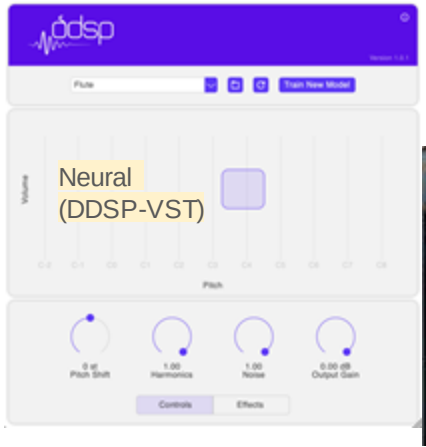
Timbre transfer

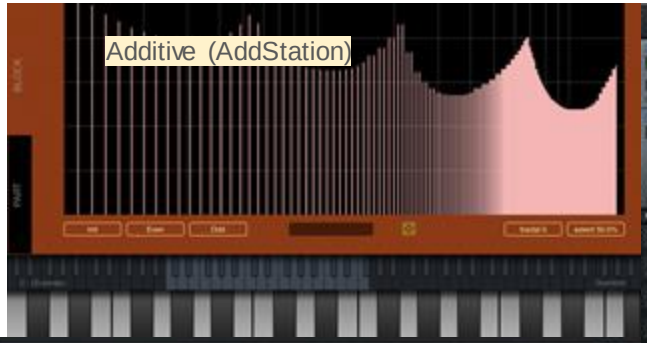
- **Timbre** *(or tone, tone color, tone quality)* is the “perceived sound quality of a musical note, sound or tone. Timbre distinguishes different types of sound production, such as choir voices and musical instruments”. [ref]

- **Timbre transfer** is a “task concerned with modifying audio samples such that their timbre is reformed while their semantic content is persisted”. [ref]

Existing approaches for generating sound

- Classic DSP
  - Additive, subtractive, FM, granular synthesis
  - Vocoding
  - Sampers/romplers
- Neural synthesis





DSP

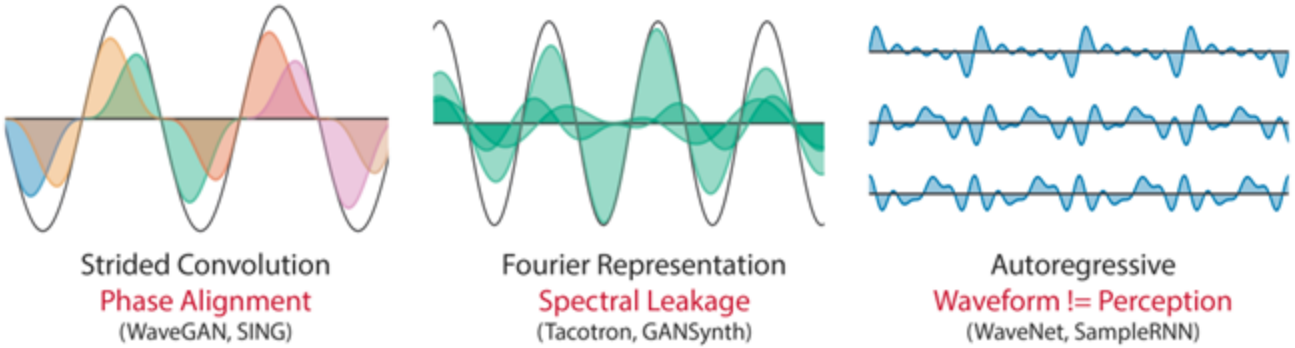
- **Classic synthesis** techniques result in unconvincing results, but are very interpretable.
- **Physical modeling** has good quality and good controls, but requires a lot of effort to create such models; computationally expensive.
- **Samplers** are extremely expensive to build, provide good quality and low or none controls.
- **Neural synthesis** as possible alternative to all of them.

Types of neural synthesis

- Input → Latent space → Waveform
- Input → Latent space → Spectrogram → Waveform
- In case of Audio → Audio synthesis (e.g. in timbre transfer):
  - Waveform → Latent space → Waveform
  - Waveform → Spectrogram → Latent space → Waveform
  - Waveform → Spectrogram → Latent space → Waveform
  - Waveform → Spectrogram → Latent space → Spectrogram → Waveform

Types of neural synthesis. Conclusions

- All of them lack interpretability.
- The output quality of the sound is poor.



[ref]

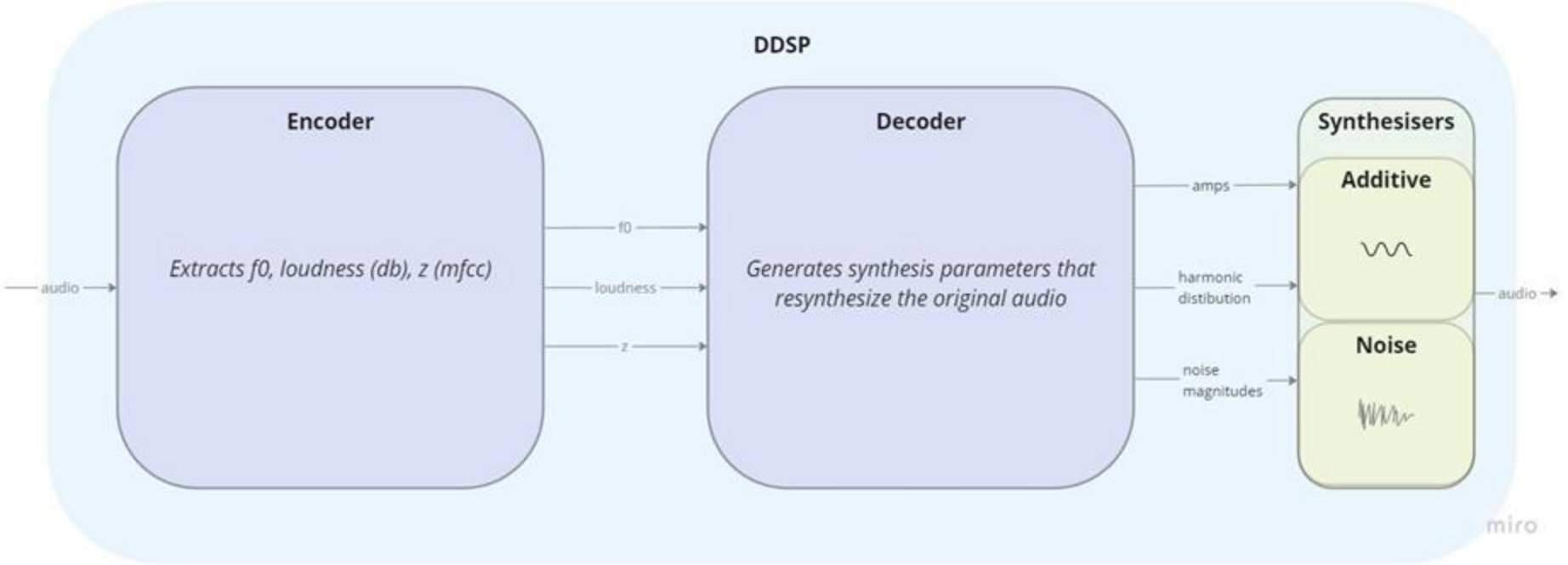
Figure 1: Challenges of neural audio synthesis. Full description provided in Section 1.1.

Introducing Google Magenta’s DDSP

DDSP – Differentiable Digital Signal Processing

- Uses neural network to predict parameters for additive and noise synthesisers.
- Waveform –*neural network*→ Synth Parameters –*synths*→ Waveform [ref]
- “DDSP enables an *interpretable* and *modular* approach to generative modeling, without sacrificing the benefits of deep learning” [ref]

DDSP – Differentiable Digital Signal Processing



DDSP – Differentiable Digital Signal Processing

- Advantages: [ref]
  - Provides good audio quality without noticeable artefacts
  - Outputs are interpretable
  - Can do timbre transfer

DDSP – Differentiable Digital Signal Processing

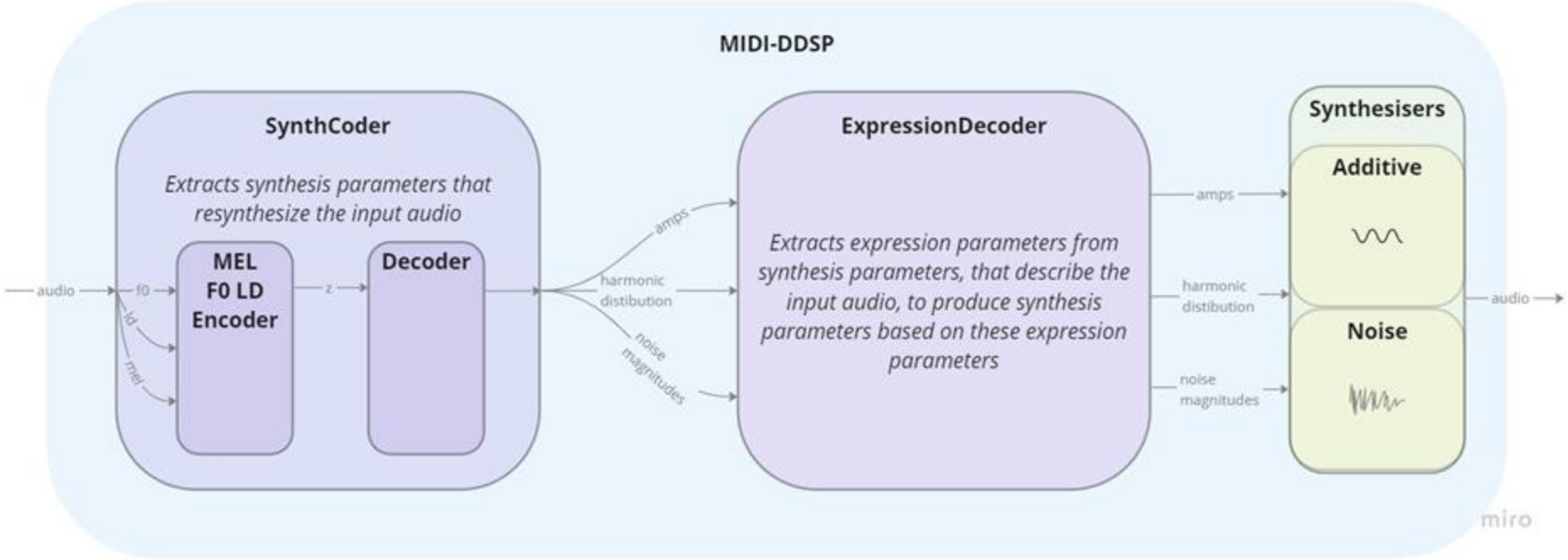
- Disadvantages:
  - Lack of inputs interpretability [ref]

- We have also found following issues with these models:
  - F0 swipes
  - Output audio somewhat lacks realism and details **(MSSL=5.0** [ref]**)**

MIDI-DDSP

- Improved version that uses additional interpretable input parameters:
  - Volume
  - Volume fluctuation
  - Volume peak position
  - Vibrato
  - Brightness
  - Attack Noise
- This is why it is called *MIDI*-DDSP: it is useful when generating sound from symbolic data (midi, automations, etc)

MIDI-DDSP


MIDI-DDSP
SynthCoder
Extracts synthesis parameters that resynthesize the input audio
audio
f0
ld
mel
MEL
F0 LD
Encoder
z
Decoder
amps
harmonic distibution
noise magnitudes
ExpressionDecoder
Extracts expression parameters from synthesis parameters, that describe the input audio, to produce synthesis parameters based on these expression parameters
amps
harmonic distibution
noise magnitudes
Synthesisers
Additive
Noise
audio
miro

MIDI-DDSP vs DDSP

- However, it cannot be the only reason why MIDI-DDSP produces better results.

Model name	MSSL* (Multi-scale spectral loss), eval	Source
vn DDSP	5.00	Magenta
vn MIDI-DDSP (full)	4.97	Magenta
vn MIDI-DDSP (synthcoder)	4.20	Magenta

* sum of $L_i = \|S_i - \hat{S}_i\|_1 + \alpha \|\log S_i - \log \hat{S}_i\|_1$

Our work

Essence of our work

- Our main objective is to check whether MIDI-DDSP model is capable of performing timbre transfer.
- To achieve this, we have conducted a series of experiments to assess the capabilities and restrictions of proposed DDSP & MIDI-DDSP models.

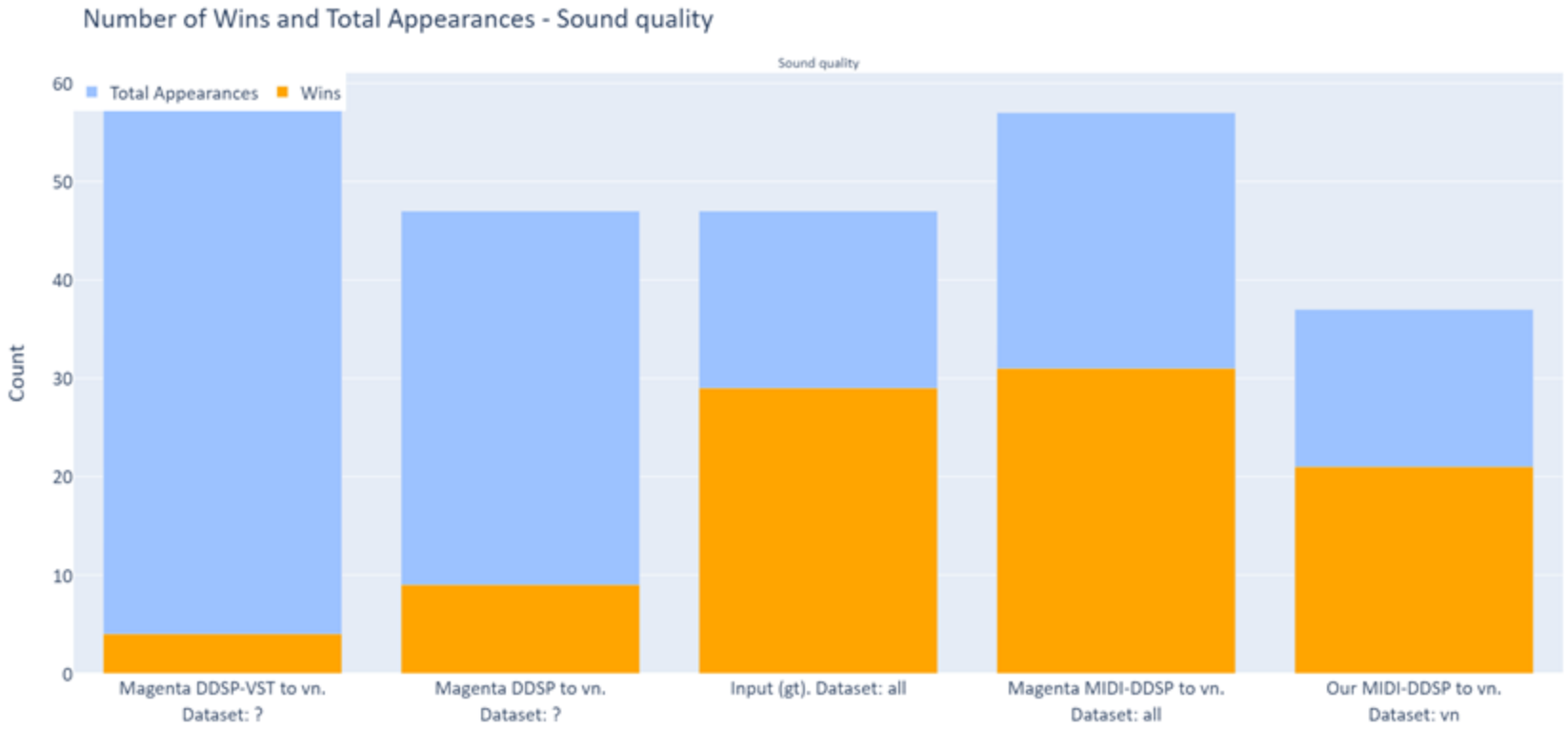
- Our secondary goal is to better explain the performance of DDSP and MIDI-DDSP models; to show some nuances that were omitted or explained poorly in the original papers.

SynthCoder + ExpressionDecoder

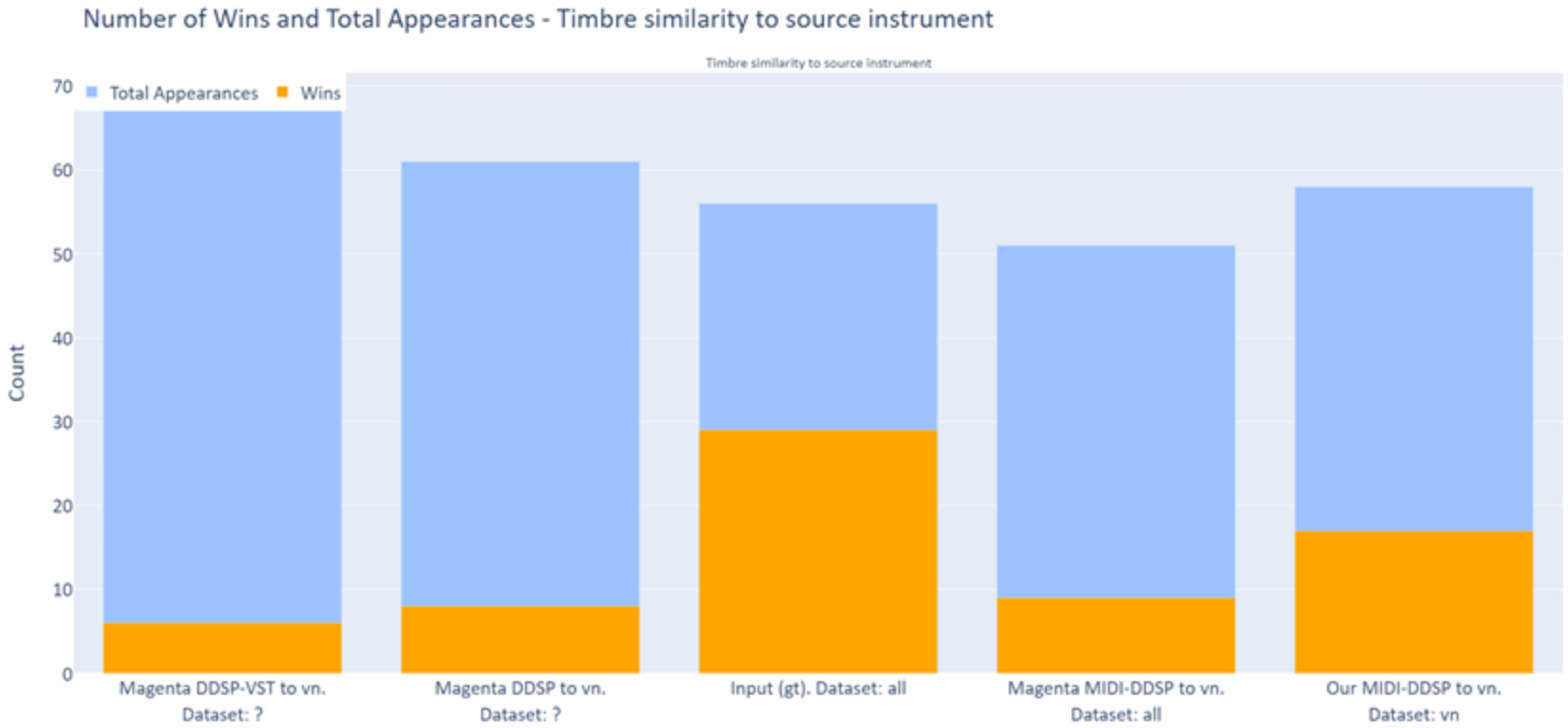
#1: Can MIDI-DDSP be used for timbre transfer?

- **Yes**, the whole MIDI-DDSP model is capable of doing timbre transfer.
- It has far better results then original DDSP model.

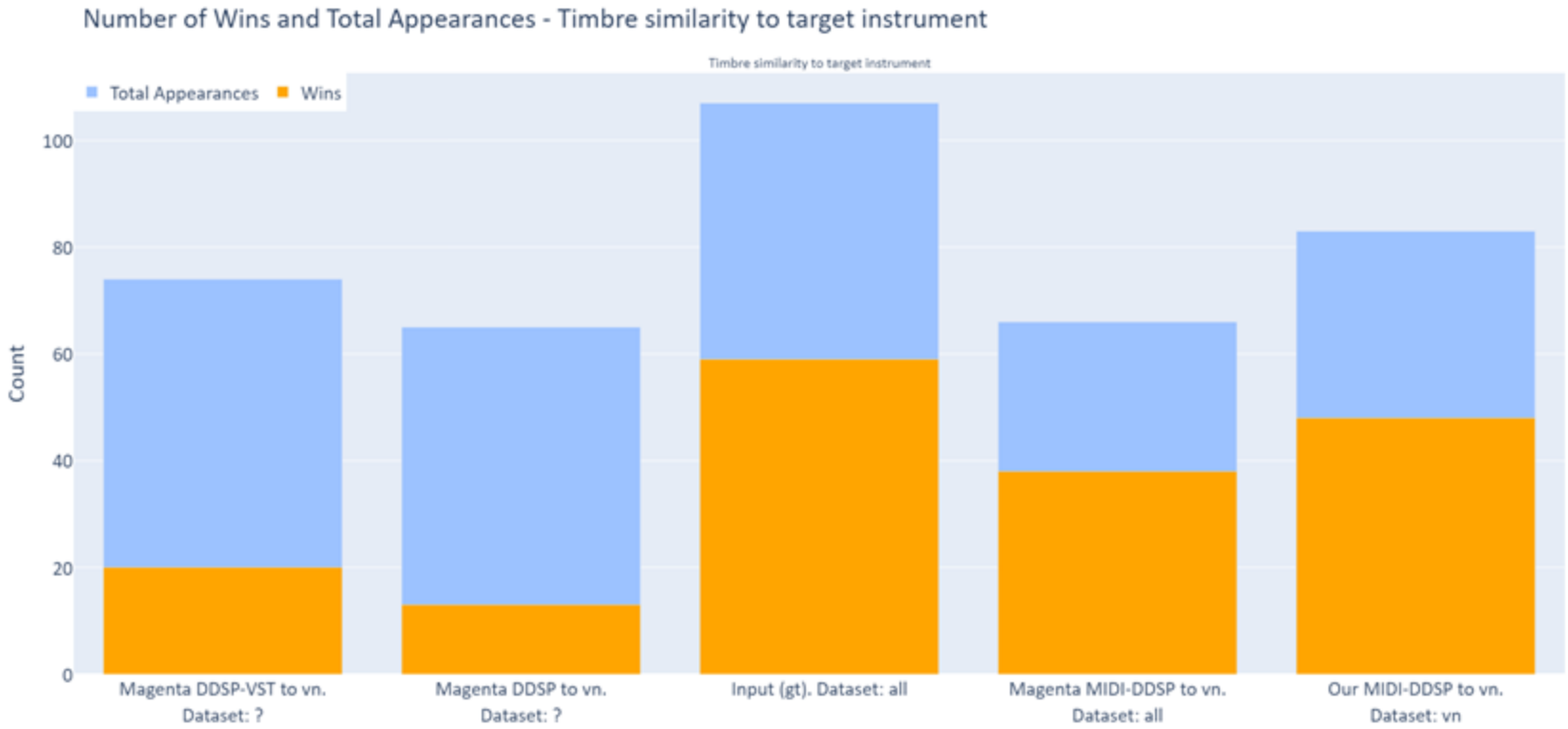
#1: Can MIDI-DDSP be used for timbre transfer?



#1: Can MIDI-DDSP be used for timbre transfer?



#1: Can MIDI-DDSP be used for timbre transfer?



#1: MIDI-DDSP: timbre transfer to violin

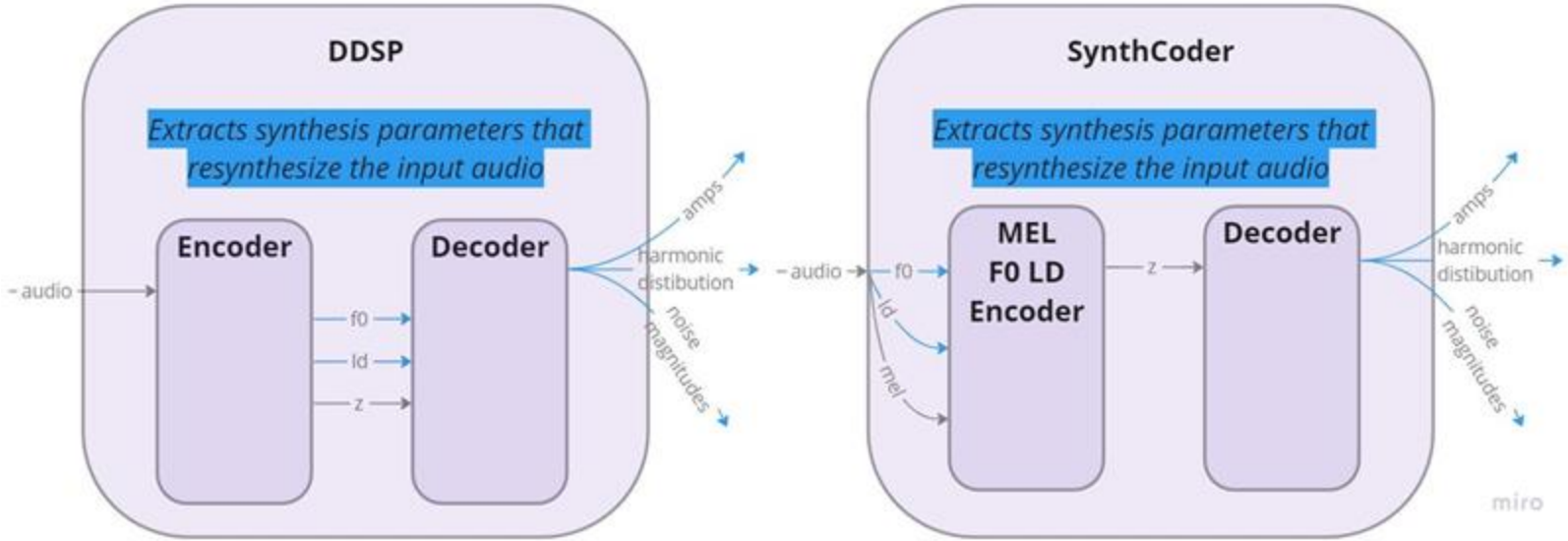
- Here are some examples of timbre transfer from other instruments to violin.
- The model was trained on violin only.

Model name	Source instrument name	Source audio	Target instrument name	Output audio	Comment
all MIDI-DDSP (full)	Oboe		Violin		Legato and vibrato are preserved well.
all MIDI-DDSP (full)	Clarinet		Violin		Articulations are preserved well. Some artifacts with harm. distr.
all MIDI-DDSP (full)	Cello		Violin		Violin, as an instrument, is not capable of playing such low notes.
all MIDI-DDSP (full)	Trumpet		Violin		Preserves staccato articulation.
all MIDI-DDSP (full)	Violin		Violin		Identity transformation.

SynthCoder

#2: Can SynthCoder be used for timbre transfer?

- SynthCoder is similar to original DDSP model
- Original DDSP model was capable of performing timbre transfer
- Hypothesis: we can use SynthCoder for timbre transfer and achieve good results with greater inference speed compared to full MIDI-DDSP model



#2: Can SynthCoder be used for timbre transfer?

- **Answering this question implies that SynthCoder does not generalize as an audio reconstruction model.**
- So if it is bad at reconstructing, it *could* be good at timbre transfer.

#2a: Can SynthCoder generalize to unseen instruments?

- Mostly yes, but sometimes no.
- We have a source audio, instrument_id and f0 for that audio. We replace each of these parameters:

Model name	MSSL (Multi-scale spectral loss)	Source for initial audio, f0 and instrument id	Source audio	Target instrument_Id	Target f0	Target input audio	Output audio	#Steps	Notes
vn SynthCoder	-	URMP Violin		Violin	same	same		high	
vn SynthCoder	–, no gt	URMP Clarinet		Violin	same	same		high	No timbre transfer
vn SynthCoder	–, no gt	URMP Clarinet		Violin	oct up	same		high	pitch shifting
vn SynthCoder	–, no gt	URMP Clarinet		Violin	same	Uniform Noise		high	

#2a: Can SynthCoder generalize to unseen instruments?

- More extreme substitutions to explore limits of SynthCoder:

Model name	MSSL (Multi-scale spectral loss)	Source for initial audio, f0 and instrument id	Source audio	Target instrument_Id	Target f0	Target input audio	Output audio	#Steps	Notes
vn SynthCoder	–, no gt	URMP Clarinet		Violin	same	URMP Violin 		high	f0 is played perfectly fine. Timbre is mangled.
vn SynthCoder	–, no gt	URMP Clarinet		Violin	same	Synthesized Rhythmic Pluck 		high	The rhythm is preserved! Timbre is somewhat too
vn SynthCoder	–, no gt	URMP Clarinet		Violin	oct up	Synthesized Bass Rhythmic Pluck 		high	The rhythm is preserved! Timbre is somewhat too
vn SynthCoder	–, no gt	URMP Clarinet		Violin	same	Sampled Violin Orchestra 		high	

#2a: Can SynthCoder generalize to unseen instruments?

- What about poor-trained SynthCoder?
- It's still doing resynthesis.

Model name	MSSL (Multi-scale spectral loss)	Source for audio, f0 and instrument id	Source audio	Target instrument_Id	Target f0	Target input audio	Output audio	Model training dataset	Steps
vn SynthCoder	-	URMP Violin		Violin	same	same		URMP violin	low
vn SynthCoder	–, no gt	URMP Clarinet		Violin	same	same		URMP violin	low

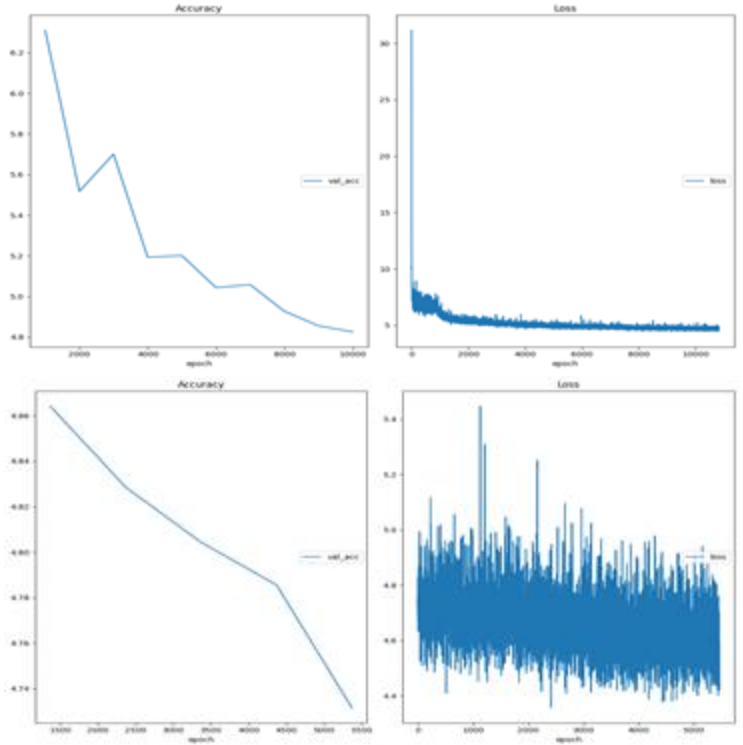
#2: Can SynthCoder be used for timbre transfer?

- So the answer is in most cases no, because it's a quite good resynthesizer.
- And it can generalize to other instruments and be used for pitch shifting.

#3: Can SynthCoder train on dataset w/ multiple instruments?

- Yes. It has MSSE=4.74 and it could be pushed even further by training longer.

Model name	MSSL (Multi-scale spectral loss)	Source	Dataset	Steps
vn DDSP	5.00 (eval)	Magenta	URMP violin	no data
vn MIDI-DDSP (full)	4.97 (eval)	Magenta	URMP violin	50k
vn MIDI-DDSP (synthcoder)	4.20 (eval)	Magenta	URMP violin	10k/10k
vn MIDI-DDSP (synthcoder)	4.74 (train)	Magenta	URMP violin	~4k/10k
vn MIDI-DDSP (synthcoder)	4.42 (eval)	Us	URMP violin	6.2k/6.2k
all MIDI-DDSP (synthcoder)	4.74 (eval)	Us	URMP violin,viola, cello, clarient, trombone, saxophone, tuba..	15k/15k

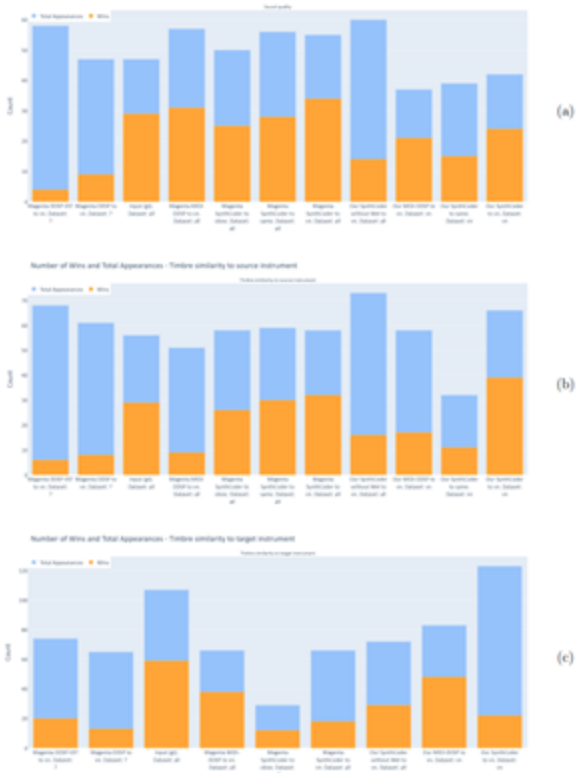


#4: Can multi-instr. SynthCoder be used for timbre transfer?

- It is bad at timbre transfer.

#5: Why is SynthCoder such a good resynthesizer? (and bad timbre transfer model)

- Ablation study



Real-time applications

DDSP-VST

- Magenta has a VST plugin that runs the original DDSP model (with frame size of 1024 samples).
- Because Tensorflow applies DENSE layers to the last dimension, and our time dimension is first after the batch size, we can generate audio frame by frame
- We can’t run the model in real time with tiny frame size because it requires high computing power.
- We can try to change frame size to be bigger than during training and see how the model behaves.

#6: How does SynthCoder react to bigger frame size?

- The audio quality does not degrade (because synths interpolate parameters they receive)
- F0 has ‘slides’ artefacts (as in DDSP which also had big frame size)

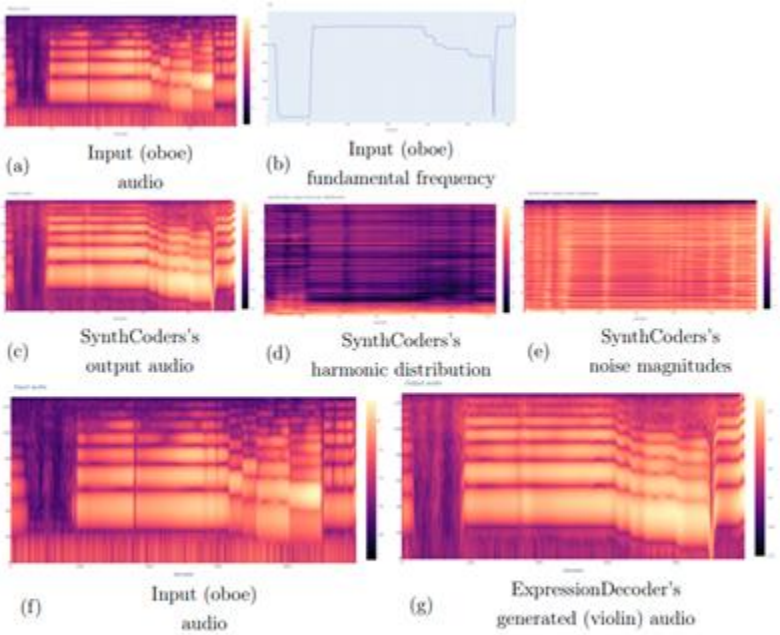


Figure 15: Outputs of the 'vn' SynthCoder model for oboe → violin timbre transfer with inputs downsampled 8 times. The spectrograms are created using MelSTFT [24] [27] with frame_size=64, n_fft=1024, num_mels=128.

#7: Can SynthCoder be used in real time?

- Yes.

Conclusions

Conclusions

- **MIDI-DDSP**

    1. MIDI-DDSP is capable of performing timbre transfer with great fidelity and articulation preservation,

- **SynthCoder**

    1. SynthCoder is generally not capable of performing timbre transfer
        a. The reason is that SynthCoder extracts too much information from input audio (compared to original DDSP)
    2. But SynthCoder is quite good at resynthesis, generalizes to other instruments and can be used for pitch shifting
    3. SynthCoder cannot perform timbre transfer with bigger frame sizes, performance degrades significantly.
    4. We have proposed a real-time implementation of SynthCoder

Future work

1. Adapt MIDI-DDSP so that it would take only audio as input (and no additional technical data used for training)
2. Make MIDI-DDSP model train faster and/or train on smaller datasets
3. Adapt MIDI-DDSP to real time
4. Polyphonic MIDI-DDSP
5. Try using other synthesizers to see their effect on training speed and output quality

Thank you for your attention