Sound Morphing between instruments with DDSP models

Jimena Arruti, Jordie Shier, Anna Xambó Sedó
Centre for Digital Music
Queen Mary University of London
London, UK

***Abstract*—Sound morphing between instruments in neural synthesis typically requires retraining a model whenever a new instrument is added to a morphing system. We explore two alternatives that avoid retraining: interpolating the predicted synthesis parameters of two independently trained DDSP Harmonic + Noise models, and interpolating the weights of two models fine-tuned from a common pre-trained model. Using the URMP dataset, we evaluate cross-instrument morphs with both objective metrics and a listening test. Results show that weight interpolation only produces meaningful morphs when both models are fine-tuned from a common pre-trained model, while parameter interpolation succeeds for independently trained models as well. Listening test results confirm both methods produce perceptually intermediate timbres for most trials, although the morphing exhibits non-monotonic timbral progression for a substantial minority. Some degree of audio degradation is observed when morphing, although audio quality is still situated between "good" (4) and "fair" (3) (MOS is 3.89 for parameter interpolation of independently trained models and 3.79 for weights interpolation of fine-tuned models). These results demonstrate the viability of sound morphing with the proposed approaches for DDSP models, and we suggest future work to further validate them.**

***Index Terms*—DDSP, sound morphing, timbre transfer, model weights interpolation**

I. Introduction

Sound morphing can be defined as the process of creating hybrids between two sounds, blending their auditory features, while enabling gradual transitions between one sound and the other (Caetano and Rodet, 2013). This family of techniques has great potential for sound design, and can currently be explored directly in Digital Audio Workstations with plug-ins such as NSynth[1], ASAP Spectral Morphing[2] and MORPH[3].

Engel et al. (2020) proposed Differentiable Digital Signal Processing (DDSP) as a paradigm to combine deep learning methods with classical signal processing elements expressed differentiably, allowing them to be used within gradient-based optimisation training. They demonstrated a DDSP implementation of Spectral Modeling Synthesis (Serra and Smith, 1990) is successful in reproducing the timbre of monophonic harmonic instruments by learning synthesis parameters from audio data. The resulting models additionally enable timbre transfer, where input audio recordings are transformed to sound like the instrument used during model training. This model is highly explainable, and has the additional advantage of being small when compared to other architectures used for timbre transfer such as RAVE (Caillon and Esling, 2021), largely due to the incorporation of inductive bias within the model. This computational efficiency makes the inference easily run on most consumer hardware, requires less data for training, and can be trained in a reasonable amount of time[4].

Given the benefits of the DDSP Spectral Modeling Synthesis implementation, this project aims to extend its capabilities to enable sound morphing between models. To define the direction of the research, having a concrete use case in mind, it is helpful to imagine a scenario where a sound designer would like to interactively morph between different pairs of instruments. If they would like to add a new instrument to the morphing pair possibilities, any retraining needed for the model behind the morphing should be minimised as much as possible. While previous research on morphing with neural synthesis suggests an approach of conditioning the learning process and producing the morphing with variations in the latent space, this would imply that every time a new instrument is to be added to the morph possibilities, the model needs to be retrained with the data corresponding to all the instruments we would like to morph. Instead, we propose to explore two different approaches to sound morphing between two models: the interpolation of synthesis parameters predicted by the two models, and the interpolation of the learned weights of the two models. These methods have the benefit of maintaining the DDSP architecture, without adding complicated conditioning mechanisms.

To provide an initial validation of these approaches, this work aims to answer the following research questions:

- *RQ1) Does interpolating the weights of two DDSP models, or interpolating the synthesis parameters predicted by two DDSP models, result in a perceived morph of the two instruments?*
- *RQ2) When the interpolation of the weights or the parameters is linear, how does the timbre of the resulting morphing behave?*
- *RQ3) Is there audio quality degradation when applying the proposed methods for morphing?*
- *RQ4) How do the two methods compare overall?*

[1]https://magenta.withgoogle.com/nsynth-instrument
[2]https://forum.ircam.fr/shop/en/asap/spectral-morphing
[3]https://www.zynaptiq.com/morph/

[4]The research team (Magenta) states it takes 2 to 3 hours to train the model for 10 minutes of audio using Google Colab on their training notebook.

We evaluate both methods with objective and subjective metrics, and show that a condition for enabling weight interpolation is that the DDSP models are fine-tuned from a common pre-trained model.

II. Related Work

A. Sound Morphing and Neural Synthesis

As introduced above, sound morphing involves gradually transitioning between the auditory features of two sounds (Caetano and Rodet, 2013). This should ideally be explored by interpolating meaningful descriptors of timbre (Caetano and Rodet, 2010), and ensuring these variations are perceived as linear transitions (Caetano and Rodet, 2013).

Since the popularisation of neural networks, new data-driven approaches have been developed for sound morphing. Engel et al. (2017) utilised a WaveNet-style autoencoder which allowed to interpolate between instrument embeddings, and further provided a MaxForLive device to enable creative exploration of this technique[5]. Esling et al. (2018), on the other hand, proposed the regularisation of the latent space of Variational Auto-Encoders (VAEs) by leveraging perceptual distances analysed in timbre studies. Çakir and Virtanen (2018) obtained results for morphing pairs of instruments by interpolating their latent representations in a variational, convolutional recurrent autoencoder. RAVE (Caillon and Esling, 2021), allowed for navigating a VAE's latent space to retrieve new sounds in real-time, and has been widely adopted for creative purposes. DrumGAN (Nistal et al., 2020) attempted to enhance the control over the output timbre by conditioning on high-level timbre features.

B. Differentiable Digital Signal Processing (DDSP)

Differentiable Digital Signal Processing (DDSP) (Engel et al., 2020) was proposed as a paradigm that integrates the potential of deep learning methods with the domain-based knowledge of classic signal processing elements, providing direct access to synthesis parameters. Other benefits of this architecture include high computational efficiency — since the signal can be externally generated by an efficient synthesis algorithm — the reduction of audio artefacts previously found due to phase discontinuity, high interpretability of results, and requiring considerably less training data than other currently proposed neural audio synthesis approaches (Hayes et al., 2024).

Engel et al. (2020) demonstrate the DDSP paradigm by implementing a differentiable version of the Harmonic + Noise model from Spectral Modeling Synthesis (SMS) (Serra and Smith, 1990). This model is intended for monophonic harmonic instruments, as it can only represent a harmonic source with one fundamental frequency at a time. This allows to parameterise the sound by combining an additive synthesiser, which adds harmonics of the fundamental frequency, and a subtractive synthesiser, which is obtained by filtering white noise.

[5]https://magenta.withgoogle.com/nsynth-instrument

C. Weights interpolation

In order to avoid continuously retraining every time we add a new instrument, we need to be able to combine existing models for different instruments that were trained independently. In the context of deep learning models, and with the goal of achieving the best performance, the fusion of existing models is widely explored in the literature with different approaches, including weight averaging (Li et al., 2023).

For the case of combining two models representing two different instruments, the idea of interpolating between the models' weights seems particularly compelling. Directly interpolating the weights of two independently trained models will probably not give any meaningful results, since there is no guarantee that individual weights will contribute to the same outputs when training different models from random initialisation. Prior work exploring the relationship between fine-tuned models and the pre-trained models they have as origin observed that "fine-tuned models optimized independently from the same pre-trained initialization lie in the same basin of the error landscape" (Neyshabur et al., 2020). This inspired model soups (Wortsman et al., 2022), which proposed to fine-tune multiple models and average their weights to achieve the best performing model.

This suggests a possible approach for our use case would be pre-training a single DDSP model on a diverse dataset of sounds and then fine-tune the model on specific instruments to get models for each instrument. Because each model has the same initialisation, interpolation between the weights may be perceptually relevant. New models could then be obtained by only fine-tuning on new datasets.

III. Methodology

A. DDSP Harmonic + Noise Model

The DDSP model used for this work is the Harmonic + Noise model described by Engel et al. (2020) for the supervised case and solo violin dataset, as depicted in Figure 1. The input audio (target audio) is analysed for loudness, and the fundamental frequency is detected using a pretrained CREPE model (Kim et al., 2018). The Decoder (further detailed in Appendix B) operates with those latent representations and outputs the necessary parameters for synthesising a harmonic audio output and a filtered noise audio output. After synthesising the partial outputs, the addition of both signals is passed through a reverb module, obtaining the final synthesised audio. A multi-scale spectrogram loss is utilised to evaluate the similarity between the target audio and the synthesised audio, and adjust the weights of the model accordingly during training.

1) Harmonic synthesis: Given a harmonic signal with a fundamental frequency, we can express the frequency of its k harmonics as

$$f_k(n) = k f_0(n)$$

Additionally, we can express the amplitude of the harmonics as a global amplitude multiplied by an amplitude per harmonic

$$A_k(n) = A(n) c_k(n)$$

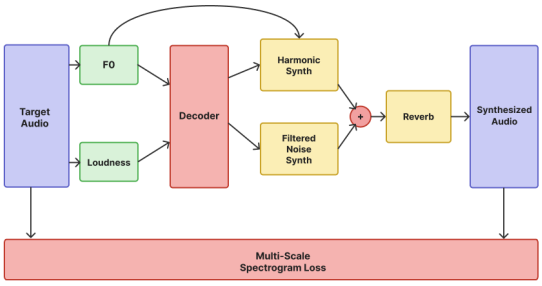


Fig. 1: DDSP Harmonic + Noise model, based on the diagram of Engel et al. (2020)

Finally, the harmonic signal can be parametrised as

$$x(n) = \sum_{k=1}^{K} A_k(n) sin(2\pi f_k(n) + \phi_{0,k})$$

where $\phi_{0,k}$ is the initial phase for the harmonic k, which we fix to 0. The learnt parameters are the global amplitude and the relative amplitudes for each harmonic, per frame.

2) Subtractive synthesis: The frequency sampling method is employed to characterise the filter used to affect the noise signal. This relies on the frequency-domain transfer of a FIR filter H_l, that is applied to a single frame l of the signal at a time.

$$Y_l = H_l N_l$$

where N_l is the DFT of uniform noise [-1, 1]. The learnt parameters correspond to the magnitude values per frequency band that form the frequency response of the filter, per frame.

3) Reverb: Reverberation is designed as a convolutional operation, and implemented by convolving the signal with the reverb module's impulse response. The learnt parameters correspond to the noise samples, decay rate, and wet/dry mix level that are later for computing the impulse response of the reverb module.

B. Morphing framing

DDSP models allow timbre transfer from a monophonic instrument audio input to the timbre learnt by a given model. In this work, we want to extend the possibilities so that given two DDSP models, each representing an instrument, we can perform the timbre transfer to a hybrid instrument obtained by morphing between both timbres. Assuming a linear trajectory between them, we will control where in that line the morphing will take place with the morphing factor α: if we are morphing from a violin to a trumpet, $\alpha = 0$ will correspond to the violin, and $\alpha = 1$ will correspond to the trumpet. $\alpha = 0.5$ should correspond to the hybrid instrument at the midpoint between the two timbres if the effects of the interpolation were perceived to be linear. The interpolation occurs at different points in the architecture, depending on the morphing method, as detailed in the following sections.

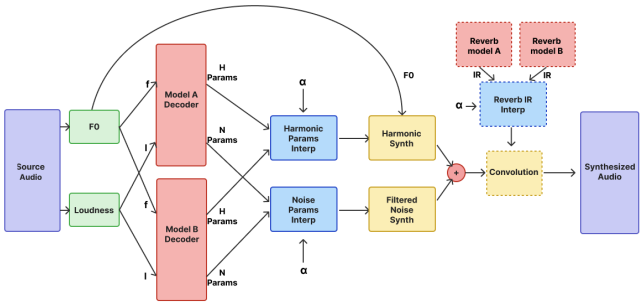


Fig. 2: Diagram for output parameter interpolation

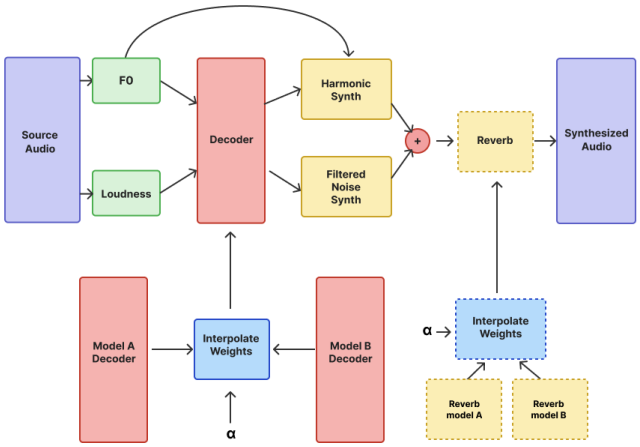


Fig. 3: Diagram for weights interpolation

C. Interpolation of synthesiser parameters

This first morph is obtained by linearly interpolating the predicted parameters for DDSP's Harmonic + Noise model of two separate models, and then synthesising the resulting audio. In the case that reverb is desired to be added, the impulse responses of both reverb blocks are linearly interpolated as well. We will refer to this morphing method as *parameters interpolation* in the rest of the text, and it is described as a diagram in Figure 2.

D. Interpolation of weights

Based on the requirements discussed in II-C, this method is only expected to produce meaningful results if the two models that are being interpolated were fine-tuned from the same pre-trained model. To assess if the interpolation of weights results in a perceivable interpolation of resulting timbres, it is proposed to explore the simplest interpolation possible: linearly interpolate the weights of the learned layers of the model, namely the MLPs and the GRUs of the Decoder, and the noise, decay and wet parameters of the Reverb module. We will refer to this morphing method as *weights interpolation* in the rest of the text, and it is depicted in diagram form in Figure 3.

E. Dataset

This work was realised using the University of Rochester Multi-Modal Music Performance (URMP) Dataset (Li et al., 2018), since it was desired to use expressive recordings of similar nature to the ones that could be used when working with music, as opposed to datasets containing single notes such as NSynth (Engel et al., 2017). Only the audio samples of the dataset were used, comprising ensembles playing classical music, with instruments separately recorded with a sampling rate of 48kHz, 24-bit depth, and in mono. The included instruments can be divided into strings (violin, viola, cello and double bass), woodwinds (flute, oboe, clarinet, bassoon, soprano saxophone and alto saxophone) and brass (trumpet, horn, trombone and tuba). From each family, the instrument with the most recorded material was selected as a representative for the merging experiments: violin, flute, and trumpet.

As it is noted by Wu et al. (2021), we must be certain of not including the same piece played by different instruments in both training and test sets, which would cause data leakage. While they propose to split the dataset based on a random shuffle of the recordings and then adjusting the split ad-hoc, an automated split algorithm was created so that the dataset is divided automatically into train and test sets at the piece level, maintaining a desired ratio between both on a recording level.

The resulting division yields the train and test datasets with durations and file counts as shown in Table I. The training data was used to train the models, while the test dataset was used to generate the audio examples that were later evaluated. In addition, the dataset was preprocessed by normalising the audio to -1dB and filtering with a high pass filter at 80Hz in order to remove low frequency noise.

Instrument	Train Duration (Count)	Test Duration (Count)
Violin (vn)	**0:37:54 (24)**	**0:23:02 (10)**
Viola (va)	0:12:08 (8)	0:13:55 (5)
Cello (vc)	0:16:32 (9)	0:05:53 (2)
Double Bass (db)	0:03:58 (2)	0:02:08 (1)
Flute (fl)	**0:28:05 (13)**	**0:04:47 (5)**
Oboe (ob)	0:10:22 (4)	0:01:19 (2)
Clarinet (cl)	0:15:17 (8)	0:02:48 (2)
Bassoon (bn)	0:02:54 (1)	0:01:19 (2)
Saxophone (sax)	0:15:08 (10)	0:00:39 (1)
Trumpet (tpt)	**0:31:39 (15)**	**0:09:24 (7)**
Horn (hn)	0:07:09 (2)	0:03:02 (3)
Trombone (tbn)	0:14:10 (5)	0:03:02 (3)
Tuba (tba)	0:08:02 (4)	0:00:53 (1)
Total	**3:23:18 (105)**	**1:11:31 (44)**

TABLE I: Train/test split durations and file count by instrument

F. Types of models

We train three types of models, since weights interpolation requires two models fine-tuned from the same pre-trained model (base model). *Trained-from- scratch* models for violin, flute, and trumpet are trained separately from random initialisation. The **base model** is trained on all instrument data except violin, flute and trumpet, so it can later be fine-tuned. **Fine-tuned** models start from the base model checkpoint and are separately fine-tuned on violin, flute and trumpet data

G. Training of models

Initial tests included training for 100k steps, as suggested by the original default in the implementation used as a starting point for this project[6], which led to overfitting to noise artefacts present in the training data. The analysed models were therefore trained for a maximum of 30k steps, as recommended in the training notebook provided by the research team of Engel et al. (2020)[7]. The models are trained with a multiscale STFT loss with 2048, 1024, 512, 256, 128, and 64 sample windows, using an ADAM optimiser and a learning rate that decays exponentially every 10k steps. Training a model for 30k steps with a batch of 16 takes 2:40 hours on a JupyterLab server with a shared NVIDIA A40 GPU.

For the base model, as a means of incorporating domain data from orchestral instruments, we train with data for all instruments except violin, flute and trumpet. The data is shuffled randomly during training, so that the model learns from the average information instead of focusing on a single instrument at a time. For the results presented in this paper, the base model was trained for 30k steps, and all fine-tuned models were trained for 30k steps starting from the base model checkpoint.

H. Generating audio results

Audio is generated for all pairs, where we compute morphs between instrument A and instrument B. Given each pair, we process each test file for instrument A and generate:

1) the resynthesised audio obtained with the instrument A model
2) the synthesised audio obtained with the instrument B model
3) the synthesised timbre interpolation audio for $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$, for both weight interpolation and parameters interpolation methods

These outputs are generated both for the trained-from-scratch models and the fine-tuned models respectively, for further comparison.

I. Code

The code developed during this work[8] took the DDSP Pytorch implementation by IRCAM-ACIDS[9] as a base. This is stated to be a Pytorch implementation of the system described in (Engel et al., 2020).

[6]https://github.com/acids-ircam/ddsp_pytorch

[7]https://colab.research.google.com/github/magenta/ddsp/blob/main/ddsp/colab/demos/Train_VST.ipynb

[8]The codebase can be found in: https://github.com/jimearruti/ddsp_interpolate

[9]https://github.com/acids-ircam/ddsp_pytorch

Two major bugs were addressed as discussed in the reference repository Issues[10]. Additionally, the learning rate during training was changed to comply with (Engel et al., 2020), and the calculation of standard deviation for the datasets was corrected. The code was adapted to enable training fine-tuning, as well as either training models for individual instruments or for a combination of instruments for the base models. Monitoring for training was added to use Weights and Biases[11], visualising training loss among other parameters. Additionally, all the code regarding interpolation and audio generation is original.

IV. Evaluation

A. Objective metrics

For objective metrics, we followed the methodology outlined by (Caspe et al., 2025), where the authors evaluate the performance of their timbre transfer model while noting that this is a task for which no paired ground truth is available. They were able to analyse the audio quality and timbral changes the model causes, relying on statistical and probabilistic approaches. On the other hand, their deterministic loudness and fundamental frequency assessment is not reproduced in this work, since our model is trained by seeing that information explicitly.

1) Audio quality: Fréchet Audio Distance (FAD) (Kilgour et al., 2018) was originally designed as a metric for music enhancement tasks, and has been widely adopted in generative audio to measure audio quality. Instead of comparing obtained results to their paired ground truth, FAD compares two statistical distributions: one of a reference or background set (for example the train set), and one of the generated results. The embeddings corresponding to audio clips of the two sets are obtained using a pre-trained audio embedding model such as VGGish. Then each set's embeddings are fitted to a Gaussian distribution, and the Fréchet Distance is calculated between the two distributions.

For each pair of instruments, we calculate FAD by having the joint training data of both instruments as background, and the following as generated sets:

- Resynthesis with the model corresponding to the original instrument of each recording
- Results obtained for morphing between the models, separating between edge cases ($\alpha \in \{0, 1\}$) and true interpolation ($\alpha \in \{0.25, 0.50, 0.75\}$)

This is calculated separately for the fine-tuned and trained-from-scratch models with both interpolation methods. The distance to the test set for the same instruments is included as a baseline. It is worth noting that the flute-trumpet test dataset is less than the recommended minimum data for using FAD (25 minutes), so the results need to be treated as suggestive rather than conclusive.

[10]https://github.com/acids-ircam/ddsp_pytorch/issues/37 and https://github.com/acids-ircam/ddsp_pytorch/issues/31

[11]https://wandb.ai/site/

Additionally, for each instrument pair, for the fine-tuned model with the weights interpolation and the trained-from-scratch model with the parameters interpolation, FAD was calculated with the morphs of edge cases as a baseline and the morphs with true interpolation as the generated results. This was done to enable the direct comparison with the subjective metrics.

2) Timbre: Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) is a two-sample statistical test that allows assessing if two sets of samples are drawn from different distributions. Following the procedure described in (Caspe et al., 2025), MMD is calculated for sets of *texture windows*, obtained by calculating the Mel-frequency cepstral coefficients (MFCCs) of all recordings in a given audio corpus and dividing them into windows. When calculated between two audio corpora, a greater MMD value means it is less likely that the audios of the two sets belong to the same timbral distribution.

For this case, a more granular division of the results was performed. For each directed pair of instrument morphs, instrument A $\rightarrow$ instrument B, the timbre distribution computed on the train set is compared to the timbre distribution of the morphs for the test set of instrument A, calculated separately for $\alpha = [0, 0.25, 0.5, 0.75, 1]$. This calculation is repeated for the fine-tuned and trained-from-scratch models, and for both interpolation methods.

B. Subjective evaluation

To gauge the subjective perception of the results, a listening test was devised by adapting the MUSHRA framework (International Telecommunication Union, 2015), and implemented by modifying webMUSHRA (Schoeffler et al., 2018)[12]. The test was designed to compare the results obtained by interpolating between weights of two fine-tuned models, with the ones obtained by interpolating between synthesis parameters of two models trained from scratch. This decision was made as they are the configurations most likely to become use cases, and it was designed to last approximately 20 minutes. Note that this means model type and interpolation method are entangled in the comparisons that follow.

For each possible pair of morphs, instrument A $\rightarrow$ instrument B, one short[13] fragment of a test recording from instrument A was selected, and two test screens were constructed, one for each interpolation method. The fragments were chosen considering their expressivity and audio quality by listening to the DDSP resynthesis results. In order to only evaluate the morphing and not the base DDSP timbre transfer capabilities, the original recordings are not shown to the participants at any time.

In each test screen, two reference tracks are presented, one consisting of the morph at $\alpha = 0$, labelled as instrument A,

[12]The code can be found at https://www.github.com/jimearruti/webMUSHRA and the listening test can be accessed at https://listening-test.jimearruti.com/?config=morphing.yaml

[13]The fragments last around 5-7 seconds, enough to reflect a musical phrase or idea

and another consisting of the morph at $\alpha = 100$, labelled as instrument B. These are used as hidden references in the test stimuli. Additionally, another three test stimuli are included, consisting of the morphs obtained for $\alpha = [0.25, 0.50, 0.75]$. The five stimuli are presented in random order, and must be rated by the participants in a line between instrument A (marked as 0) and instrument B (marked as 100). Additionally, they were asked to rate the sound quality of the stimuli with respect to the sound quality of the references on a Mean Opinion Score (MOS) scale ranging from Excellent to Bad. Screenshots of the test screens can be seen in Appendix A. All stimuli are normalised to -14 LUFS in order to ensure they present the same loudness.

A questionnaire is also included to collect demographic information including age group, gender, type of musical experience and years of musical experience. Space for optional comments was included on all screens.

V. Results

A. Objective metrics

1) Audio quality: For all pairs of instruments reported in Figure 4, the fine-tuned model's resynthesis shows a higher FAD than the trained-from-scratch model's, which indicates further research needs to be done to find the best fine-tuning configuration. Additionally, both methods of interpolation stay close to their respective resynthesis values at the edge cases $\alpha \in \{0, 1\}$, indicating similar instrument reconstruction capabilities within each model. This pattern reverses sharply at intermediate α values, however, where it is evident that the weights interpolation method does not work for the model trained from scratch, as hypothesised, since the FAD for that case is several points above the other cases. For the fine-tuned model, on the other hand, the performance of the weights interpolation seems to be slightly better for the flute-trumpet and flute-violin cases, while the contrary happens for the trumpet-violin cases. Particularly, for the flute-trumpet pair, the performance of the weights interpolation is similar for both groups of α.

Instrument Pair	Model/Interpolation method	FAD
Flute↔Trumpet	Fine-tuned/Weights	**0.3831**
	From Scratch/Parameters	0.5899
Flute↔Violin	Fine-tuned/Weights	**0.6599**
	From Scratch/Parameters	1.5838
Trumpet↔Violin	Fine-tuned/Weights	**0.5364**
	From Scratch/Parameters	1.0273

TABLE II: Fréchet Audio Distance (FAD) per instrument pair, model and interpolation method, as is experienced in the listening test. Calculation is done with the results of edge cases of α as background, and for the results of interpolated α values as a generated set. Lower scores are better as they show less degradation.

Additionally, the results of Table II suggest there is a greater degradation of quality when interpolating parameters for trained-from-scratch models when compared to interpolating weights of fine-tuned models.

Pair	Model / Interpolation method	Intermediate / Total	Proportion
fl↔tpt	From-Scratch/Parameters	12/26	46.2%
	Fine-tuned/Weights	20/26	**76.9%**
fl↔vn	From-scratch/Parameters	21/26	**80.8%**
	Fine-tuned/Weights	18/26	69.2%
tpt↔vn	From-scratch/Parameters	18/26	**69.2%**
	Fine-tuned/Weights	16/26	61.5%
Overall	**From-scratch-Parameters**	**51/78**	**65.4%**
	Fine-tuned/Weights	**54/78**	**69.2%**

TABLE III: Trials where intermediate alphas were perceived as falling between the extreme alphas, as a proportion of total trials, by unordered instrument pair and interpolation method.

2) Timbre: The results seen in Figure 5 show that for the weight interpolation applied to the fine-tuned models, the MMD metric monotonically approaches or grows distant from the reference instrument, except for the flute→trumpet (with respect to the trumpet) and violin→flute (with respect to the violin) cases: the closer α is to the reference instrument, the greater the chance the morphing results share its timbral distribution. For the parameter interpolation applied to the trained-from-scratch model, these changes are not monotonic except for violin→trumpet (with respect to the trumpet) and, more marginally, trumpet→violin (with respect to the trumpet). However, the results show non-monotonicity in the same two cases for both interpolation methods when applied to the fine-tuned model — markedly fewer than the trained-from-scratch case. This suggests there may be a benefit to applying the interpolated parameters method to fine-tuned models compared with trained-from-scratch models, but more research is needed to draw final conclusions.

B. Subjective metrics

The listening test was completed by 14 participants, out of which one needed to be removed for rating all stimuli with 100 and all audio qualities as 5. The audio quality was only analysed for 12 of the remaining participants, since one had participated in a pilot before final modifications to the question asked.

1) Timbre Perception: Table III shows that for the majority of the trials, the intermediate values of α were perceived to have a timbre that was in between the timbre of the references, with interpolating weights of fine-tuned models causing less confusion than the interpolation of parameters of trained-from-scratch models on average. In a test that was designed with hidden references to set the extreme values, however, having over 30% of incorrectly set references could mean the task is too hard, or the test was not completed with enough attention. For the analyses that follow, the data was cleaned by filtering out trials which had a rating of over 20 for the reference for $\alpha = 0$, or a rating of less than 80 for the reference for $\alpha = 100$.

As shown in Table IV, the perception of morphing does not always vary in a monotonic way, indicating the stimuli are ordered in an unexpected way for 44% of the parameter

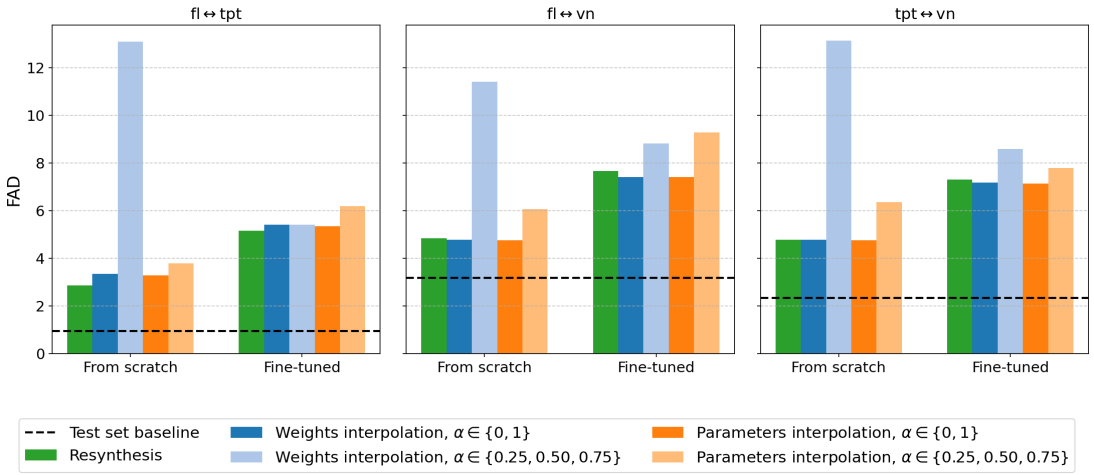


Fig. 4: Fréchet Audio Distance to the joint test sets of instrument A and instrument B, for morphs in both directions between instrument A and instrument B. VGGish embeddings. Lower scores are better.

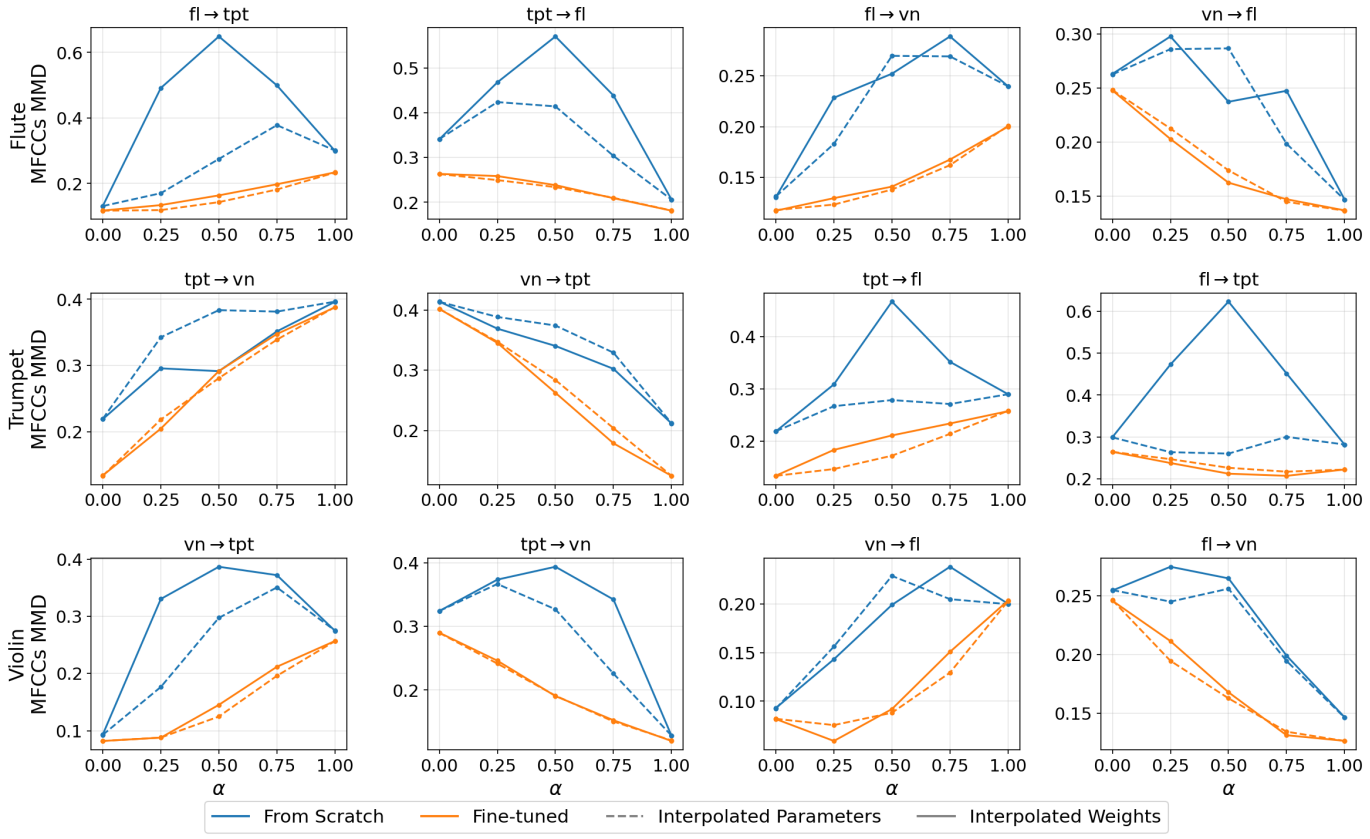


Fig. 5: Results for MMD calculated on texture windows of MFCC features by model and morphing method. The reference audio corpus is the train set of the instrument corresponding to each row. The pair of instruments involved in the morphing and the direction of the morph is on top of each subplot.

Pair	Model / Interpolation method	Non-monotonic / Total (Filtered)	Proportion
fl↔tpt	From Scratch/Parameters	8/15	**53.3%**
	Fine-tuned/Weights	13/21	61.9%
fl↔vn	From Scratch/Parameters	7/22	31.8%
	Fine-tuned/Weights	5/21	**23.8%**
tpt↔vn	From Scratch/Parameters	11/22	50.0%
	Fine-tuned/Weights	5/18	**27.8%**
Overall	**From Scratch/Parameters**	**26/59**	**44.1%**
	Fine-tuned/Weights	**23/60**	**38.3%**

TABLE IV: Non-monotonic trials as a proportion of trials remaining after filtering, by unordered instrument pair and interpolation method.

interpolation and 38% of the weights interpolations. This effect seems to be more extreme for the flute-trumpet morphs, and less prominent for flute-violin morphs.

2) Audio quality: The audio quality ratings descriptive statistics comparing both interpolation methods over all instrument pairs and overall are shown in Table V. The MOS results show that both model-interpolation method pairs present some audio quality degradation with respect to the references, with interpolating parameters showing slightly better audio quality than interpolating weights overall. However, we must consider that 12 participants are not enough to confirm any trends, and an Omnibus repeated measures ANOVA reported in Appendix F showed these results are not statistically significant. On an exploratory note, for the current data, when considering different pairs of instruments separately, the interpolation of parameters seems to result in a better audio quality for flute-violin and trumpet-violin pairs, while the opposite happens for flute-trumpet.

Pair	Model / Interpolation method	Mean	Std
fl↔tpt	From Scratch/Parameters	3.71	0.86
	Fine-tuned/Weights	**3.92**	0.72
fl↔vn	From Scratch/Parameters	**3.92**	0.78
	Fine-tuned/Weights	3.62	1.13
tpt↔vn	From Scratch/Parameters	**4.04**	0.69
	Fine-tuned/Weights	3.83	1.05
Overall	**From Scratch/Parameters**	**3.89**	**0.78**
	Fine-tuned/Weights	**3.79**	**0.98**

TABLE V: Mean audio quality rating by interpolation method, overall and by instrument pair (any direction), for 12 participants. 5 is Excellent and 1 is Bad.

VI. Discussion

A. Limitations

The FAD calculations were done for less data than the minimum recommended, and therefore they can only be taken as a rough guide. Likewise, the listening test was completed by 14 participants, out of which only 12 were considered for the audio quality ratings, and 13 for the timbre ratings. Moreover, the analysis was done for one fragment of audio for each directed morph between an instrument pair, which permits observing effects but not making generalised affirmations. Most of the findings in this paper, therefore, are not to be taken as confirmed facts, but as hypotheses built to explore further. Additionally, the fine-tuned models used for the evaluation were always trained for 30k steps, starting from a base model trained for 30k steps. It is expected that different results would be obtained for different configurations, particularly regarding audio quality. Finally, the listening test compared only two of the four possible method/model-type combinations (weights interpolation on fine-tuned models, and parameters interpolation on from-scratch models); weights interpolation on from-scratch models was excluded following informal listening indicating the resulting morphs did not resemble either source timbre, and this design choice means model type and interpolation method remain entangled throughout the subjective results reported below.

B. RQ1: Does interpolating the weights of two DDSP models, or interpolating the synthesis parameters predicted by two DDSP models, result in a perceived morph of the two instruments?

The listening test results confirm the timbre of the weights interpolation method of two fine-tuned models for intermediate values of α is perceived to be in between edge values of α for the majority of the trials (69%). This technique does not work for models trained from scratch, as shown by their higher FAD in Figure 4, consistent with prior findings on the shared error-landscape basin of fine-tuned models. For the case of parameter interpolation for models trained from scratch, this is also supported by 65% of the trials. All the generated audio is included in the supplementary material, allowing readers to independently assess the results.

C. RQ2: When the interpolation of the weights or the parameters is linear, how does the timbre of the resulting morphing behave?

The listening test results show there is a relatively high degree of non-monotonicity in the ordering of the stimuli (44.1% for the parameter interpolation method, and 38.3% for the weights interpolation method, after filtering the trials where the hidden references were not identified correctly). This shows there is some degree of confusion regarding the ordering of the stimuli for different values of α in the morphing test screens. Considering a listening test design where the order of the stimuli is fixed could help assess the morphing effects more reliably.

On the other hand, the MMD results show that for weights interpolation applied to fine-tuned models, timbral non-monotonicity occurred in 2 out of 12 (16%) of the directed-pair/reference-instrument combinations, and parameters interpolation applied to fine-tuned models showed the same pattern. By contrast, parameters interpolation applied to trained-from-scratch models was non-monotonic in 10 of 12 (83%) combinations, which indicates there may be a benefit in fine-tuning the model when using this method.

As a next step, generating results for a greater number of intermediate α values, or a sweeping value of α, could also

yield more information regarding the timbre of the morphing results.

D. RQ3: Is there audio quality degradation when applying the proposed methods for morphing?

When comparing the audio quality of the morphing for $\alpha \in \{0.25, 0.50, 0.75\}$ to the morphing for $\alpha \in \{0, 1\}$, the listening test confirmed there is a perceived degradation of audio quality for intermediate values of α. Additionally, the results suggest that the morphing by interpolation of parameters for the model trained from scratch might achieve a better audio quality than the morphing by interpolated weights for this particular fine-tuned model overall. FAD, however, suggests that there is less degradation for the fine-tuned case. Further evaluation is needed to compare the two methods' performance, as we do not have enough data to make final claims.

FAD results analysis shows the fine-tuned model performs worse than the model trained from scratch when comparing the obtained audio quality of the audio in the train sets overall. This should not be read as a problem with fine-tuned models in general, since it should be considered that we have not explored how to perform the fine-tuning to achieve the best possible model. To fully disentangle model performance from interpolation-method performance, it would be necessary to repeat the listening test and evaluate both interpolation methods only on the fine-tuned model.

E. RQ4: How do the two methods compare overall?

Taken together, the metrics reported above do not favour one method consistently: weights interpolation tends to edge ahead on audio quality (FAD), while parameters interpolation performs slightly better on perceived audio quality (MOS) and ordering consistency in the listening test. Notably, the two methods showed similar rates of non-monotonic timbral progression (MMD) when applied to the same (fine-tuned) model type, suggesting model type—rather than interpolation method—may be the larger driver of monotonic timbral behaviour. On an architectural level, while parameters interpolation allows the simpler option of using models trained from scratch, it requires both models to compute their parameter predictions at inference time, which is likely to be less computationally efficient. Additionally, we hypothesise that the weight interpolation method may enable more temporally dynamic interpolations because the neural networks themselves are being morphed, although validating this hypothesis is beyond the scope of this work.

VII. Conclusions

This work explored weight and parameter interpolation between DDSP Harmonic + Noise models as retraining-free approaches to cross-instrument timbre morphing. Weight interpolation produced perceptually intermediate timbres only when both models were fine-tuned from a common base, while parameter interpolation worked for independently trained models as well. Both methods showed perceived quality degradation within acceptable ratings, and non-monotonic timbral progression for a substantial minority of trials, while neither dominated consistently across metrics. Given the modest sample size, these results are suggestive rather than conclusive, and future work should test a larger participant pool, a finer granularity of α, and a full crossing of interpolation method and model type.

VIII. Ethical standards

The listening test was conducted in accordance with Queen Mary University of London policies and data protection guidelines. The protocol was approved by Queen Mary University of London School of Electronic Engineering and Computer Science (EECS) Research Ethics Committee (DSREC) (reference: QME26.00055.DSEECS26.055). Participants provided informed consent and participation was voluntary.

Claude and GitHub Copilot were used for coding assistance and help with minor tasks as disclosed in the Gen AI statement.

Acknowledgments

We would like to thank Charalampos Saitis, Franco Caspe, Haokun Tian, Pablo Tablas de Paula and Aditya Bhattacharjee for the fruitful discussions on topics related to the project.

The research that gives rise to the results presented in this publication received funding from Uruguay's National Agency for Research and Innovation (ANII), under code POS_EXT_2024_1_185707.

References

Caetano, M. F. and Rodet, X. (2010), Automatic timbral morphing of musical instrument sounds by high-level descriptors, *in* 'International Computer Music Conference', pp. 11–21.

Caetano, M. and Rodet, X. (2013), 'Musical instrument sound morphing guided by perceptually motivated features', *IEEE Transactions on Audio, Speech, and Language Processing* **21**(8), 1666–1675.

Caillon, A. and Esling, P. (2021), 'Rave: A variational autoencoder for fast and high-quality neural audio synthesis', *arXiv preprint arXiv:2111.05011* .

Çakir, E. and Virtanen, T. (2018), Musical instrument synthesis and morphing in multidimensional latent space using variational, convolutional recurrent autoencoders, *in* 'Audio Engineering Society Convention', AES Audio Engineering Society, p. 10035.

Caspe, F., Shier, J., Sandler, M., Saitis, C. and McPherson, A. (2025), 'Designing neural synthesizers for low-latency interaction', *arXiv preprint arXiv:2503.11562* .

Engel, J., Hantrakul, L., Gu, C. and Roberts, A. (2020), 'Ddsp: Differentiable digital signal processing', *arXiv preprint arXiv:2001.04643* .

Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D. and Simonyan, K. (2017), Neural audio synthesis of musical notes with wavenet autoencoders, *in* 'International conference on machine learning', PMLR, pp. 1068–1077.

Esling, P., Chemla-Romeu-Santos, A. and Bitton, A. (2018), Bridging audio analysis, perception and synthesis with perceptually-regularized variational timbre spaces., *in* 'ISMIR', pp. 175–181.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B. and Smola, A. (2012), 'A kernel two-sample test', *Journal of Machine Learning Research* **13**(25), 723–773.
URL: *http://jmlr.org/papers/v13/gretton12a.html*

Hayes, B., Shier, J., Fazekas, G., McPherson, A. and Saitis, C. (2024), 'A review of differentiable digital signal processing for music and speech synthesis', *Frontiers in Signal Processing* **3**, 1284100.

International Telecommunication Union (2015), Method for the subjective assessment of intermediate quality level of audio systems, Technical report, ITU-R Recommendation BS.1534-3.
URL: *https://www.itu.int/rec/R-REC-BS.1534-3-201510-I/en*

Kilgour, K., Zuluaga, M., Roblek, D. and Sharifi, M. (2018), 'Fréchet audio distance: A metric for evaluating music enhancement algorithms', *arXiv preprint arXiv:1812.08466* .

Kim, J. W., Salamon, J., Li, P. and Bello, J. P. (2018), Crepe: A convolutional representation for pitch estimation, *in* '2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)', IEEE, pp. 161–165.

Li, B., Liu, X., Dinesh, K., Duan, Z. and Sharma, G. (2018), 'Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications', *IEEE Transactions on Multimedia* **21**(2), 522–535.

Li, W., Peng, Y., Zhang, M., Ding, L., Hu, H. and Shen, L. (2023), 'Deep model fusion: A survey', *arXiv preprint arXiv:2309.15698* .

Neyshabur, B., Sedghi, H. and Zhang, C. (2020), 'What is being transferred in transfer learning?', *Advances in neural information processing systems* **33**, 512–523.

Nistal, J., Lattner, S. and Richard, G. (2020), 'Drumgan: Synthesis of drum sounds with timbral feature conditioning using generative adversarial networks', *arXiv preprint arXiv:2008.12073* .

Schoeffler, M., Bartoschek, S., Stöter, F.-R., Roess, M., Westphal, S., Edler, B. and Herre, J. (2018), 'webmushra—a comprehensive framework for web-based listening tests', *Journal of open research software* **6**(1), 8–8.

Serra, X. and Smith, J. (1990), 'Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition', *Computer Music Journal* **14**(4), 12–24.
URL: *http://www.jstor.org/stable/3680788*

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S. et al. (2022), Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, *in* 'International conference on machine learning', PMLR, pp. 23965–23998.

Wu, Y., Manilow, E., Deng, Y., Swavely, R., Kastner, K., Cooijmans, T., Courville, A., Huang, C.-Z. A. and Engel, J. (2021), 'Midi-ddsp: Detailed control of musical performance via hierarchical modeling', *arXiv preprint arXiv:2112.09312* .

APPENDIX A
LISTENING TEST SCREENS

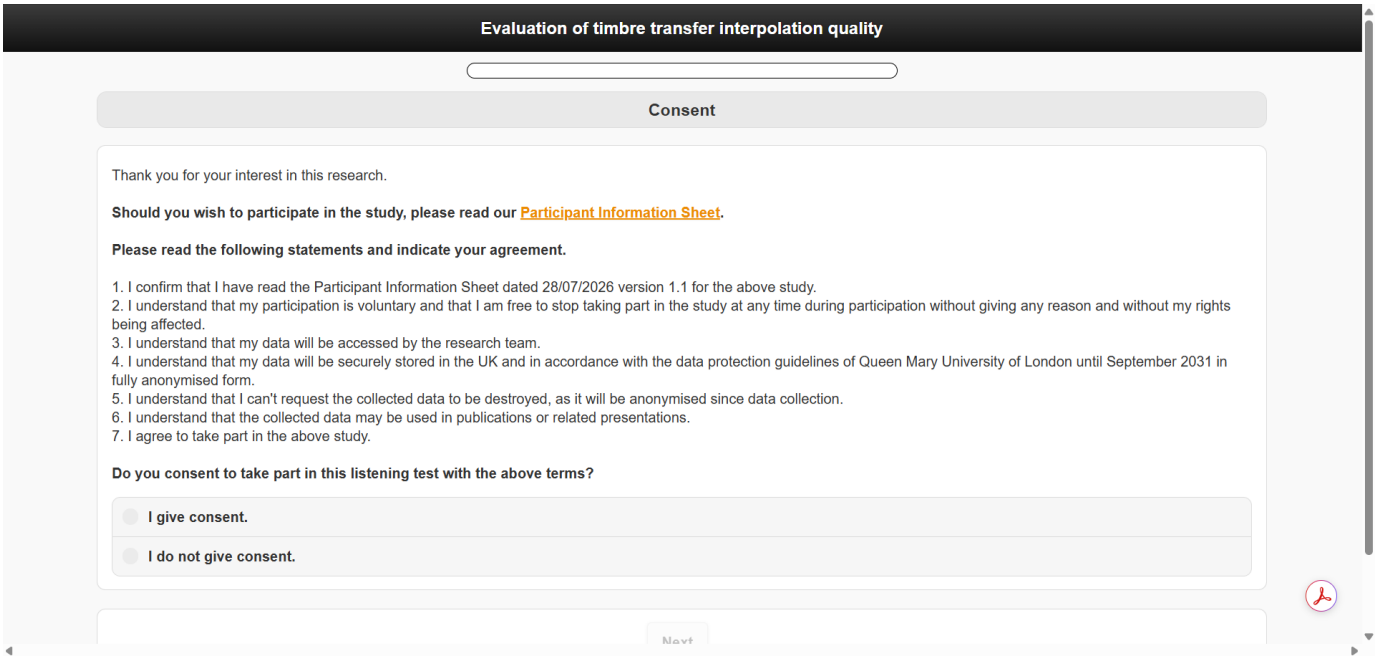


Fig. 6: Screenshot of listening test consent screen.

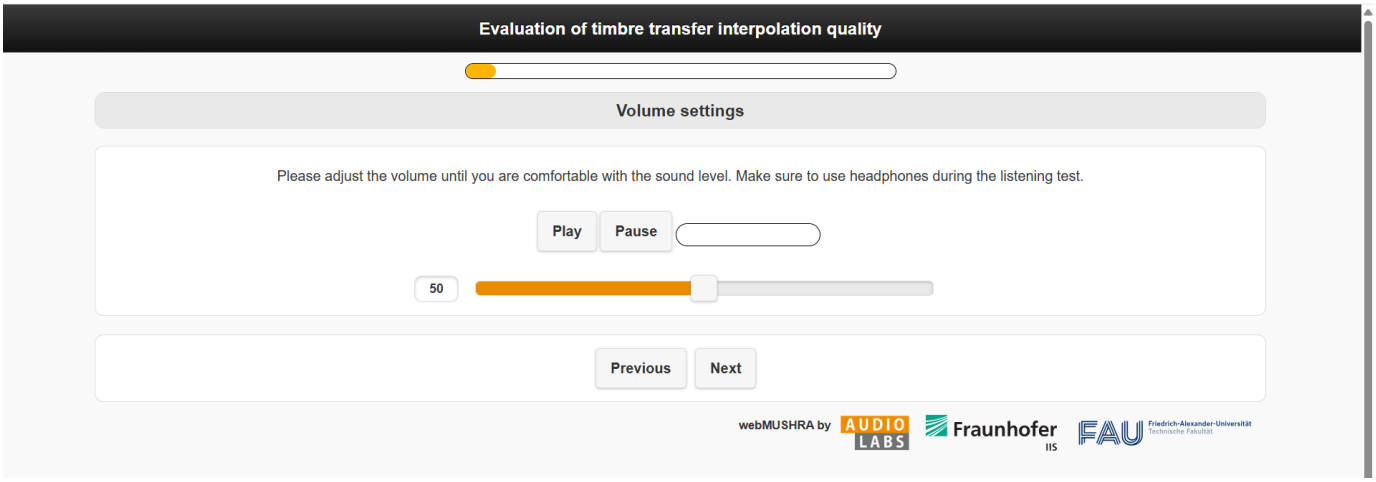


Fig. 7: Screenshot of listening test volume adjustment screen.

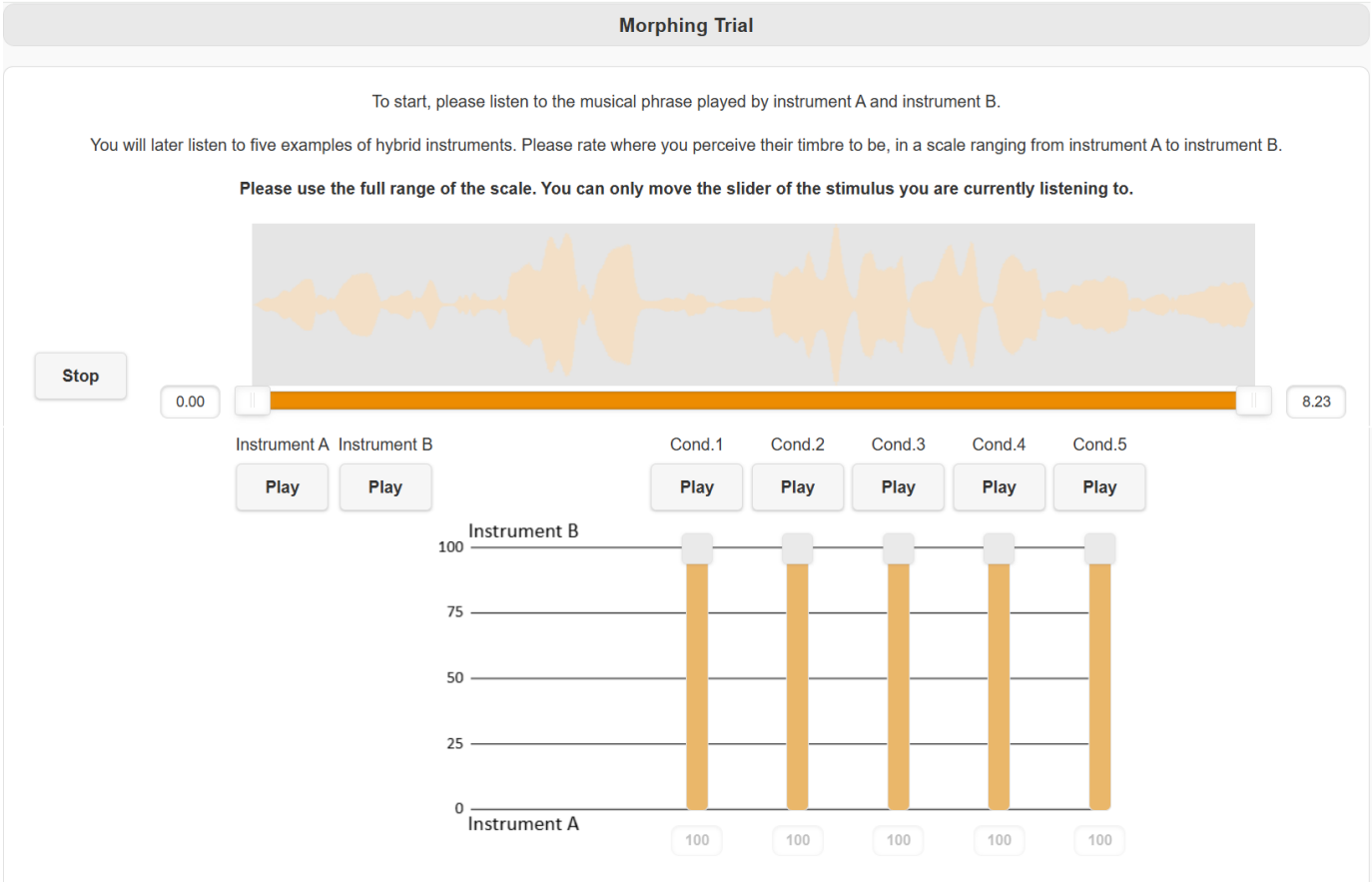


Fig. 8: Screenshot of listening test morphing trial screen.

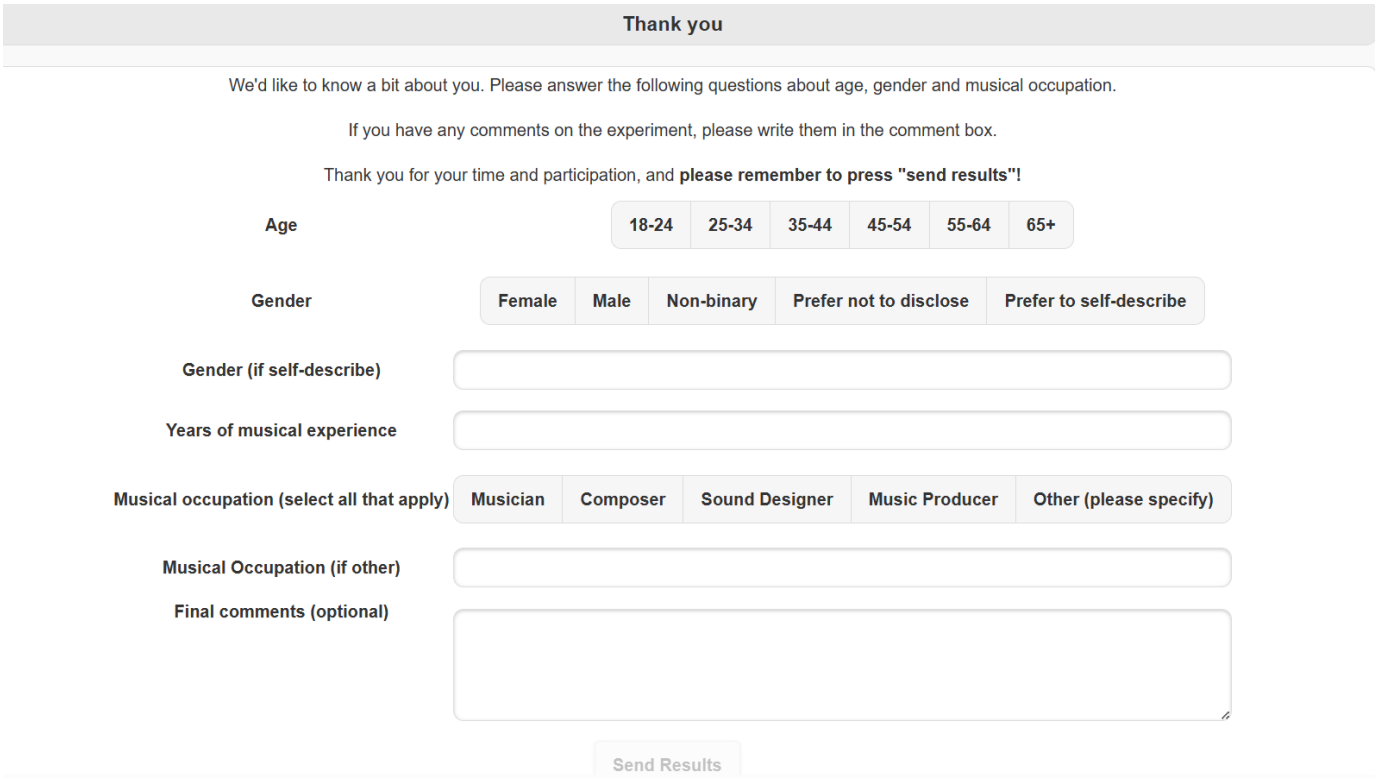


Fig. 9: Screenshot of listening test demographic questionnaire screen.

APPENDIX B
DECODER ARCHITECTURE

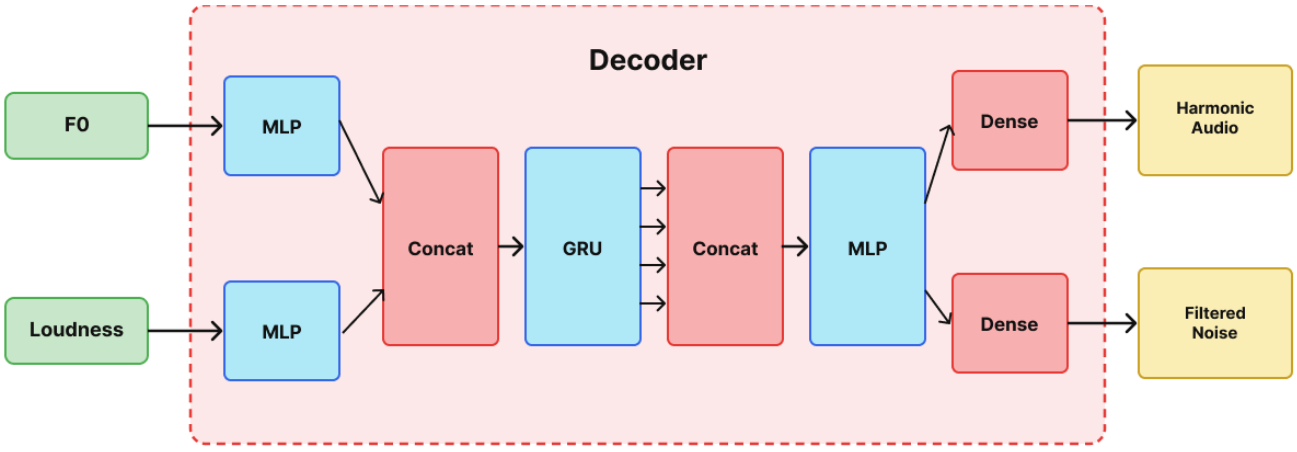


Fig. 10: Model's decoder internal architecture. Image based on diagram from (Engel et al., 2020).

APPENDIX C
TRAINING LOSS PER MODEL

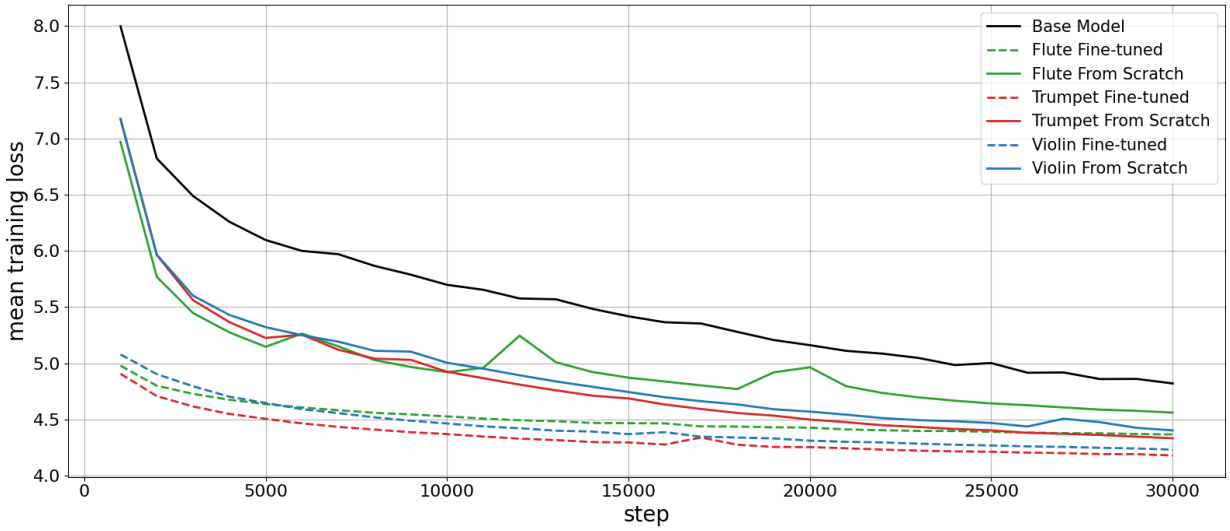


Fig. 11: Training loss per model, for 30k steps

APPENDIX D
FRÉCHET AUDIO DISTANCE RESULTS

TABLE VI: Fréchet Audio Distance (FAD) across evaluation categories, by instrument pair, model and interpolation method. Background is the corresponding train set of both instruments in the pair of each row.

Instrument Pair	Model	Test	Resynth.	Interp. Weights		Interp. Parameters	
				$\alpha \in \{0, 1\}$	$\alpha \in \{0.25, 0.5, 0.75\}$	$\alpha \in \{0, 1\}$	$\alpha \in \{0.25, 0.5, 0.75\}$
Flute↔Trumpet	Fine-tuned	0.9448	5.1444	5.4007	5.4081	5.3543	6.1912
	From Scratch		2.8617	3.3423	13.0897	3.2896	3.7843
Flute↔Violin	Fine-tuned	3.1670	7.6574	7.3961	8.8249	7.4085	9.2811
	From Scratch		4.8486	4.7805	11.4043	4.7484	6.0626
Trumpet↔Violin	Fine-tuned	2.3433	7.2982	7.1806	8.5879	7.1409	7.7754
	From Scratch		4.7758	4.7776	13.1214	4.7554	6.3573

APPENDIX E
MFCCS MAXIMUM MEAN DISCREPANCY RESULTS

TABLE VII: Reference MMD to the Violin distribution. *Test* is the MMD for real test recordings of the source instrument; *Resynth* is the MMD from that instrument's resynthesis, for each model. Lower means they are more likely to belong to the same timbral distribution.

Source	Test	Resynth (From Scratch)	Resynth (Fine-tuned)
Violin	0.012	0.096	0.082
Trumpet	0.371	0.282	0.258
Flute	0.239	0.203	0.199

TABLE VIII: MFCCs MMD to the train violin dataset distribution for morphs between each instrument pair, across interpolation weights α, comparing models trained from scratch versus fine-tuned. Lower means they are more likely to belong to the same timbral distribution.

		From Scratch		Fine-tuned	
Pair	α	Parameters	Weights	Parameters	Weights
vn→tpt	0	0.093	0.093	0.082	0.082
	0.25	0.177	0.330	0.088	0.088
	0.5	0.297	0.387	0.125	0.145
	0.75	0.351	0.372	0.196	0.212
	1	0.275	0.275	0.257	0.257
tpt→vn	0	0.324	0.324	0.290	0.290
	0.25	0.366	0.373	0.241	0.246
	0.5	0.327	0.394	0.191	0.190
	0.75	0.226	0.342	0.151	0.153
	1	0.128	0.128	0.121	0.120
vn→fl	0	0.092	0.093	0.082	0.082
	0.25	0.156	0.143	0.075	0.059
	0.5	0.229	0.199	0.088	0.092
	0.75	0.205	0.238	0.129	0.151
	1	0.200	0.200	0.203	0.203
fl→vn	0	0.255	0.254	0.246	0.246
	0.25	0.245	0.275	0.195	0.211
	0.5	0.256	0.265	0.163	0.168
	0.75	0.195	0.199	0.134	0.131
	1	0.146	0.147	0.126	0.127

TABLE IX: Reference MMD to the Flute distribution. *Test* is the MMD for real test recordings of the source instrument; *Resynth* is the MMD for that instrument's resynthesis, for each model. Lower means they are more likely to belong to the same timbral distribution.

Source	Test	Resynth (From Scratch)	Resynth (Fine-tuned)
Flute	0.040	0.147	0.129
Trumpet	0.299	0.289	0.217
Violin	0.165	0.255	0.238

TABLE X: MFCCs MMD to the train flute dataset distribution for morphs between each instrument pair, across interpolation weights α, comparing models trained from scratch versus fine-tuned. Lower means they are more likely to belong to the same timbral distribution.

		From Scratch		Fine-tuned	
Pair	α	Parameters	Weights	Parameters	Weights
fl→tpt	0	0.131	0.131	0.117	0.117
	0.25	0.170	0.492	0.119	0.134
	0.5	0.275	0.649	0.143	0.164
	0.75	0.378	0.500	0.182	0.198
	1	0.301	0.300	0.233	0.234
tpt→fl	0	0.341	0.341	0.263	0.264
	0.25	0.424	0.468	0.250	0.259
	0.5	0.414	0.571	0.234	0.238
	0.75	0.304	0.439	0.210	0.209
	1	0.206	0.206	0.182	0.181
fl→vn	0	0.131	0.131	0.118	0.117
	0.25	0.183	0.228	0.123	0.130
	0.5	0.270	0.252	0.138	0.141
	0.75	0.269	0.289	0.162	0.168
	1	0.240	0.240	0.201	0.200
vn→fl	0	0.262	0.263	0.248	0.248
	0.25	0.286	0.298	0.212	0.203
	0.5	0.287	0.237	0.174	0.162
	0.75	0.198	0.247	0.145	0.147
	1	0.147	0.147	0.136	0.137

TABLE XI: Reference MMD to the Trumpet distribution. *Test* is the MMD from real test recordings of the source instrument; *Resynth* is the MMD from that instrument's resynthesis, for each model. Lower means they are more likely to belong to the same timbral distribution.

Source	Test	Resynth (From Scratch)	Resynth (Fine-tuned)
Trumpet	0.024	0.212	0.128
Violin	0.348	0.401	0.387
Flute	0.267	0.255	0.218

TABLE XII: MFCCs MMD to the test trumpet dataset distribution for morphs between each instrument pair, across interpolation weights α, comparing models trained from scratch versus fine-tuned. Lower means they are more likely to belong to the same timbral distribution.

		From Scratch		Fine-tuned	
Pair	α	Parameters	Weights	Parameters	Weights
tpt→vn	0	0.220	0.219	0.134	0.134
	0.25	0.343	0.296	0.219	0.205
	0.5	0.384	0.292	0.281	0.292
	0.75	0.381	0.352	0.339	0.348
	1	0.396	0.396	0.388	0.388
vn→tpt	0	0.414	0.414	0.402	0.402
	0.25	0.389	0.369	0.348	0.346
	0.5	0.374	0.341	0.284	0.263
	0.75	0.329	0.302	0.204	0.179
	1	0.212	0.212	0.125	0.125
tpt→fl	0	0.219	0.219	0.134	0.134
	0.25	0.267	0.309	0.147	0.183
	0.5	0.279	0.467	0.172	0.211
	0.75	0.271	0.352	0.214	0.234
	1	0.290	0.290	0.258	0.257
fl→tpt	0	0.299	0.299	0.264	0.264
	0.25	0.264	0.473	0.247	0.238
	0.5	0.260	0.623	0.226	0.212
	0.75	0.300	0.452	0.217	0.207
	1	0.282	0.282	0.222	0.222

APPENDIX F
LISTENING TEST RESULTS

For all tables and figures, parameters interpolation is applied to the model trained from scratch, and weights interpolation is applied to the fine-tuned model.

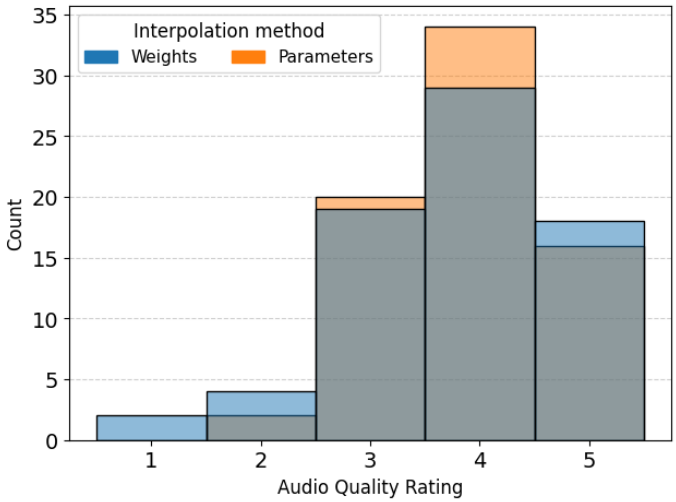


Fig. 12: Histogram of audio quality Ratings by interpolation method, pooled across all instrument pairs.

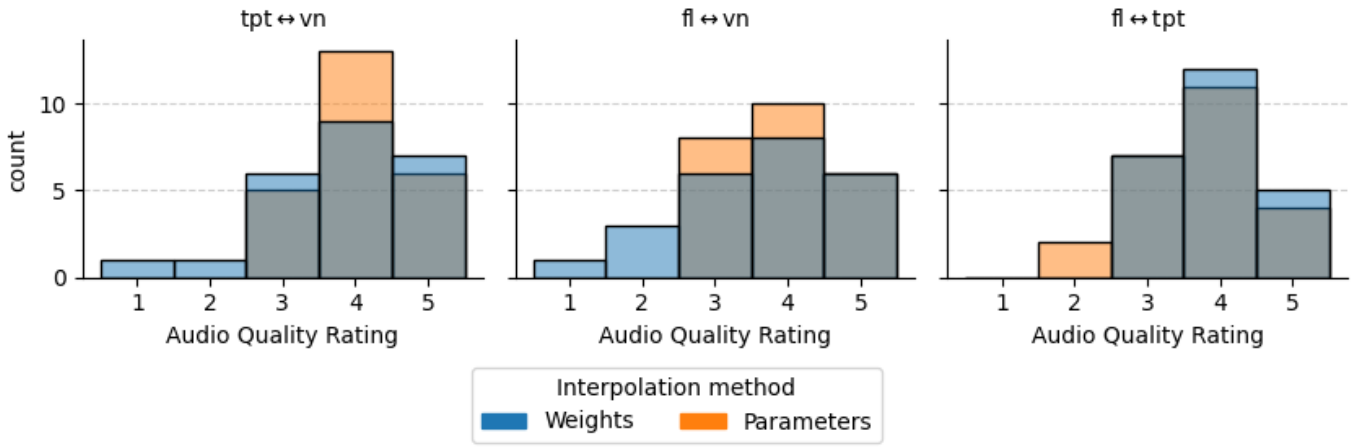


Fig. 13: Histogram of audio quality Ratings by interpolation method, for each pair of instruments, without distinguishing morph direction

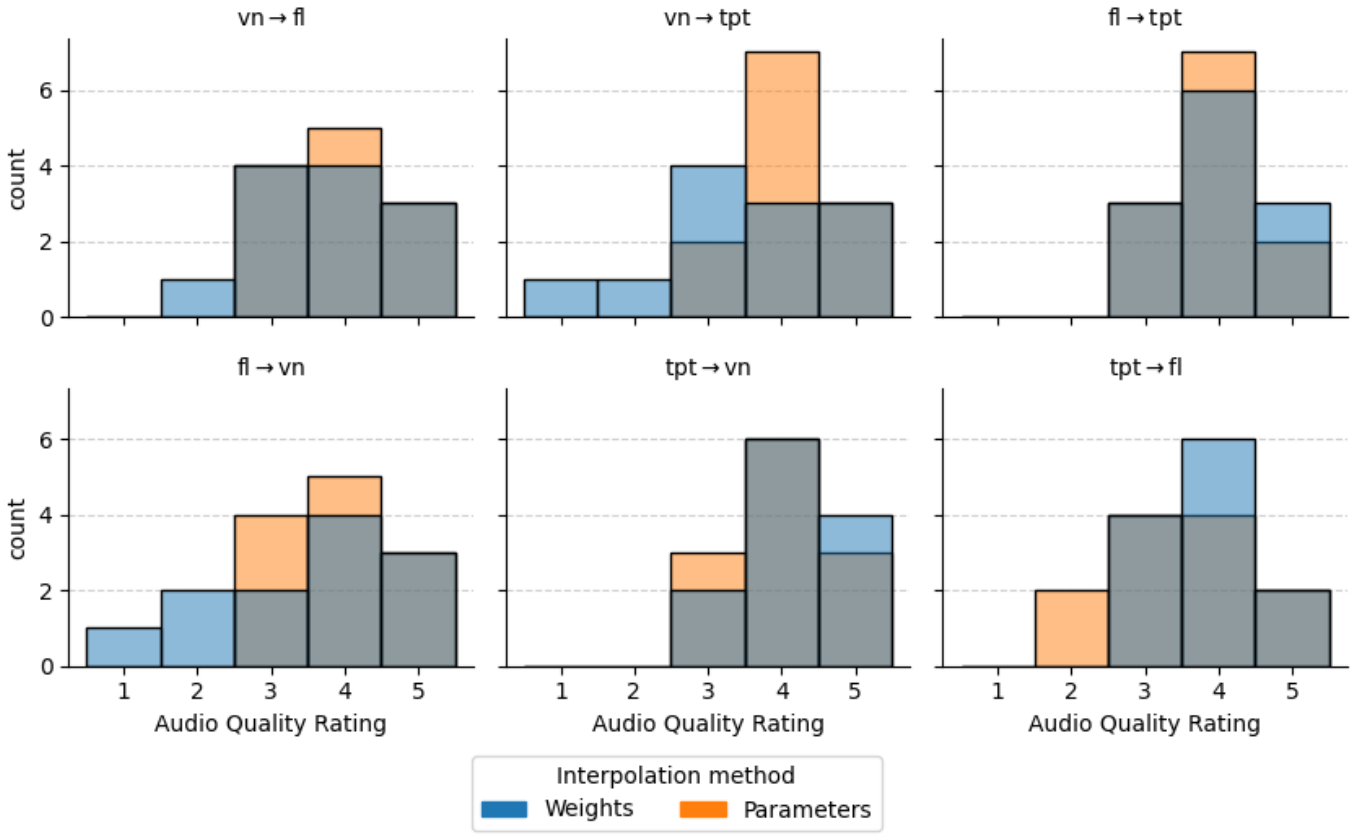


Fig. 14: Histogram of audio quality Ratings by interpolation method, for each pair of instruments, distinguishing morph direction

TABLE XIII: Mean audio quality rating by pair, considering morph direction and interpolation method.

Pair	Method	Mean	Std
fl→tpt	Parameters	3.92	0.67
	Weights	4.00	0.74
fl→vn	Parameters	3.92	0.79
	Weights	3.50	1.31
tpt→fl	Parameters	3.50	1.00
	Weights	3.83	0.72
tpt→vn	Parameters	4.00	0.74
	Weights	4.17	0.72
vn→fl	Parameters	3.92	0.79
	Weights	3.75	0.97
vn→tpt	Parameters	4.08	0.67
	Weights	3.50	1.24

TABLE XIV: Omnibus repeated-measures ANOVA of subjective rating by method and directed instrument pair.

Source	F	df_1	df_2	p	p_{GG}	η^2_G
Method	0.56	1	11	.472	.472	.003
Pair	1.93	5	55	.104	.136	.028
Method × Pair	2.50	5	55	.041	.070	.035

A. Timbre

Directed Pair	Method	Non-monotonic / Total (Filtered)	Proportion
fl→tpt	Parameters	3/8	37.5%
fl→tpt	Weights	6/12	50.0%
fl→vn	Parameters	4/12	33.3%
fl→vn	Weights	2/10	20.0%
tpt→fl	Parameters	5/7	71.4%
tpt→fl	Weights	7/9	77.8%
tpt→vn	Parameters	5/10	50.0%
tpt→vn	Weights	3/11	27.3%
vn→fl	Parameters	3/10	30.0%
vn→fl	Weights	3/11	27.3%
vn→tpt	Parameters	6/12	50.0%
vn→tpt	Weights	2/7	28.6%
Overall	**Parameters**	**26/59**	**44.1%**
Overall	**Weights**	**23/60**	**38.3%**

TABLE XV: Non-monotonic trials as a proportion of trials remaining after filtering, by directed instrument pair and method.

Directed Pair	Method	Perceived Intermediate / Total (Filtered)	Proportion
fl→tpt	Parameters	8/8	100.0%
	Weights	12/12	100.0%
fl→vn	Parameters	11/12	91.7%
	Weights	9/10	90.0%
tpt→fl	Parameters	4/7	57.1%
	Weights	8/9	88.9%
tpt→vn	Parameters	10/10	100.0%
	Weights	10/11	90.9%
vn→fl	Parameters	10/10	100.0%
	Weights	9/11	81.8%
vn→tpt	Parameters	8/12	66.7%
	Weights	6/7	85.7%
Overall	**Parameters**	**51/59**	**86.4%**
	Weights	**54/60**	**90.0%**

TABLE XVI: Trials where intermediate alphas were perceived as falling between the extreme alphas, as a proportion of trials remaining after filtering, by directed instrument pair and interpolation method.

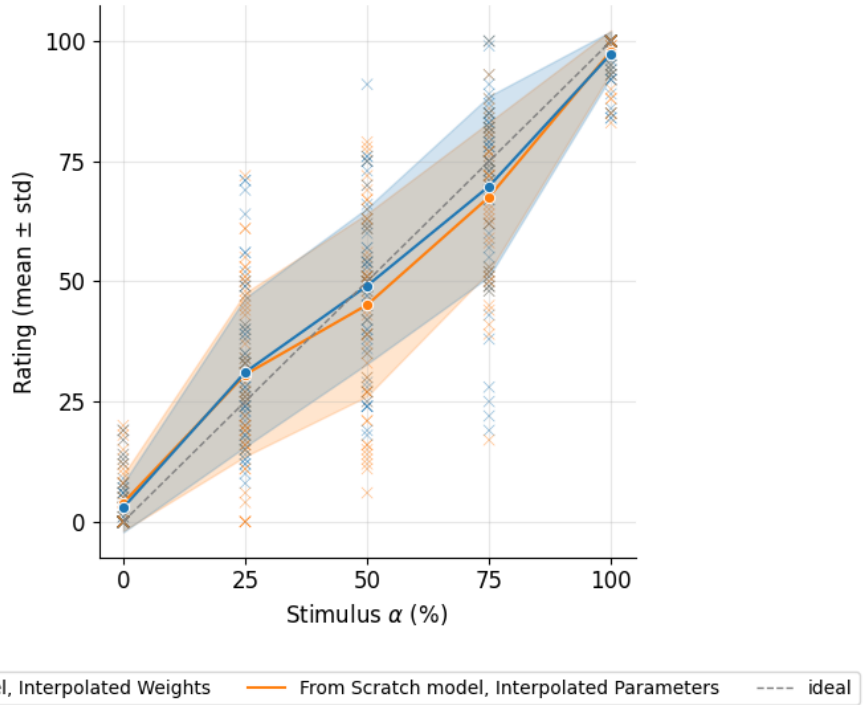


Fig. 15: Mean and standard deviation for listening test timbre ratings. Instrument A → instrument B morphs, for stimuli created for $\alpha = [0, 0.25, 0.5, 0.75, 1]$. Pooled across instrument pairs, by method.

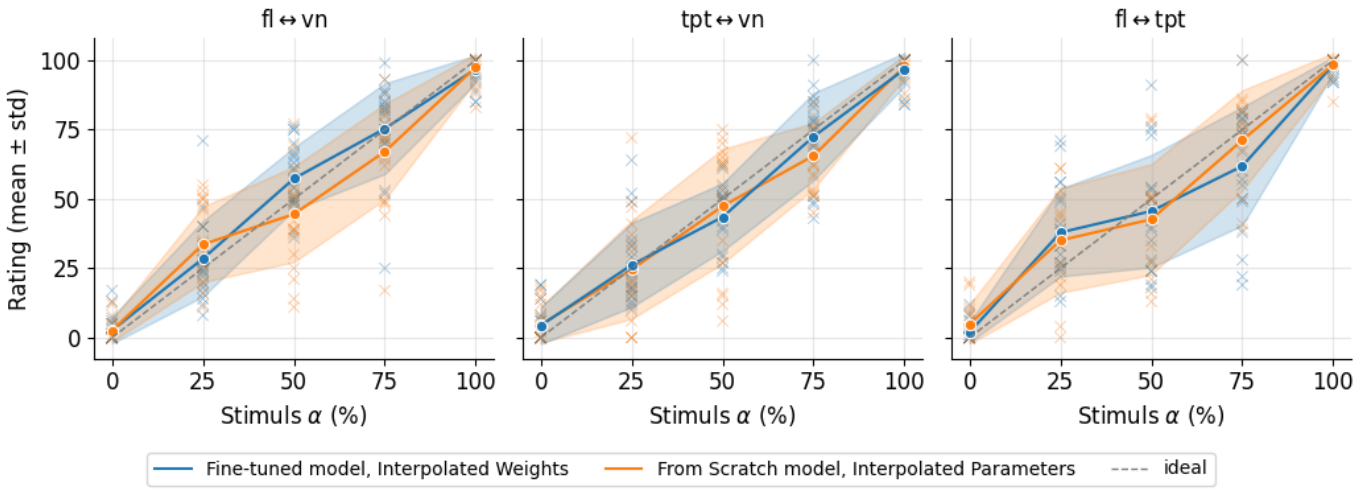


Fig. 16: Mean and standard deviation for listening test timbre ratings. Instrument A $\rightarrow$ instrument B morphs, for stimuli created for $\alpha = [0, 0.25, 0.5, 0.75, 1]$. By instrument pairs regardless of morph direction, by method.

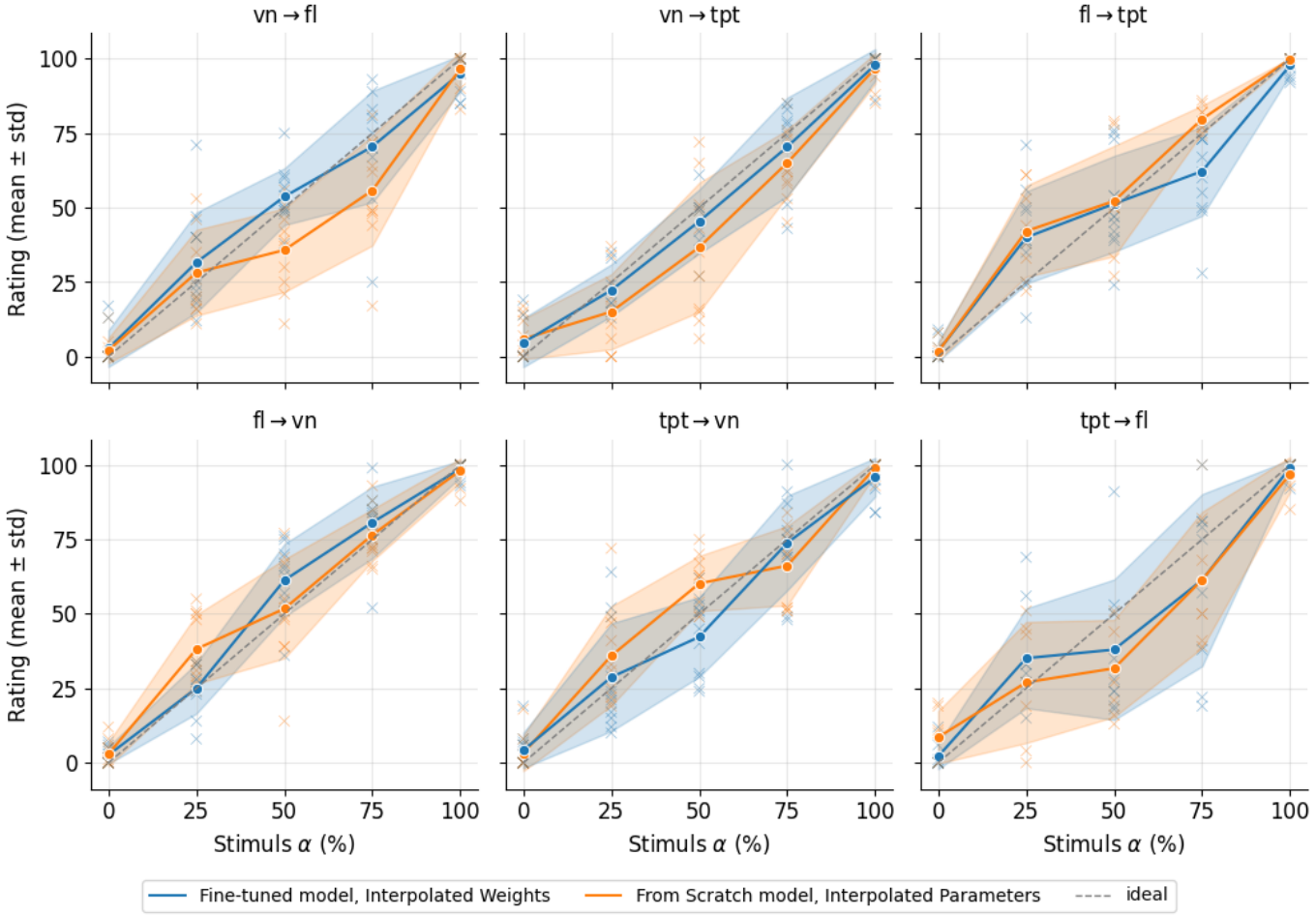


Fig. 17: Mean and standard deviation for listening test timbre ratings. Instrument A $\rightarrow$ instrument B morphs, for stimuli created for $\alpha = [0, 0.25, 0.5, 0.75, 1]$. By instrument pairs considering morph direction, by method.

ECS750P/ECS753P/ECS754P– MSc Project Module
Generative AI Tools – Student Accountability Statement

The completed and signed statement must be included in the appendix of the final dissertation paper.

Student Name: Jimena Arruti
Project Title: Sound Morphing between instruments with DDSP
Supervisor Name: Anna Xambó Sedó

1. Have you used Generative AI tools for your project?
YES X
NO
If yes, please complete sections 2 and 3 and sign the declaration below.
If no, please sign the declaration below.

2. Please provide details of all the AI tools that you used in your project work.

AI Tool	Version	Publisher	Access URL
Claude	Sonnet	Anthropic	https://claude.ai/
GitHub Copilot	-	Microsoft	https://github.com/features/copilot

3. If you have used Artificial Intelligence (AI) tools in any part of your project please include in table format how, where, and why these tools were used. This should include references to the relevant parts of your project work — for example:

- Specific chapters or sections of the report (e.g. literature review, methodology, requirements)I
- Software development (e.g. AI-assisted code generation, debugging, documentation)
- Interview or survey materials (e.g. AI-generated or refined questions/prompts)
- Datasets (e.g. AI-assisted data cleaning, labelling, or synthesis)
- The showcase video (e.g. AI-assisted scripting, narration, or editing support)

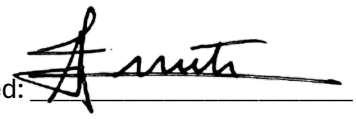
Be specific and transparent about your use of AI tools, including both direct outputs (e.g. text, code, images) and indirect uses (e.g. idea generation, editing assistance). The goal is to ensure academic integrity and provide a clear record of how AI contributed to your work.

Project Section	Description	Justification
Literature Review	Asked for paper recommendations	I used GenAI to complement the results of Google Scholar searches, but thoroughly read and analysed the relevance of the papers myself

Overall	Brainstorming, concepts explanation, and getting feedback	At the beginning I used it for a session of brainstorming to discuss different ideas on possible directions for the project, then explored the options myself. I also asked it to explain concepts in different ways so I could understand them better. I sometimes explained my reasoning or line of thought to it and asked for feedback, as a way to get myself to consider other viewpoints I might have missed.
Diagrams	Approximating an existing diagram in Figma	I asked it to translate the main DDSP and DDSP decoders from Engel et al. (2020) to Figma code, so I could use them as a base for my diagrams.
Code	Checks for bugs	I inserted parts of the code and the error I was getting and asked for an explanation of what might be wrong to speed up debugging
Code	Assistance with refactoring (e.g. help detect repeated code or recommendations for optimization)	I had it analyse the codebase and give suggestions on code I should optimise, or opportunities for converting repeated code into functions
Code	Modifications to webMUSHRA page	The listening test code was developed iteratively with Gen AI's help, since I did not have enough time to focus on programming on JavaScript, and it was not a core part of my research
Code	Scripts creation for rearranging results in a given folder structure	I asked it to create two scripts: reorder_dataset.py and reorder_results.py, to restructure files in a given folder structure
Code	Help understand reference Pytorch code	At the beginning, I used it to help me understand how to use different Pytorch functions, and help me understand the code I used as a starting point
Code	Improving figures	I asked it how to modify the figures in certain ways to make them more visually appealing
Dissertation document and reflective essay	Review and feedback	I used it to get feedback on typos, grammar and general writing quality of the documents
Dissertation document	Help with Latex table creation from csv files	I used it to help me rapidly create the Latex tables in the dissertation document

3. Generative AI Student Declaration

- I have declared all uses of AI tools (e.g., coding assistants, text generators, data analysis tools) in this project.
- I have reviewed and verified all AI-assisted content to ensure accuracy and suitability.
- I take full responsibility for the originality and integrity of my project submission.
- I have referenced all sources, including those suggested or produced with AI support.
- I have not used AI in ways that undermine academic honesty (e.g., plagiarism, unverified code submission).
- My use of AI was supplementary and did not replace my own independent learning and problem-solving.

Signed:  Date: 22/08/2026

Guidance notes for students

- Students are strongly encouraged to review the following resources
    - Student Guide to Gen AI https://www.qmul.ac.uk/library/academic-skills/student-guide-to-generative-ai/
    - Academic Integrity and Misconduct Policy https://www.qmul.ac.uk/governance-and-legal-services/media/dgls-media/policy/current-policies/Academic-Integrity-&-Misconduct-Policy.pdf

- The use of essay mills, homework help sites, plagiarism, collusion, falsification, impersonation, or actions that may give an unfair advantage is strictly prohibited in your final year project. Please refer to the QMUL Academic Integrity and Misconduct policy above.
- It is never acceptable to use GenAI tools to write your entire literature review and submit it as your own work; doing so is considered a form of plagiarism since GenAI outputs are derived from ideas of human authors without proper referencing.
- You may use GenAI tools for checking grammar and spelling, provided the tool is not rewriting a large portion of your work for your final year project.
- Using GenAI for idea generation or initial planning is generally appropriate but always consider the context and assessment requirements for your project. If in doubt, please talk to your supervisor.
- If you are ever unsure whether GenAI is permitted for any part of your project always ask your project supervisor for explicit clarification.

- Unless stated otherwise, GenAI should only be used to assist your learning at the initial stages of your project and your final submission of your project, and the code must still be substantially your own original work.
- Inappropriate use of GenAI will be viewed as plagiarism under the QMUL academic integrity and misconduct policy this includes submitting work where GenAI's contribution exceeds what is reasonably allowed in the final year project.
- Attempting to pass off GenAI-generated content as your own can result in disciplinary investigation under academic misconduct regulations; you may be asked to attend a viva to confirm authorship.
- Before using GenAI, make sure:
    - You are permitted (or required) to use it for your specific task in your final year project.
    - You fully understand the potential risks, limitations, and how GenAI operates.
    - Your work remains genuinely your own and reflects your understanding.
- Please note that GenAI is helpful for gathering background information, but outputs may be biased, inaccurate, outdated, or fabricated.
- Always consult and cite original, credible sources; GenAI should never be treated as an original source and should not be cited as an author or in the reference list.
- Critically assess all GenAI output for accuracy, double-check any facts, quotations, or citations it suggests especially if you use them in your literature review or other chapters.
- If GenAI has been used for drafting, planning, or structuring, you must clearly acknowledge and document its use in your submission. Therefore, you should keep drafts and records of your own process to show your independent scholarly work and reading notes
- GenAI use may not require acknowledgement for:
    - Familiarising yourself with a topic or research landscape.
    - Summarising information to prepare for more in-depth research.
    - Improving grammar or writing structure.
- Please note that output from GenAI is typically treated as having no author so you cannot cite Gen AI as an author in your report
- When in doubt, transparency and honesty about your methods and sources including GenAI are expected and protect your academic integrity