Multi-Aspect Conditioning for Diffusion-Based Music Synthesis: Enhancing Realism and Acoustic Control

Ben Maman* Johannes Zeitler* Meinard Müller* Amit H. Bermano†

*International Audio Laboratories Erlangen, Germany
† Tel Aviv University

***Abstract*—Music synthesis aims to generate audio from symbolic music representations, traditionally using techniques like concatenative synthesis and physical modeling. These methods offer good control but often lack expressiveness and realism in timbre. Recent advancements in diffusion-based models have enhanced the realism of synthesized audio, yet these models struggle with precise control over aspects like acoustics and timbre and are limited by the availability of high-quality annotated training data. In this paper, we introduce an advanced diffusion-based framework for music synthesis that further improves realism and introduces control through multi-aspect conditioning. This allows the synthesis from symbolic representations to accurately replicate specific performance and acoustic conditions. To address the need for precise multi-instrument target annotations, we propose using MIDI-aligned scores and automatic multi-instrument transcription based on neural networks. These methods effectively train our diffusion model with authentic audio, enhancing realism and capturing subtle nuances in performance and acoustics. As a second major contribution, we adopt conditioning techniques to gain control over multiple aspects, including score-related aspects like notes and instrumentation, as well as version-related aspects like performance and acoustics. This multi-aspect conditioning restores control over the music generation process, leading to greater fidelity in achieving the desired acoustic and stylistic outcomes. Finally, we validate our model's efficacy through systematic experiments, including qualitative listening tests and quantitative evaluation using Fréchet Audio Distance to assess version similarity, confirming the model's ability to generate realistic and expressive music, with acoustic control. Supporting evaluations and comparisons are detailed on our website (benadar293.github.io/multi-aspect-conditioning).**

***Index Terms*—Multi-Instrument Synthesis, Diffusion.**

I. Introduction

Music synthesis, the process of generating audio from symbolic music representations, is an important and long-studied area of research with applications in music creation and production. Traditional signal processing approaches, employing techniques such as concatenative synthesis [1, 2, 3, 4] and physical modeling [5, 6], have been used to produce high-quality audio for specific instruments. These methods provide explicit control over the audio output, thanks to note-wise rendering and the manipulation of pre-recorded samples. Even though these methods offer a variety of timbres, they still restrict flexibility and often lack expressiveness and realism, especially in terms of timbre and acoustic conditions. For example, replicating the unique sound of a specific orchestra or instrument poses significant challenges, such as producing audio that resembles the Berlin Philharmonic Orchestra or emulating the guitar sound from a 1975 recording by Segovia.

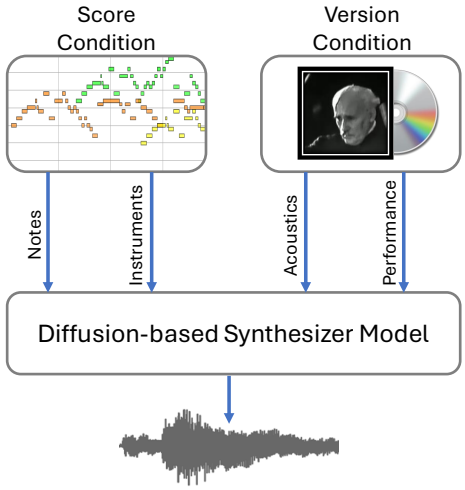

Fig. 1. Schematic overview of the proposed approach for diffusion-based music synthesis using multi-aspect conditioning. Our model generates audio conditioned both on the *score*, and on the *version*. The score contains the notes, instruments and timing, providing the musical content. The version corresponds to the specific acoustic and performance aspects, including timbre, rocording environment, and style. Version conditioning provides control and flexibility, enabling to generate different timbres of the same instrument (e.g., different types of guitar), or different room acoustics.

Recent advancements in data-driven generative modeling have addressed some of these limitations by enabling the inference of semantic aspects from example data [7, 8, 9, 10, 11]. In particular, Denoising Diffusion Probabilistic Models (DDPMs) have been extremely successful in rendering realistic images and learning styles from example images [11, 12, 13, 14]. Similarly, such models have been used to learn sounds from examples, significantly improving the expressiveness and realism of synthesized audio, including speech [15, 16] and music [17, 18].

However, these data-driven models require large datasets

This work was supported by Len Blavatnik and the Blavatnik family foundation, the ISF (grant 1337/22), and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grant No. 328416299 (MU 2686/10-2). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Fraunhofer Institute for Integrated Circuits IIS.

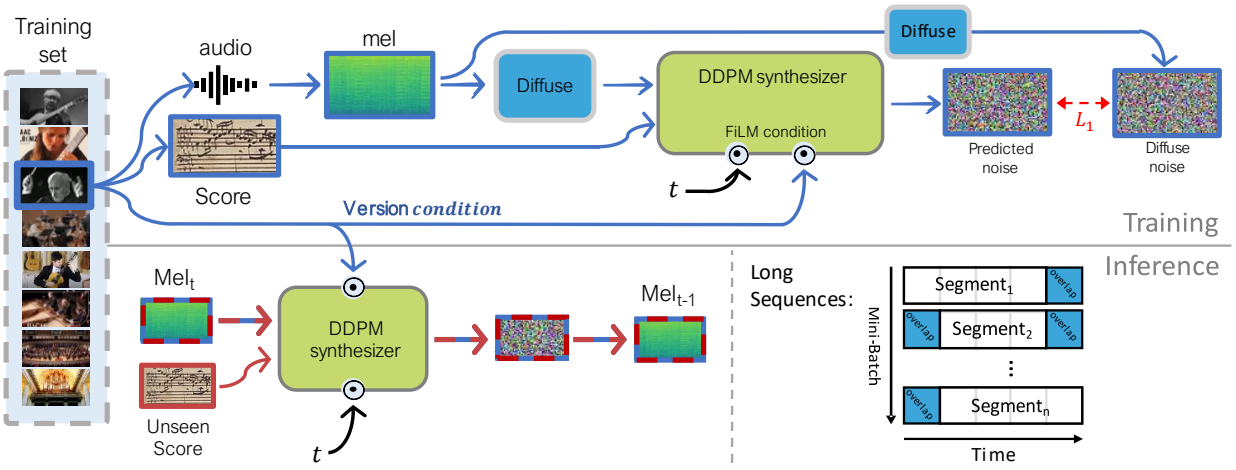

Fig. 2. Overview of our proposed diffusion-based synthesis model. Our model learns to denoise mel-spectrograms conditioned on score-based and version-based information. Version conditioning (determining the acoustics, specific timbre, recording environment and style) is done through FiLM layers at each block, which can be applied to either a T5 transformer or a U-Net. The version condition is represented by the performance ID, and is inserted at each layer by concatenation with the diffusion timestep. The score condition is provided as additional input, in the form of a MIDI-like piano-roll representation. For sampling consistent segments with smooth transitions, we use an overlapped generation technique borrowed from Computer Vision [19, 20].

with detailed annotations of notes and instruments, confining them to specific curated datasets or necessitating the use of lower-quality synthesized data. For example, the DDPM-based approach described by Hawthorne et al. [17] uses data based on concatenative synthesis, compromising audio quality and resulting in flat, less realistic audio.

Another problem is that these approaches often face challenges in precisely controlling aspects such as acoustics and timbre. In particular, these data-driven generative methods may encounter problems such as instrument drift, where the same instrument is not rendered coherently in different parts of the generated audio.

In this paper, building upon our previous work [21], we introduce an advanced diffusion-based framework for music synthesis that incorporates multi-aspect conditioning on notes, instruments, performance styles, and acoustic details to significantly enhance realism and control. Consequently, our method facilitates the synthesis of music signals from symbolic representations, accurately replicating specific performance and acoustic conditions as provided in example recordings. For instance, our model is capable of reproducing the guitar sound from a 1975 recording of Segovia playing Albéniz's "Capriccio Catalán", now applied to another piece, such as Jobim's "Felicidad." Furthermore, even in more complex musical settings, our model can transfer certain timbre and acoustic conditions from orchestral recordings, such as Karajan's 1962 recordings of Beethoven's symphonies with the Berlin Philharmonic, to new musical works presented in symbolic format. To the best of our knowledge, our work is the first to address such challenges in a multi-instrument setting.

The main contributions of this paper, which substantially extends our initial study [21], can be summarized as follows. First, to meet the demand for precisely annotated multi-instrument symbolic data, we propose two approaches: one using MIDI-aligned symbolic scores as in [21], and the other employing recent multi-instrument transcription methods [22]. We demonstrate that in both cases, the quality of the target data is sufficient to leverage uncurated multi-instrument real-world audio for training our diffusion model. This use of authentic training data enables our model to capture genuine musical performance characteristics, including subtle nuances in timbre and acoustics.

As a second main contribution, we expand the capabilities of diffusion models by integrating control over multiple musical aspects, including score-related aspects such as notes and instrumentation, as well as version-related aspects such as acoustics and performance. Specifically, we demonstrate how version control can be achieved using conditioning techniques based on Feature-wise Linear Modulation (FiLM) layers [23]. Our multi-aspect conditioning approach allows us to train a single model on massive amounts of uncurated multi-instrument performances with diverse instrumentation, including symphonic orchestras, chamber orchestras, church organs, harpsichords, violins, guitars, and more. Furthermore, we show how version conditioning restores control over the music generation process, leading to greater fidelity in achieving the desired acoustic and stylistic outcomes. For an overview of our DDPM-based model, see Figure 2.

As a third contribution, we validate the efficacy of our model through rigorous evaluation methods. These include listening tests and the application of the Fréchet Audio Distance [24] to assess both realism and version similarity, which confirms the model's ability to generate realistic and expressive audio. These evaluations are complemented by a wide range of synthesized music examples and comparisons with prior methods, all accessible on our freely available website.[1]

The remainder of this paper is organized as follows. Section II discusses related work, focusing primarily on music synthesis. Section III introduces the computational pipelines of our diffusion-based approach to music synthesis. In Section IV, we present our main technical contribution, the multi-aspect conditioning framework, which includes our extensions

[1] benadar293.github.io/multi-aspect-conditioning

TABLE I
OVERVIEW OF PREVIOUS WORK INDICATING ABILITY TO RENDER MULTIPLE INSTRUMENTS SIMULTANEOUSLY (**MULTI**), VERSION CONTROL (**VERSION**), GENERATING ORCHESTRAL SYMPHONIES (**SYMPH.**), **DATA** SIZE, AND RATIO OF REAL VS. SYNTHETIC DATA USED FOR TRAINING (**REAL%**).

	Multi	Version	Symph.	Data	Real%
[25]	✗	✓	✗	∼140H	100%
[26]	✗	✗	✗	∼1H	100%
[27]	✗	✗	✗	∼1H	0%
[28]	✗	✗	✗	∼1H	100%
[29]	✗	✗	✗	∼3H	100%
[17]	✓	✗	✗	∼1500H	∼ 2%
[30]	✗	✗	✗	∼93H	∼ 3%
Ours	✓	✓	✓	∼58H	100%

of score conditioning using alignment and transcription techniques, with our proposed version conditioning. In Section V we present our proposed technique for consistent and smooth segment-wise generation. Section VI details the evaluation criteria and presents experiments, including qualitative listening tests and quantitative evaluations. Finally, Section VII concludes the paper and discusses future work, highlighting the potential of version conditioning for automated instrumentation achievable with pitch-only input.

II. RELATED WORK

Audio synthesis in current literature can be done autoregressively, where models directly construct a waveform sample-by-sample [25, 31, 32]. Another approach, which we take, operates in the spectral domain. This requires a subsequent step to convert the generated spectral representation (whether the short-time Fourier transform or mel-spectrogram) into a waveform, but it is computationally more efficient.

For a data-driven approach, large amounts of labeled data are required, i.e., paired datasets of audio recordings and their corresponding time-aligned transcriptions. While such datasets exist for instruments such as the piano thanks to special equipment (e.g., Disklavier), this is not the case for other instruments. Thus, previous works mainly focus on generating piano performances, monophonic (single-voice) music (which is easier to label), or music produced by a concatenative synthesizer [17, 27], which is trivially supervised.

Table I provides a summary of existing methods for score-to-music synthesis. Wang and Yang [26] use a U-Net to synthesize solo violin, cello, and flute performances, requiring a separate model for each instrument. It is trained with a spectral reconstruction loss, without diffusion. Similarly, Dong et al. [28] use a Transformer architecture to synthesize solo violin or piano. Both works produce only monophonic and single-instrument music (i.e., only a single note or single instrument is synthesized at any given time).

Wu et al. [29] learn a parametric model of a musical performance, synthesizing from performance controls such as intensity, vibrato, etc. Although promising, a central drawback is that this work only operates on monophonic single-instrument data, similar to former works—due to the higher complexity of polyphonic music, and lack of high-quality polyphonic training data.

In multi-instrument synthesis, Hawthorne et al. [17] use a T5 Transformer-based diffusion model. While this method is promising and produces high-fidelity audio, it has limitations: It does not have control over the version, acoustics, and style (e.g., specific type of organ when several exist, or recording environment), and produces less realistic sound, due to lack of real annotated data, as demonstrated on our project page.

Kim et al. [30] use a diffusion-based approach following Hawthorne et al. [17], with a down-scaled T5 model, for guitar synthesis. The work is limited to guitar alone, and is trained mainly on synthetic data due to the aforementioned lack of real annotated data.

Other works exist for generating audio conditioned on text prompts [33, 34], however, in this work we focus on score-conditioned music synthesis.

III. DIFFUSION-BASED MUSIC SYNTHESIS

An overview of our method is depicted in Figure 2. We seek to enhance the generation quality and control of an off-the-shelf diffusion-based music generator using a collection of real multi-instrument performances with corresponding scores and version information. Hence, we start from a dataset $\mathcal{D} = \{(a_i, s_i, v_i)\}_{i=1}^{N}$, comprising audio performances a_i, their symbolic score annotation s_i, and information regarding the version v_i in the form of an identifier for the recording or performance. We represent a version condition v_i as an integer number, where recordings performed by the same ensemble in the same recording environment are assigned the same number (Section IV-C). With this dataset, we train a music synthesizer using different state-of-the-art architectures, namely T5 Transformer [17] or U-Net [11], infused with version conditioning.

Similar to previous works [17], we operate in the spectral domain, using a mel-spectogram representation. We postulate our method can be adapted to larger spectral representations (e.g., STFT), or the waveform domain, though at significantly higher computational costs. To convert mel-spectrograms into audio, we rely on the state-of-the-art Soundstream vocoder [35], which is the same vocoder used by Hawthorne et al. [17]. It was trained with a combination of spectral reconstruction and adversarial losses.

Finally, to generate long audio streams in a segment-wise fashion, while ensuring seamless transitions between generated segments, we adapt an overlapping technique, borrowed from visual generation (Section V). This technique is complemented by version conditioning, which also implicitly enforces consistency across different generated segments, thus preventing timbre drift.

A. Denoising Diffusion Probabilistic Models

We train our neural synthesizer as a Denoising Diffusion Probablistic Model (DDPM) [11]. DDPMs are trained to estimate the inverse of a Gaussian diffusion noising process, parameterized by a noise level schedule β_t, $t = 0, \dots T$, typically increasing from 0 to 1. By substituting noise level β_t

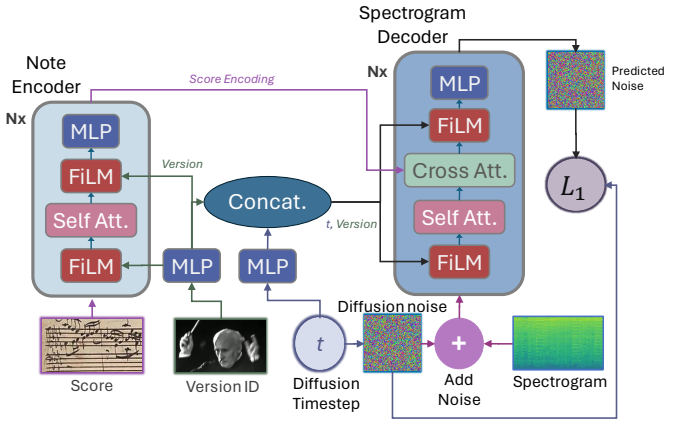

Fig. 3. Architecture of the T5 Transformer. The difference from Hawthorne et al. [17] is incorporation of version conditioning using FiLM layers. In the spectrogram decoder the version representation is concatenated to the diffusion timestep representation, and both are inserted into FiLM layers. In the score encoder, the version representation is inserted through FiLM layers. Note that the score encoding is independent of the diffusion timestep, similar to Hawthorne et al. [17].

with signal level, $\alpha_t = 1 - \beta_t$, the process can be represented as a Markovian chain, in the following recursive form:

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1 - \alpha_t}\epsilon \tag{1}$$

where $t \geq 1$, x_0 is a datapoint, i.e. a clean spectrogram, and $\epsilon \sim \mathcal{N}(0, I)$. By further substituting $\bar{\alpha}_t = \prod_{i \leq t} \alpha_i$, one can equivalently represent the process by the non-recursive form:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon. \tag{2}$$

A DDPM $\epsilon_\theta(x_t, t)$ estimates the reverse process by predicting the normal noise ϵ, given the noisy spectrogram x_t and the diffusion timestep t. In our case of multi-aspect-conditioned music synthesis, it can be trained by minimizing the empirical loss:

$$\mathbb{E}_{(x_0,s,v)\sim\mathcal{D},\, \epsilon\sim\mathcal{N}(0,I),\, t\sim U(\{1,\dots T\})} \|\epsilon_\theta(x_t, t, s, v) - \epsilon\|_1 \tag{3}$$

where s, v are the score and version conditions, respectively. In visual models the L^2-norm is common, however we follow Hawthorne et al. [17] and use the L^1-norm.

Note that $\bar{\alpha}_0 = 1, \bar{\alpha}_T = 0$, and $x_T \sim \mathcal{N}(0, I)$. Therefore, the model learns to sample from the latent data distribution p by mapping isotropic noise to data points:

$$p(x_0|x_T, s, v), \quad x_T \sim \mathcal{N}(0, I), \tag{4}$$

thus modelling the variation in the data. In our approach to multi-aspect-conditioned music synthesis, we use DDPMs to capture the variations in musical performances and to account for subtle nuances, since the same musical score can have infinitely many interpretations, even when played by the same musicians under identical acoustic conditions.

B. Architecture

We experiment with two architectures: A T5 Transformer used for score-conditioned spectrogram synthesis [17], and a U-Net originally used for images [11]. We enhance both models with version conditioning. The T5 architecture, which is our main focus, is depicted in Figure 3.

The **T5**, borrowed from Hawthorne et al. [17], comprises a transformer decoder, which is the generative backbone that denoises the spectrogram, and a transformer encoder, providing a representation of the score condition as auxiliary input to the decoder, for note and instrument control. The decoder receives the encoded score through cross-attention layers. A central motivation for separating denoising from score conditioning in the architecture is modularity—it theoretically enables separate training of each component, even on different datasets, which is common in text-to-image models [12, 13, 14]. Hawthorne et al. [17] train this model on a synthetic dataset, without version conditioning. We incorporate version conditioning into this model, and train it from scratch on real performances alone. Both the original T5 model of Hawthorne et al. [17] trained on synthetic data, and the T5 trained on our dataset, without version conditioning, serve as baselines for comparison, to evaluate both the effect of version conditioning, and the effect of using real training data (Section VI).

Although most of the evaluation in this work (including the listening tests) is done with the T5, we also experimented with a **U-Net**. We use the exact same architecture as Ho et al. [11], but adapt it to spectrogram synthesis by modifying all operations (convolution, attention, and group normalization) to be 1D rather than 2D, regarding the frequencies as channels. This allows for interactions between distant frequencies, inherent in spectrograms (partial frequencies), and is common practice in spectrogram synthesis (e.g., Wang and Yang [26] use a 1D U-Net without diffusion). Results for the U-Net can be found in the Appendix on our project page. As can be seen there, and from a qualitative observation, the T5 achieves slightly better results, but the U-Net requires significantly less training time.

IV. Multi-Aspect Conditioning

Our model is conditioned on multiple aspects: The noise level (as is common in diffusion models), the musical score information, and the version. In the following sections we explain how each condition is incorporated.

A. FiLM Layers and Diffusion Time Step Condition

A central mechanism we use for conditioning is Feature-wise Linear Modulation (FiLM) layers [23], which are especially suitable for multi-task cases, where a model learns many similar tasks simultaneously. For example, a diffusion model typically learns to denoise with many different noise levels—each noise level can be regarded as a slightly different task. FiLMs enable the different tasks to share most parameters while maintaining flexibility. The features at each level of the network are slightly modified with an affine transformation, according to the condition. Such transformations are also referred to as modulations (not to be confused with a musical modulation of key). The modulations are applied on the network features according to the formula:

$$\text{FiLM}(x_{i,j}|\gamma_{i,j}, \beta_{i,j}) = \gamma_{i,j} x_{i,j} + \beta_{i,j},$$

where $x_{i,j}$ is the j-th feature of the i-th layer of the network.

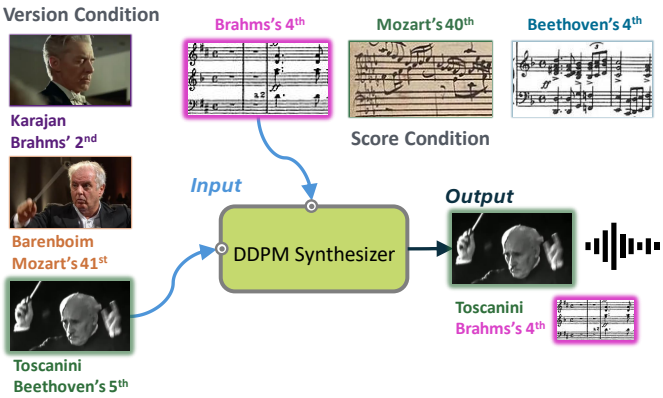

Fig. 4. Multi-aspect conditioning. We condition our model both on the score and on the version. The version is defined by a set of recordings by the same performer, in the same recording environment, e.g., a recording of Toscanini performing Beethoven's 5th symphony from 1952. Our model enables to synthesize new scores with the target version. In the latter example, the model can generate a performance of Brahms' 4th Symphony, with the acoustics and style of the 1952 Toscanini recording of Beethoven's 5th symphony.

FiLMs are typically used in diffusion models to handle different noise levels, i.e., diffusion timesteps [17]. In our case of synthesis using version conditioning, we further extend the modulations at each layer to control the desired acoustics depending on the version v (Section IV-C).

We apply FiLMs by predicting an affine transformation for each block of the network using multi-layer perceptrons (MLPs). More explicitly, given a condition c (representing a diffusion timestep t, or an ID number v corresponding to the version, or a combination of both), we learn an MLP embedding $C = M_\theta(c) \in \mathbb{R}^e$. For the i-th block of the network with input features $x_i \in \mathbb{R}^{n_i}$, we learn linear layers $L_i^\gamma, L_i^\beta : \mathbb{R}^e \to \mathbb{R}^{n_i}$ and modulate the input features to obtain new features x_i':

$$x_i' = (1 + \gamma_i)^\top x_i + \beta_i \tag{5}$$

where $\gamma_i = L_i^\gamma(C)$ and $\beta_i = L_i^\beta(C)$. See Section IV-C for details on combining the version condition with the diffusion timestep condition.

B. Score-Based Condition

A necessary requirement for any synthesizer is control over notes and instrumentation in generated performances. When following a data-driven approach, such control requires training data consisting of musical performances paired with their corresponding reference transcriptions. Any score-related aspect we seek to control should be faithfully represented in the corresponding reference transcription. Therefore, control over notes, exact timing, and instrumentation requires training data to have accurate transcriptions of all these aspects.

Music transcription in general, and especially of multi-instrument performances, is known to be a hard problem due to temporal-spectral overlaps and the lack of training data. Thus, previous DDPM-based works resort to artificial data generated by concatenative synthesis, for which transcriptions are trivially available [17, 30]. Unfortunately, this comes at the expense of realism, as the model learns to imitate the sound of a concatenative synthesizer, rather than real musical performances, and thus does not leverage the full generative power of DDPMs.

Recent work by Maman and Bermano [22] shows potential in transcribing general multi-instrument recordings. They propose a unified framework for automatic music transcription, and audio-score alignment based on neural features of transcription models. We show that this approach provides effective score conditioning, enabling to train a diffusion-based synthesizer on large amounts of uncurated real performances of diverse instrumentation, while maintaining control over notes and instrumentation.

1) Alignment vs. Transcription: The score conditions for the synthesizer should be exact score representations of the corresponding audio. These can be obtained in two ways:

- Alignment of an existing digital score representation with the audio
- Transcription predicted by an automatic transcriber

Both approaches have benefits and limitations in terms of robustness and accuracy: Alignment is generally easier than transcription and is constrained to a well-defined set of note events, including instrument information, eliminating confusion between pitches and instruments. Previous work shows that a weak transcriber can still produce an accurate alignment [22, 36]. On the other hand, automatic prediction can be more robust in case of alignment errors, which often occur [22, 36].

From a practical perspective, transcription is more favorable than alignment, as for most musical genres and performances (e.g. jazz or rock music) accurate scores are not easily available, or do not exist. A "universal" automatic transcriber (for which recent work [22] shows promising results) provides transcriptions for a wider range of musical performances, facilitating the use of massive training data for the synthesizer, similar to large text-to-image models [12, 13, 14].

In this work, we experiment with both approaches and show that they produce comparable results. Due to the major practical benefits of the transcription approach, we perform qualitative listening tests (Sections VI-C1-VI-C2) on a synthesizer trained with transcriptions as conditions. In addition, we quantitatively evaluate and compare both score conditioning approaches, and show comparable performance (see Appendix and samples on project page).

2) Score Errors: We note that both approaches may lead to errors, or deviations between the musical performance and the assigned reference notes. However, a small amount of errors will not necessarily negatively impact the model, as the model is trained to generate performances from the distribution of real performances, that do not contain errors, from possibly noisy scores, thus learning to correct errors. Still, investigating the effect of transcription or alignment accuracy on the model's performance is an important direction for future work.

C. Version-Based Condition

In musical performances, many aspects other than the musical score itself bear significant influence on the outcome, including both acoustic- and performance-related aspects.

Acoustic-related aspects include specific instrument timbre, e.g., the specific kind of guitar used, the singer identity, the church organ register, or more general acoustics such as the room acoustics, the orchestra size, position of the audience or recording device relative to the orchestra, and many more. Performance- or style-related aspects include, for example, dynamics and expression—the same musical segment can be played in infinitely many styles with varying intensity or fluctuation, e.g., using vibrato, tremolo, or staccato, legato, etc. All of these aspects are reflected in the specific *version* of the musical piece.

A successful synthesizer should take into account the latter aspects associated with the specific version, in addition to the score. Therefore, we condition our generated performances on the target version, in order to obtain its special characteristics. This *version conditioning* enables acoustic and style control, and reduces the ill-posedness of score-conditioned music synthesis. The different aforementioned version-related factors such as timbre, acoustics, and style are thus represented implicitly by the *version condition*. Using this mechanism, we can generate different types of the same instrument (e.g., guitars of different timbres), or different orchestral sounds. We demonstrate this on our project page, by synthesizing the same scores with different orchestras, harpsichords, church organs, and guitars.

We represent a version condition as an integer number v_i, where recordings performed by the same ensemble in the same recording environment are assigned the same number. Each condition can represent a single recording of a few minutes in the train set (e.g., Segovia playing Albéniz's Capriccio Catalán on the guitar), or a set of recordings of several hours (e.g., of Beethoven's concertos for piano and orchestra performed by Mitsuko Uchida and The Royal Concertgebouw Orchestra).

1) Version Conditioning with FiLM Layers: We incorporate the version condition using FiLM layers, similar to diffusion timesteps. As mentioned in Section IV-A, FiLM layers are suitable for multi-task cases. In the case of a synthesizer trained on many unrelated performances of diverse instrumentation, each performance with its corresponding acoustics and style should be regarded as a slightly different task.

As mentioned in Section III-B, the T5 architecture we use [17] consists of a spectrogram decoder, denoising the spectrogram, and a score encoder, providing a representation of the notes as auxiliary input to the decoder through cross-attention layers. Note that Hawthorne et al. [17] condition only on the diffusion timestep and not on the version. They use FiLMs in the decoder to incorporate the diffusion timestep into denoising, but not to the score encoder; i.e., the score representation is independent of the noise level.

We incorporate the version condition into the decoder and encoder using FiLMs in the following manner: We learn two MLP representations for the version v, and diffusion timestep t, which we denote by $M_\theta(v)$, $N_\theta(t) \in \mathbb{R}^e$. For the decoder FiLMs, we concatenate them, and use the combined condition:

$$C = C(v,t) = \text{concat}(M_\theta(v), N_\theta(t)) \in \mathbb{R}^{2e} \tag{6}$$

in Equation 5 (see also Figure 3). For the score encoder FiLMs we use the version condition alone, without the diffusion timestep, as explained above, leading to the condition:

$$C = C(v) = M_\theta(v) \in \mathbb{R}^e. \tag{7}$$

For the U-Net, we use FiLMs to condition each block on the combination of the diffusion timestep and the version, as in Equation 6.

V. Temporal Coherency & Smooth Transitions

We generate long performances of several minutes by segments of ∼5 seconds each, dictated by memory constraints. This raises an issue of coherency and consistency between segments. A naïve approach of block-wise synthesis will allow abrupt changes in volume, timbre, expression, ambience, etc. in transition points, due to different trajectories in the diffusion reverse process. Moreover, even with smooth transitions, timbre drift can occur, i.e., changes in timbre between segments. We propose a simple and effective overlapped generation technique for smooth transition between segments, adapted from computer vision.

A. Smooth Transitions

For smooth transition between segments, we generate segments with short overlaps, smoothly interpolating between consecutive segments in each step of the sampling process. We use an interpolation coefficient linearly decreasing from 1 to 0 along the overlap, in the predicted sample $\hat{x}_0$ of each step, derived from the predicted noise $\hat{\epsilon}$ from the formula $\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}}{\sqrt{\bar{\alpha}_t}}$. Borrowed from motion generation [19, 20], this is an effective and convenient approach, performed solely at the sampling stage, and requiring no additional training components, contrary to Hawthorne et al. [17].

B. Acoustic Consistency

Smooth transitions enabled by the proposed interpolation do not guarantee consistent timbre, as timbre could still drift smoothly between segments, as is demonstrated on our project page. As can be observed in the samples, version conditioning is an effective way to address this issue, as it implicitly creates acoustic consistency and stability between segments.

VI. Evaluation

In this section, we discuss the evaluation of our proposed DDPM-based synthesizer. In Section VI-A we define the desired properties of a high-quality synthesizer and explain the general criteria and methods for evaluation. In Section VI-B we present the datasets used for evaluation. In Section VI-C we present the qualitative listening tests we have performed for evaluating the model. In sections VI-D and VI-E we present quantitative results using established score metrics based on the Fréchet Audio Distance (FAD) [24], and transcription metrics.

A. Evaluation Criteria

A high-quality synthesizer should possess both high generative power on the one hand, and elaborate control on the other hand, while producing high-quality realistic sound. This can be broken down into the following required properties:

*a) **Realism & Quality**:* The synthesizer should generate realistic and high-quality sound. Ideally, the generated performances should be indistinguishable from real musical performances, i.e., pass the Turing Test [37].

*b) **Score Control**:* This involves the played notes or pitches, instruments, and timing, including note onset and offset. Notice that note onset and offset timing are also related to *performance control*, as discussed below.

*c) **Acoustic Control**:* This involves the specific instrument timbre (e.g., which type of violin), room acoustics, location of listener or microphone relative to the orchestra, orchestra size, and so on.

*d) **Performance Control**:* This encompasses aspects such as note intensity or strength (also referred to in the literature as the *velocity* in which the instrument was struck), degree of vibrato or tremolo, and more. Also, expressive interpretation of note timing can be regarded as a part of performance aspects.

*e) **Generating Unspecified Aspects**:* While control over the aforementioned aspects (score, acoustics, performance) is highly desirable, it is also required that the model can realistically generate aspects or confounding factors unspecified by the user, facilitating the generation process. This is required since many aspects of control, especially performance controls, such as vibrato and velocity, can be extremely laborious to define, and require highly skilled musicians.

In this work we focus on *realism and quality, score control*, and *acoustic control*. We rely on the model to *implicitly generate performance aspects* such as vibrato and intensity. Examples can be found on our project page demonstrating the ability of the model to generate such aspects, e.g., a violin playing with vibrato. We argue that explicit performance control in a data-driven approach requires the transcriptions to contain this information, and leave this for future work.

To evaluate *realism and quality*, in Section VI-C1 we conduct a MUSHRA listening test [38] comparing performances of the same musical excerpts generated on various synthesizers, both concatenative and diffusion-based, with and without version conditioning, as well as real versions, both original and vocoded.

Furthermore, to evaluate *acoustic control*, in Section VI-C2 we conduct an additional listening test to measure the effect of version conditioning in achieving the target acoustics, to which we refer to as *version similarity*.

In Section VI-D we complement the qualitative listening tests with quantitative perceptual metrics based on the Fréchet Audio Distance (FAD). We introduce two ways of measuring the FAD: *All-FAD* to measure realism and quality, and *Group-FAD* to measure version similarity. Finally, in Section VI-E, to evaluate *score control*, we apply quantitative transcription metrics to evaluate the faithfulness of our generated performances to the target notes and instruments.

B. Datasets

a) Train Set: We train our model on 197 performances of Western classical music, including symphonies, chamber music, and solo pieces, comprising 19 instruments, and totaling 58:06:07 hours. The data consists of performances from YouTube [39] and Musopen [40], with corresponding MIDI transcriptions from KunstDerFugue [41], aligned as proposed by Maman and Bermano [22]. Following the same work, we augment the data by pitch-shifting up to ± 2 semitones (larger ranges of shifts did not improve performance). We label the data with version condition IDs by assigning numerical indices to the different performances, where typically the same index is given to an entire set of recordings (e.g., a CD box with Beethoven's Piano Trios recorded by the same ensemble in the same studio). For church organ performances, where different organ registers are used in different audio tracks, resulting in significant timbre differences, we assign an individual version ID for each track. We denote this dataset $\mathcal{D}_{\text{train}}$.

b) Listening Tests Evaluation Set: For the listening tests, we use 12 MIDI performances of Western classical pieces, none of which appear in the train set, but containing the same instruments as in $\mathcal{D}_{\text{train}}$. The pieces include orchestral music, wind quintet, church organ, and harpsichord pieces. We use the first 3:00 minutes of each MIDI performance, which yields a total duration of 36:00 minutes. Short excerpts used for the listening tests were drawn randomly from these performances. As references for comparison, for each MIDI performance we use a corresponding real musical performance of the same score, of the same instrumentation, but of a version that does not appear in $\mathcal{D}_{\text{train}}$. We denote this dataset $\mathcal{D}_{\text{listen}}$.

c) Large Quantitative Evaluation Set: We quantitatively evaluate our models with 58 MIDI performances of Western classical pieces of a total duration of 5:09:30 hours, none of which appear in the train set, but containing the same instruments as in $\mathcal{D}_{\text{train}}$. For each test MIDI, we randomly sample three version conditions for synthesis. For example, the test MIDI can be a performance of Mozart's 40th symphony, and the version condition can be the performance of the Berlin Philharmonic Orchestra playing Brahms' Haydn Variations. This is the same evaluation set used by Maman et al. [21]. We denote this dataset $\mathcal{D}_{\text{quant}}$.

See our project page for the complete ensemble distribution of the datasets $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{listen}}$, $\mathcal{D}_{\text{quant}}$, and for the list of pieces used for the listening tests.

C. Evaluation Based on Listening Tests

We performed two distinct listening tests, to evaluate both realism (Section VI-C1) and similarity to the target version (Section VI-C2). Both listening tests are publicly available on our project's listening test page[2].

1) Realism Listening Test: In this listening test, we aim to evaluate to what degree our generated performances resemble real musical performances, as opposed to synthesized ones. Ideally, generated performances should be indistinguishable from authentic recordings. For this, we follow an adaptation of the MUSHRA (Multiple Hidden Stimuli with Hidden Reference and Anchor) protocol [38] to evaluate realism of generated performances.

[2]https://benadar293.github.io/listening-tests

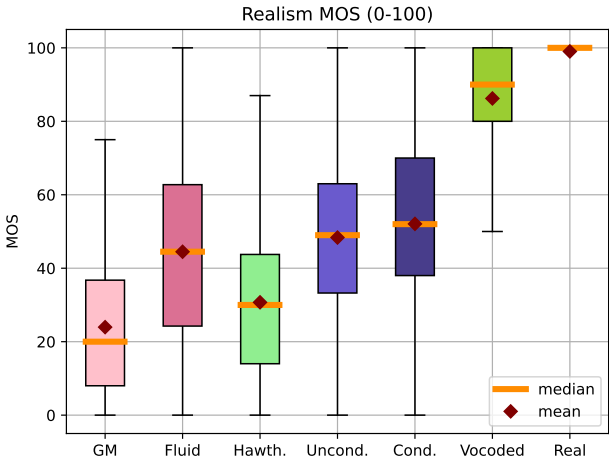

Fig. 5. MUSHRA realism listening test results in box plot form. For the exact Mean Opinion Scores (MOS) appearing in this figure, see Table II. 'GM' and 'Fluid' are soundfonts used for concatenative synthesis. 'Hawth.' is the model of Hawthorne et al. [17]. 'Uncond.' is our model without version conditiong (but with score conditioning). 'Cond.' is our model with both version conditioning and score conditioning. 'Vocoded' is the reference sample, after applying the vocoder to its mel-spectrogram, and is an upper bound on the achievable realism. It can be seen that both our models ('Cond.' and 'Uncond.') produce significantly more realistic performances than all compared models and methods.

TABLE II
REALISM MUSHRA LISTENING TEST MEAN OPINION SCORES (MOS). THESE ARE THE RESULTS APPEARING IN FIGURE 5 IN BOX PLOT FORM. WE LIST THE ABBREVIATIONS APPEARING IN FIGURE 5.

Model / Method	Abbreviation	Realism MOS ↑
Windows GM Soundfont	GM	24.0
Fluid R3 GM Soundfont	Fluid	44.5
Hawthorne et al. [17]	Hawth.	30.7
Ours w/o version condition	Uncond.	48.4
Ours w/ version condition	Cond.	52.1
Vocoded	Vocoded	86.2
Reference	Real	99.0

MUSHRA is a protocol for assessing audio quality, originally created for evaluating audio compression. Typically, the same excerpt is provided multiple times, restored with different codecs or algorithms. Each comparison of excerpts contains both a reference sample and one or more anchor samples. The reference represents the ideal quality, i.e., no compression is applied. The anchors represent baselines, against which different methods can be compared, and are used to calibrate the rating scale. The listener is provided with the reference sample and the set of samples for evaluation, also including a hidden version of the reference sample. The listener is asked to rate all samples on a scale from 0 to 100, and is required to rate the hidden reference with a score 90-100 (depending on the standard that is used). The MUSHRA protocol is considered robust, enabling statistically meaningful results with relatively few participants.

We apply the same methodology to evaluate the *realism* of our generated performances. Specifically, we want to find out if and to what extent the generated performance sounds like a real musical performance, as opposed to one performed on a synthesizer. The listeners are presented with an audio excerpt from a real musical performance as a reference, which

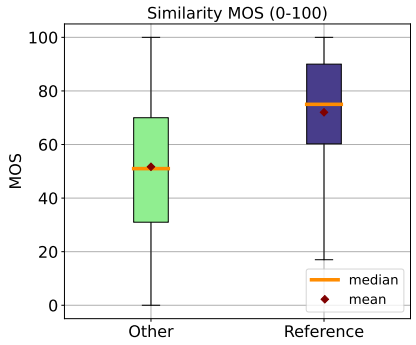

Fig. 6. Similarity listening test results. We synthesize the same score excerpt using our model, with three different version conditions, which are of performances with the same instrumentation. One of the three serves as a reference version. We ask the user to rate the similarity of each of the synthesized excerpts to a random excerpt of different score content, sampled from the reference version. Left ('Other'): Similarity to the reference version of excerpts synthesized with version conditions other than the reference version. Right ('Reference'): Similarity to the reference version of excerpts synthesized with the reference version condition.

represents ideal realism, and multiple synthesized versions of the same score content appearing in the excerpt, using different models or methods. As lower anchors we use synthesized versions obtained by concatenative synthesis, using two different soundfonts: The standard Windows Media Player GM soundfont, and the Fluid Release 3 General-MIDI soundfont, which is of higher quality. Concatenative synthesis serves as a baseline for achievable realism. We also include the vocoded version of the reference, in order to isolate the influence on realism of the vocoder from the spectrogram synthesizer. The vocoded version was obtained by calculating the mel-spectrogram of the reference sample and restoring the audio with the same vocoder we use for synthesis. Note that the vocoder quality is an *upper bound* on the quality achievable by our method. We compare excerpts of the same score content generated by the following neural synthesizers:

1) The original T5 model of Hawthorne et al. [17], which was trained mainly on data produced by a concatenative synthesizer, without version conditioning.
2) Our T5 model trained on our dataset, without version conditioning.
3) Our T5 model trained on our dataset, with version conditioning.

Note that the latter three configurations are all based on the same architecture. The differences are the type of data, real or synthetic, and whether or not version conditioning is used.

In total, seven versions were compared in each question: Two from concatenative synthesizers, three from neural synthesizers, the vocoded version, and the hidden reference. The seven versions were presented in random order. We asked the listeners to rate the realism of each excerpt from 0 (unrealistic) to 100 (realistic) and informed the listener that there is a hidden reference that must be ranked close to 100. The listener had to rate the reference with a score above 90 more than 85% of the time, otherwise the questionnaire was discarded.

The test comprised 32 participants, five of whom were discarded, leaving 27 participants. The questionnaire included ten questions. In each question, a twelve-second random

excerpt was drawn from a MIDI sampled from $\mathcal{D}_{\text{listen}}$, and sonified in the seven aforementioned variations (concatenative synthesizers, diffusion model, vocoder, etc.). The reference excerpt was obtained from a real performance of the same piece as the MIDI, through automatic alignment, which was manually verified to ensure it contained the same content. The test was conducted using the webMUSHRA implementation by Schoeffler et al. [42].

Results can be seen in Figure 5 and Table II. It is evident that our model trained on real performances produces more realistic results than all the baselines—both concatenative synthesizers (GM, Fluid), and the model of Hawthorne et al. [17], by a significant margin. This holds whether or not using version conditioning. For example, the basic concatenative synthesizer (GM) obtains 24.0 MOS in realism, where our model with version conditioning obtains 52.1 MOS. Our model also surpasses the more advanced concatenative synthesizer (Fluid), improving MOS from 44.5 to 52.1. Noteably, although we use the same architecture as Hawthorne et al. [17], the difference in realism is significant—over 20 MOS difference, from 30.7 to 52.1, due to the difference in training data. Also note that generating with or without version conditioning yields comparable realism scores, where version conditioning slightly improves MOS from 48.4 to 52.1. We note in this context that version conditioning mainly influences version similarity, discussed in Section VI-C2. Lastly, note that the vocoder quality has a non-negligible effect on the final result, as merely restoring the audio from the mel-spectrogram with the vocoder reduces the score from 99.0 to 86.2.

We conclude from these results that using real musical performances in a diffusion-based approach for music synthesis significantly improves realism compared to data based on traditional (concatenative) synthesis.

2) Version Similarity Listening Test: In this listening test, we are interested in evaluating if and to what extent our generated performances perceptually resemble the reference version used for conditioning, in terms of acoustics, timbre, performance, style, etc. For example, if generating an organ piece with a version condition of a specific organ, we want to evaluate to what extent the organ in the generated performance sounds like the organ given in the reference version.

In this listening test, for each comparison we randomly sample a reference audio excerpt a_{ref} with version v_{ref} and score s_{ref} from $\mathcal{D}_{\text{train}}$. We sonify a short score excerpt s (in MIDI form, of different content than the reference s_{ref}) three times, once with the same version condition v_{ref} corresponding to the reference, and with two additional randomly sampled version conditions v_1, v_2, of the same instrumentation as v_{ref}. We denote the outputs of the model by a^s_{ref}, a^s_1, a^s_2. For example, following Figure 4, our reference (a_{ref}, v_{ref}, s_{ref}) is Toscanini's performance of Beethoven's 5th Symphony. As a score condition s we use an excerpt from Brahms' 4th Symphony, which we sonify with the reference version condition v_{ref} being Toscanini's performance of Beethoven's 5th Symphony, and additional version conditions v_1, v_2 being Karajan's performance of Brahms' 2nd Symphony, and Barenboim's performance of Mozart's 41st Symphony. For the corresponding synthesized excerpts a^s_{ref}, a^s_1, a^s_2, we ask the user to rate the similarity of each to the reference excerpt a_{ref}, on a scale from 0 to 100.

TABLE III
RESULTS OF THE ALL-FAD METRIC FOR LISTENING TEST PERFORMANCES.

	All-FAD↓	
Method	VGGish	TRILL
$\mathcal{D}_{\text{listen}}$ Evaluation Set		
GM	10.07	0.47
Fluid	12.35	0.67
Hawth. [17]	7.7	0.33
Uncond.	4.03	0.18
Cond.	4.81	0.20
Vocoded	5.68	0.17
Real	2.85	0.15
$\mathcal{D}_{\text{quant}}$ Evaluation Set		
Uncond.	3.05	0.12
Cond.	3.58	0.11

The similarity test comprised 26 participants and included ten questions. For each question, we sampled a random twelve-second excerpt from a MIDI sampled from $\mathcal{D}_{\text{listen}}$, and sonified it with the three aforementioned version conditions, comparing them to the reference excerpt.

Results appear in Figure 6. In the left part, it can be seen that version similarity scores improve significantly by conditioning on the reference version, compared to conditioning on other versions, even though they are of the same instrumentation. The improvement is over 20 in the MOS, from 51.7 to 72.0 on average.

These results show that the notion of version similarity, reflected in perceptual and acoustic similarity, is indeed meaningful, and such similarity can be achieved by using version conditioning. The results clearly indicate that version conditioning is an effective means to obtain version-specific characteristics, e.g., specific timbre or acoustics that appear in a target version.

D. Quantitative Evaluation using Fréchet Audio Distance

We complement the listening tests with a quantitative evaluation using the *Fréchet Audio Distance (FAD)* [24]—a perceptual score with origins in computer vision [43]. We use this metric in different ways to quantitatively evaluate realism and quality (Section VI-D1), and version similarity, i.e., the resemblance of our generated performances to the conditioning version (Section VI-D2).

FAD relies on large models based on deep neural networks, such as TRILL [44], trained on large real-world datasets to predict embedding vectors from snippets of input audio. The assumption is that perceptually similar audio snippets yield closely spaced embedding vectors. To compute FAD between two audio datasets $\mathcal{D}_1$, $\mathcal{D}_2$ (e.g., a set $\mathcal{D}_1$ of synthesized audio conditioned on a specific version, and a set $\mathcal{D}_2$ of real recordings of the same version), the mean vectors μ_1, μ_2 and the covariance matrices Σ_1, Σ_2 are computed over all embedding vectors generated from $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. The FAD is then defined as:

$$\text{FAD}(\mathcal{D}_1, \mathcal{D}_2) = |\mu_1 - \mu_2|^2 + \text{tr}\left(\Sigma_1 + \Sigma_2 - 2(\Sigma_1\Sigma_2)^{1/2}\right). \tag{8}$$

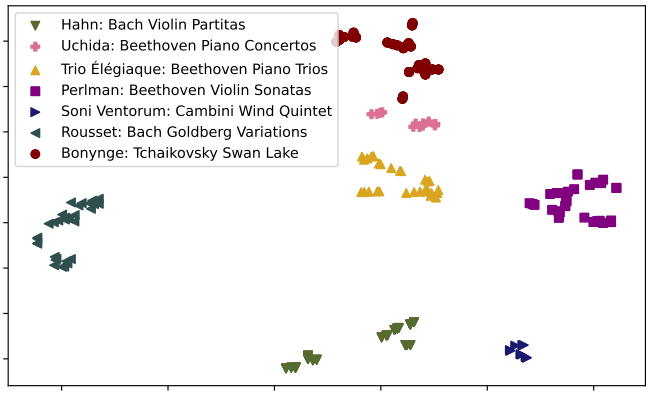

Fig. 7. T-SNE Visualization of the TRILL embedding space. Points represent audio tracks, and colors represent complete recordings of specific version IDs, comprising multiple such tracks.

Intuitively, if $\mathcal{D}_1$, $\mathcal{D}_2$ are perceptually similar, the distributions of the model's responses over the two datasets should be similar, resulting in a small distance. Kilgour et al. [24] show that FAD correlates with human perception and that increasing distortions increase the FAD. The results we present in the following sections further confirm this, as we show that qualitative listening test results are consistent with FAD scores.

We use two models as backbones for FAD, also used by Hawthorne et al. [17] for evaluation: TRILL [44] (5.9 embeddings/sec.), and VGGish [45] (1 embedding/sec.). We measure FAD in two ways, differing in the choice of the compared datasets: *All-FAD* for realism and quality (Section VI-D1), and *Group-FAD* for version similarity (Section VI-D2).

We report quantitative metrics for the generated performances used in the listening tests, and show they are consistent with the qualitative listening tests' results. For statistical stability, the metrics were measured on the entire $\mathcal{D}_{\text{listen}}$ evaluation set from which excerpts in the listening tests were sampled. In addition, for further statistical stability, we measure quantitative metrics on the large scale 5-hour evaluation set $\mathcal{D}_{\text{quant}}$ described in Section VI-B, also used by Maman et al. [21], and compare generation with and without version conditioning.

1) All-FAD—Realism & Quality: To assess realism, quality, and fidelity, we use FAD comparing the entire synthesized evaluation set to the entire train set. In the terms of Equation 8, we define $\mathcal{D}_1 = f_\theta(\mathcal{D}^{\text{midi}}_{\text{listen}})$, where $f_\theta(\mathcal{D}^{\text{midi}}_{\text{listen}})$ denotes our DDPM-based synthesizer f_θ applied to the set of MIDIs in $\mathcal{D}_{\text{listen}}$, and $\mathcal{D}_2 = \mathcal{D}^{\text{audio}}_{\text{train}}$, where $\mathcal{D}^{\text{audio}}_{\text{train}}$ denotes the audio recordings in the train set $\mathcal{D}_{\text{train}}$. Note that $\mathcal{D}^{\text{midi}}_{\text{listen}}$ can also be replaced by $\mathcal{D}_{\text{quant}}$. Also note that $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{listen}}$ contain both audio and scores, while $\mathcal{D}_{\text{quant}}$ contains only scores.

This captures the general similarity of the synthesized performances to real performances (or more specifically, to the entire set of real performances used for training), rather than resemblance to a specific version. We refer to this metric as *All-FAD*. While this metric is important for measuring realism and quality, note that the main quantitative metrics for evaluating version conditioning are the *Group-FAD* and *classification accuracy* metrics discussed later.

Results for the All-FAD metric appear in Table III (lower is better). It can be seen that using real training data significantly improves All-FAD w.r.t. all baselines, whether or not using version conditioning (Cond., Uncond.). For example, the All-FAD based on VGGish is over 10.0 for concatenative synthesizers and 7.7 for the model of Hawthorne et al. [17]. It improves to 4.03 (Uncond.) or 4.81 (Cond.) when using our model. This is consistent with the realism listening test results (Figure 5, Table II). Note that the reference data $\mathcal{D}^{\text{audio}}_{\text{listen}}$ (Real), i.e., the audio from $\mathcal{D}_{\text{listen}}$, achieves the best All-FAD scores (i.e., setting $\mathcal{D}_1 = \mathcal{D}^{\text{audio}}_{\text{listen}}$ and $\mathcal{D}_2 = \mathcal{D}^{\text{audio}}_{\text{train}}$ in Equation 8), despite being from versions that do not appear in the train set. This implies that the All-FAD metric indeed reflects realism and quality, beyond mere similarity to the train set.

Results on the large evaluation set $\mathcal{D}_{\text{quant}}$ show that version conditioning does not significantly affect All-FAD, implying that general quality is maintained. We also observe that version conditioning might slightly increase the All-FAD. This can be interpreted as follows: Version conditioning shifts the distribution of the generated performances from the general distribution of the train set $\mathcal{D}_{train}$, towards the distribution of specific versions, as explained in Section VI-D2.

2) Group-FAD: To measure version similarity, i.e., how well generated performances resemble the target conditioning version, we introduce the Group-FAD metric. To motivate its use, we show in Figure 7 a t-Distributed Stochastic Neighbor Embedding (t-SNE) visualization [46] from the train set's TRILL embedding distribution. Each point represents the

TABLE IV
RESULTS OF THE GROUP-FAD METRIC, FOR LISTENING TEST PERFORMANCES.

	Group-FAD↓	
Model	VGGish	TRILL
$\mathcal{D}_{\text{listen}}$ Evaluation Set		
GM	15.15	0.90
Fluid	15.35	1.10
Hawth. [17]	13.55	0.77
Uncond.	7.61	0.50
Cond.	**5.15**	**0.30**
Vocoded	8.69	0.45
Real	6.03	0.44
$\mathcal{D}_{\text{quant}}$ Evaluation Set		
Uncond.	7.46	0.55
Cond.	**5.68**	**0.36**

TABLE V
RESULTS OF VERSION CLASSIFICATION BASED ON GROUP-FAD, FOR LISTENING TEST PERFORMANCES.

	Classification Accuracy		
Model	Top-1	Top-3	Top-5
$\mathcal{D}_{\text{listen}}$ Evaluation Set			
GM	9.1%	9.1%	9.1%
Fluid	0.0%	0.0%	0.0%
Hawth. [17]	18.2%	18.2%	18.2%
Uncond.	18.2%	27.3%	45.5%
Cond.	**81.8%**	**90.9%**	**100.0%**
Vocoded	27.3%	36.4%	45.5%
Real	45.5%	45.5%	63.6%
$\mathcal{D}_{\text{quant}}$ Evaluation Set			
Uncond.	16.8%	31.6%	41.9%
Cond.	**66.5%**	**83.9%**	**88.4%**

mean embedding of an audio track (e.g., a movement in a symphony), and each color represents a recording, comprising multiple such tracks, corresponding to a specific version. It can be seen that tracks of the same version form clusters.

Following this insight, we define the *Group-FAD* metric: To measure how well our version-conditioned synthesized performances resemble the target version in acoustics, timbre, etc., we compute FAD comparing each score s synthesized with a version condition v, to the subset of the training set recordings corresponding to v, which we denote $\mathcal{D}_v^{\text{audio}}$. In the terms of Equation 8, we define $\mathcal{D}_1 = f_\theta(s, v)$, where f_θ denotes our DDPM-based synthesizer, and $\mathcal{D}_2 = \mathcal{D}_v^{\text{audio}}$.

Group-FAD results appear in Table IV. It can be seen that version conditioning dramatically and consistently improves Group-FAD, both compared to the baselines, and compared to our model without version conditioning. Note for example, that Group-FAD based on VGGish improves from 7.61 when using our model without version conditioning, to 5.15 when using our model with version conditioning. Notably, when using version conditioning, Group-FAD is better than that of the real samples $\mathcal{D}_{\text{listen}}^{\text{audio}}$ playing the same score but of another version (Real). For example, the TRILL-based Group-FAD for the real samples is 0.44, and improves to 0.3 for our version-conditioned samples (as explained in Section VI-C1, the reference samples $\mathcal{D}_{\text{listen}}^{\text{audio}}$ are from real performances of the test MIDIs, of versions that do not appear in the train set). Results are consistent between VGGish and TRILL, and are consistent with the version similarity listening test (Section VI-C2), indicating perceptual similarity to the target version obtained by version conditioning.

3) Version Classification: We further use the Group-FAD metric to classify the version of generated performances according to the Group-FAD nearest neighbor, over all training versions. In an initial experiment on version classification of *real recordings*, TRILL was more accurate than VGGish, therefore we use TRILL for version classification of generated performances.

Results are shown in Table V. Note, for example, that version conditioning (Cond.) improves version classification accuracy by 35-60%. The improvement is dramatic both compared to performances generated w/o version conditioning (Uncond.), and also compared to real performances (Real)—from 18.2% and 45.5% respectively to 81.8%.

E. Score Control Evaluation using Transcription Metrics

In this experiment, we quantitatively evaluate whether our DDPM-based model accurately renders the score as specified by the input score (given as MIDI representation). To this end, we use transcription metrics to determine if the generated performances contain the correct notes at the correct times, played by the correct instruments. We assess the transcription accuracy of the synthesized performances using an automatic transcriber [22, 25], trained on the same data as the synthesizer. We compare the note events in the input score to those in the transcription of the synthesized performance, and measure the F1 score for the following:

- Note: Accuracy of pitch and onset within 50ms.
- Note-with-Instrument: Accuracy of pitch, onset within 50ms, and correct instrument.
- Frame: Accuracy of note duration.

TABLE VI
TRANSCRIPTION RESULTS, FOR LISTENING TEST PERFORMANCES ('NOTE+IN.' IS THE NOTE-WITH-INSTRUMENT METRIC).

	Transcription F1% ↑		
	Note	Note+In.	Frame
$\mathcal{D}_{\text{listen}}$ Evaluation Set			
GM	60.0	27.0	59.2
Fluid	56.2	39.5	59.2
Hawth. [17]	54.3	16.4	58.8
Uncond.	**65.7**	45.8	59.3
Cond.	65.0	**48.4**	**61.5**
$\mathcal{D}_{\text{quant}}$ Evaluation Set			
Uncond.	**66.9**	**50.7**	**64.8**
Cond.	64.7	46.2	63.9

Results appear in Table VI. It can be seen that all methods produce transcription metrics that are on a rather comparable scale, except for the note-with-instrument metric. Our model (Cond., Uncond.) reaches a note-level accuracy of ∼65-67%, which is of reasonable magnitude when considering the complexity of highly polyphonic orchestral music. Note that the transcription metrics are influenced not only by the synthesizer's quality but also by the transcriber. The transcriber was trained on the same data as the synthesizer, and therefore might not perform as well on data generated by other synthesizers. Results on the large evaluation set show comparable transcription accuracy whether or not using version conditioning, similar to the All-FAD metric.

In the qualitative listening tests (Sections VI-C1, VI-C2), we focus on realism and version similarity, and not on score control. Although we provide quantitative results for score control, further listening tests for this aspect are an important direction for future work.

VII. CONCLUSIONS AND FUTURE WORK

We presented a framework for training neural diffusion-based synthesizers on real uncurated musical performances with diverse acoustics and instrumentation, including orchestral symphonies. Our proposed approach is based on diffusion models with multi-aspect conditioning, on both score and version. Through qualitative listening tests and quantitative evaluations, we have demonstrated that our approach produces performances with improved realism, and provides novel acoustic control, enabling to generate performances with version-specific characteristics, such as timbre and room acoustics. We strongly believe our approach is a significant step towards hyper-realistic and controlled music synthesis. Aside from extension to other genres, such as jazz, ethnic, and pop music, there are several important directions for future work, a few of which we outline:

1) Instrumentation Generation: Initial results provided on our project page demonstrate the ability to generate instrumentation from pitch-only input, implicitly controlling instrumentation through version-conditioning. Further investigation of this matter is an important direction for future work.

2) Unseen Versions: Our model can currently generate performances of unseen scores, with versions from the train set. Adapting a model to new unseen versions through inference from an example excerpt, or test-time adaptation, would be highly desirable.

3) Human Singing Voice & Lyrics Conditioning: While in this work we focus on instrumental music, we believe a unified diffusion-based framework for music and human speech is possible. In particular, we believe our approach could be applied to human singing voice synthesis, by applying similar additional conditioning techniques on lyrics or phonemes, and singer or speaker ID.

4) Elaborate Performance Control: In this work, we rely on the model to generate performance aspects such as note intensity and vibrato. Adding conditions for such performance aspects will enhance control, but will require enhancement of the transcriptions to comprise these aspects, which is an ongoing field of research.

5) Other Spectral Representations: In this work, we focus on mel-spectrogram synthesis and use a vocoder to convert the spectrogram to audio. We believe our approach can be applied to other spectral representations such as the magnitude-STFT or even the complex STFT, with higher computational costs.

REFERENCES

[1] D. Schwarz, "Current research in concatenative sound synthesis," in *Proceedings of the International Computer Music Conference (ICMC)*, Barcelona, Spain, september 2005.

[2] ——, "Concatenative sound synthesis: The early years," *Journal of New Music Reaserch*, vol. 35, no. 1, march 2006.

[3] B. L. Sturm, "Adaptive concatenative sound synthesis and its application to micromontage composition," *Computer Music Journal*, vol. 30, no. 4, pp. 46–66, 2006.

[4] E. Maestre, R. Ramírez, S. Kersten, and X. Serra, "Expressive concatenative synthesis by reusing samples from real performance recordings," *Computer Music Journal*, vol. 33, no. 4, pp. 23–42, 2009.

[5] K. Karplus and A. Strong, "Digital synthesis of plucked-string and drum timbres," *Computer Music Journal*, vol. 7, no. 2, pp. 43–55, 1983.

[6] J. O. Smith, "Physical modeling using digital waveguides," *Computer Music Journal*, vol. 16, 12/1992 1992.

[7] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial networks," *CoRR*, vol. abs/1406.2661, 2014.

[8] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013.

[9] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA*. Computer Vision Foundation / IEEE, 2019, pp. 4401–4410.

[10] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA*. Computer Vision Foundation / IEEE, 2020, pp. 8107–8116.

[11] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, 2020.

[12] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings of the International Conference on Machine Learning (ICML)*, Virtual, 2021, pp. 8748–8763.

[13] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition CVPR)*, New Orleans, USA, 2022, pp. 10 674–10 685.

[14] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, S. K. S. Ghasemipour, R. G. Lopes, B. K. Ayan, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi, "Photorealistic text-to-image diffusion models with deep language understanding," in *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA*, 2022.

[15] M. Jeong, H. Kim, S. J. Cheon, B. J. Choi, and N. S. Kim, "Diff-tts: A denoising diffusion model for text-to-speech," in *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia*. ISCA, 2021, pp. 3605–3609.

[16] V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, and M. A. Kudinov, "Grad-tts: A diffusion probabilistic model for text-to-speech," in *Proceedings of the International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 8599–8608.

[17] C. Hawthorne, I. Simon, A. Roberts, N. Zeghidour, J. Gardner, E. Manilow, and J. H. Engel, "Multi-instrument music synthesis with spectrogram diffusion," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2022, pp. 598–607.

[18] Q. Huang, D. S. Park, T. Wang, T. I. Denk, A. Ly, N. Chen, Z. Zhang, Z. Zhang, J. Yu, C. H. Frank, J. H. Engel, Q. V. Le, W. Chan, and W. Han, "Noise2music: Text-conditioned music generation with diffusion models," *arXiv*, vol. abs/2302.03917, 2023.

[19] J. Tseng, R. Castellon, and C. K. Liu, "Edge: Editable dance generation from music," 2022.

[20] Y. Shafir, G. Tevet, R. Kapon, and A. H. Bermano, "Human motion diffusion as a generative prior," *arXiv preprint arXiv:2303.01418*, 2023.

[21] B. Maman, J. Zeitler, M. Müller, and A. H. Bermano, "Performance conditioning for diffusion-based multi-instrument music synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Seoul, South Korea, 2024, pp.

5045–5049.

[22] B. Maman and A. H. Bermano, "Unaligned supervision for automatic music transcription in the wild," in *Proceedings of the International Conference on Machine Learning (ICML)*, Baltimore, Maryland, USA, 2022, pp. 14 918–14 934.

[23] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018, pp. 3942–3951.

[24] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, "Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms," in *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, Graz, Austria, 2019, pp. 2350–2354.

[25] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C. A. Huang, S. Dieleman, E. Elsen, J. H. Engel, and D. Eck, "Enabling factorized piano music modeling and generation with the MAESTRO dataset," in *Proceedings of the International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana, USA, 2019.

[26] B. Wang and Y. Yang, "Performancenet: Score-to-audio music generation with multi-band convolutional residual network," in *Proceedings of the Conference on Artificial Intelligence (AAAI)*, Honolulu, Hawaii, 2019, pp. 1174–1181.

[27] J. W. Kim, R. M. Bittner, A. Kumar, and J. P. Bello, "Neural music synthesis for flexible timbre control," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Brighton, UK, 2019, pp. 176–180.

[28] H. Dong, C. Zhou, T. Berg-Kirkpatrick, and J. J. McAuley, "Deep performer: Score-to-audio music performance synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Singapore, 2022, pp. 951–955.

[29] Y. Wu, E. Manilow, Y. Deng, R. Swavely, K. Kastner, T. Cooijmans, A. Courville, C.-Z. A. Huang, and J. Engel, "MIDI-DDSP: Detailed control of musical performance via hierarchical modeling," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[30] H. Kim, S. Choi, and J. Nam, "Expressive acoustic guitar sound synthesis with an instrument-specific input representation and diffusion outpainting," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 7620–7624.

[31] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," in *Proceedings of the ISCA Speech Synthesis Workshop*, Sunnyvale, USA, 2016.

[32] J. H. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proceedings of the International Conference on Machine Learning (ICML)*, vol. 70, Sydney, Australia, 2017, pp. 1068–1077.

[33] A. Agostinelli, T. I. Denk, Z. Borsos, J. H. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. H. Frank, "Musiclm: Generating music from text," *CoRR*, vol. abs/2301.11325, 2023.

[34] F. Schneider, Z. Jin, and B. Schölkopf, "Moûsai: Text-to-music generation with long-context latent diffusion," *ArXiv*, vol. abs/2301.11757, 2023.

[35] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 30, pp. 495–507, 2022.

[36] X. Riley, D. Edwards, and S. Dixon, "High resolution guitar transcription via domain adaptation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Seoul, South Korea, 2024, pp. 1051–1055.

[37] A. M. TURING, "I.—COMPUTING MACHINERY AND INTELLIGENCE," *Mind*, vol. LIX, no. 236, pp. 433–460, 10 1950.

[38] International Telecommunications Union, "ITU-R Rec. BS.1534-3: Method for the subjective assessment of intermediate quality levels of coding systems," 2015.

[39] "Youtube," https://www.youtube.com.

[40] "Musopen," https://musopen.org/.

[41] "Kunstderfugue," www.kunstderfuge.com.

[42] M. Schoeffler, S. Bartoschek, F.-R. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre, "webMUSHRA—a comprehensive framework for web-based listening tests," *Journal of Open Research Software*, vol. 6, no. 1, 2018.

[43] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA*, 2017, pp. 6626–6637.

[44] J. Shor, A. Jansen, R. Maor, O. Lang, O. Tuval, F. de Chaumont Quitry, M. Tagliasacchi, I. Shavitt, D. Emanuel, and Y. Haviv, "Towards learning a universal non-semantic representation of speech," in *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, Shanghai, China, 2020, pp. 140–144.

[45] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, New Orleans, Louisiana, USA, March 2017, pp. 131–135.

[46] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.