ProGress: Structured Music Generation via Graph Diffusion and Hierarchical Music Analysis

Stephen Ni-Hahn*
Duke University
stephen.hahn@duke.edu

Chao Péter Yang*
Duke University
peter.yang@duke.edu

Mingchen Ma
Duke University

Cynthia Rudin
Duke University

Simon Mak
Duke University

Yue Jiang
Duke University

Abstract

Artificial Intelligence (AI) for music generation is undergoing rapid developments, with recent symbolic models leveraging sophisticated deep learning and diffusion model algorithms. One drawback with existing models is that they lack structural cohesion, particularly on harmonic-melodic structure. Furthermore, such existing models are largely "black-box" in nature and are not musically interpretable. This paper addresses these limitations via a novel generative music framework that incorporates concepts of Schenkerian analysis (SchA) in concert with a diffusion modeling framework. This framework, which we call ProGress (Prolongation-enhanced DiGress), adapts state-of-the-art deep models for discrete diffusion (in particular, the DiGress model of Vignac et al., 2023) for interpretable and structured music generation. Concretely, our contributions include 1) novel adaptations of the DiGress model for music generation, 2) a novel SchA-inspired phrase fusion methodology, and 3) a framework allowing users to control various aspects of the generation process to create coherent musical compositions. Results from human experiments suggest superior performance to existing state-of-the-art methods.

Demo page with audio samples: https://huggingface.co/spaces/cpyang/progress-music-generation

1 Introduction

Music technology is expanding at a rapid pace with increasing focus on artificial intelligence (AI) [25]. Many generative audio AI companies and models have arisen recently, targeting domains such as video background music [1], responsive video game music [3], sleep and focus aids [2], and language-guided generation [6, 5, 4, 7, 13]. In particular, AI for symbolic music – music that can be written in a score – is a newly important topic in academic spheres [35, 43, 27, 15, 34, 26, 24].

One major concern with current music generation AI is a lack of music-theoretical awareness. Most existing models target the learning of music-theoretical principles in an implicit fashion by processing massive amounts of (often unethically sourced) data [28], primarily using massive models with hundreds of millions of parameters. Due to this reliance on training data without guidance from musical principles, such models fail to capture true musical structure, resulting in generated music that is incoherent, difficult to follow, and that sounds more like a "stream of consciousness." Several models have incorporated structure through musical form or meter, constraining music to a verse-chorus structure or encoding notes grouped by measure, e.g., [35, 43, 45, 6]. However, such

*Equal contribution

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Creative AI Track.

approaches do not account for the more detailed and complex voice-leading structure that is necessary for defining a musical style.

Looking to build more organically-structured music models that are guided by domain knowledge, recent promising work has incorporated features of *Schenkerian analysis* (SchA) and music-theoretical concepts within learning model algorithms and architectures [15, 29, 8, 9]. Along this vein, this paper introduces a novel generative symbolic music framework that incorporates aspects of hierarchical music theory in concert with deep learning. Our framework, which we call ProGress (Prolongation-enhanced DiGress), builds on state-of-the-art deep models for discrete graph diffusion [41, 20] with a careful integration of well-established music composition principles from SchA. In doing so, our framework allows users to control various aspects of the generation process in an interpretable manner to create novel, coherent, and musically pleasing compositions, even with highly limited training data. Concretely, our contributions include 1) novel adaptations of the DiGress model for music generation, 2) a novel SchA-inspired phrase fusion methodology, and 3) a framework allowing users to control various aspects of the generation process to create structurally coherent music. We emphasize that our model uses orders of magnitude fewer parameters than current state-of-the-art competitors, while producing *superior* generated music as evaluated by blinded human experiments.

The paper is structured as follows. Section 2 provides background information on SchA. Section 3 presents the proposed ProGress modeling framework. Section 4 discusses our experiments including a blinded human experiment, ablation studies, and genre transferability.

2 Background on Schenkerian Analysis

Schenkerian analysis (SchA) is a powerful tool for representing music's hierarchical harmonic-melodic structure, showing how harmonies are "unfolded" through time in the form of melodies [11, 37]. Vitally, SchA reveals recursive patterns in music at various levels of structure; the musical foreground (music as it is written in the score) hosts similar harmonic-melodic progressions to events in the musical middleground and background. While SchA was originally designed for western classical music of the common practice era (ca. 1600-1900), it has been adapted for analyzing music from all over the world, from Chinese opera to Ghanaian folk music [40], and over broad time periods and styles, from medieval polyphony [36] to modern rock [32].

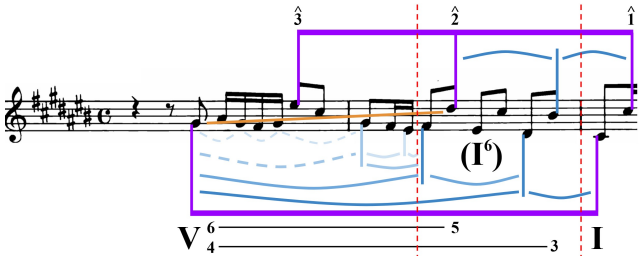

Figure 1: Example SchA of J.S. Bach's C♯ major fugue subject from *Das Wohltemperierte Klavier I*.

Figure 1 provides an example of the first author's analysis of Bach's C♯ major fugue subject from *Das Wohltemperierte Klavier I*. Here, we represent more foreground structures with lighter blue stems and slurs, while deeper middleground structures are represented with darker blue. The background structure is represented with purple. The background upper voice outlines a 3rd progression (E♯-D♯-C♯ or $\hat{3}$-$\hat{2}$-$\hat{1}$), which is a common cadential melodic pattern in tonal music. The background harmonic structure is described in Roman numerals at the bottom with red dotted lines to separate major harmonic shifts. The first measure outlines a cadential V, while measure 2 unfolds the resolution of the 6th and 4th to the tones of a dominant (G♯) harmony, which resolves to tonic I (C♯) in measure 3. The orange line connecting the bass G♯ in measure 1 to the treble D♯ in measure 2 clarifies that they are part of the same harmony in the background structure, separated by a relatively large span of time.

Note that the parenthetical I^6 in measure 2 is understood as a foreground passing harmony, *prolonging* the dominant V harmony that surrounds it. Prolongation (the inspiration for our model's name) refers to the phenomenon where certain notes or harmonies are "in control" at deeper levels of structure. While this example is relatively short, similar recursive prolongational relationships can span entire sections, movements, or even opuses.

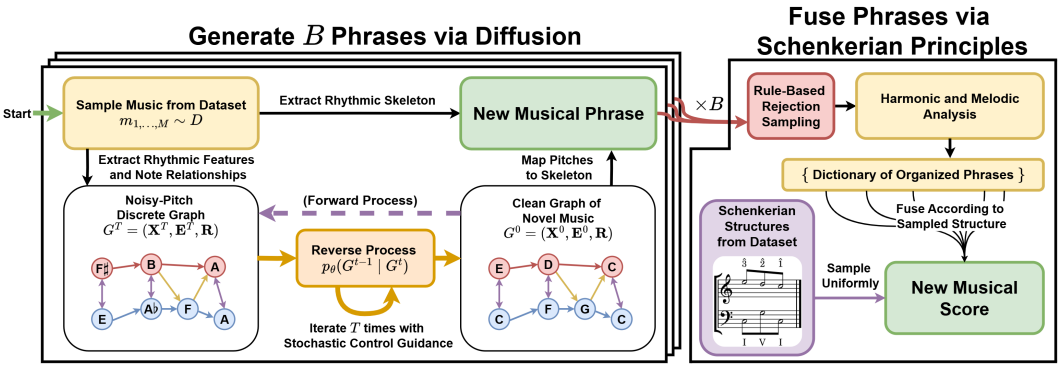

Figure 2: Overview of the phrase generation process. On the left half, B phrases are generated via diffusion, and on the right, phrases are fused together according to music theoretical principles and structures. In the generation stage, starting with the yellow block, we sample a phrase from our dataset D and extract rhythmic relationships to build a heterogeneous, discrete graph G^T. This discrete graph is iteratively passed through a denoising model p_θ to determine the notes for a novel piece of music. Finally, the inferred notes are mapped back to the rhythmic skeleton of the sampled phrase. This process is repeated B times to generate B phrases. For the fusion stage, phrases are first analyzed and organized based on harmonic and structural features. Based on user-defined rules, certain phrases are rejected. Phrases are then fused according to a sampled Schenkerian structure as described in Section 3.3.

By incorporating such harmonic-melodic structure in the generative process, music generative models can connect broader structures and produce more cohesive compositions, thus addressing a key limitation of many existing AI-based models. *ProGress*, presented next, aims to do this within a carefully-structured deep learning framework.

3 Methodology

We now describe the proposed ProGress music generation framework via discrete graph diffusion and prolongation-enhanced phrase fusion; Figure 2 visualizes its workflow. ProGress first extends a state-of-the-art diffusion model to generate a broad library of diverse musical phrases. After passing such phrases through rule-based rejection sampling, ProGress analyzes and organizes harmonic and melodic qualities of accepted phrases. Based on a sampled Schenkerian structure, individual phrases are then fused together into a structured score using music-theoretical principles. Section 3.1 describes how rhythmic information is extracted from a music dataset. Section 3.2 describes the musical representation, implementation details, and adaptations required for the employed diffusion model. Section 3.3 describes our phrase fusion methodology in finer detail.

3.1 Rhythmic Sampling

First, we sample from a musical dataset D or user input to determine a rhythmic framework as the backbone of our new music. There are numerous ways of performing this sampling to generate structured music. The simplest is to sample entire phrases from the dataset and extract their rhythm. Another is to uniformly sample various measures $m_1, \ldots, m_M$ from D and combine them into one phrase. If the latter approach is employed, it is useful to sample measures that end phrases separately, as cadential motions are often more carefully constructed. Phrases can further be combined and varied according to common patterns in the desired genre. For our figures and experiments, we focus on the case where rhythmic samples consist of entire phrases.

3.2 Discrete Graph Diffusion Modeling for Music

Next, we generalize the Discrete Graph Denoising Diffusion Model (DiGress) in [41] for musical phrase generation (see Appendix A for background and details on DiGress). The goal of such a model is to build a library of diverse musical phrases (i.e., set of pitches) given a rhythmic framework. For this model, graphs are defined by categorical node and edge attributes.

Node and Edge Categories: The nodes in our DiGress model each belong to a category from $\mathcal{X} = \{\hat{1}, \sharp\hat{1}, \flat\hat{2}, \hat{2}, \ldots, \sharp\hat{6}, \flat\hat{7}, \hat{7}, \text{rest}\}$, representing the *global scale degree* of a musical note, i.e, the note's place within the context of a piece's home key. The node category is our primary interest for inference, as it transforms the rhythmic framework into theoretical music with pitch classes. Edges each belong to a category from $\mathcal{E} = \{\text{forward}, \text{treble-voice}, \text{bass-voice}, \text{onset}, \text{sustain}, \text{structural}, \text{none}\}$, representing how notes relate to one another in the score. Structural edges connect notes that are connected with Schenkerian prolongations. Note that since there can only be one edge type between any two nodes, we must choose a precedence order for edge types that might coexist. For instance, if voice edges are overwritten, the music cannot be reconstructed. Most *Surface level* edges (edges that are inherent to the written music such as forward, onset, and sustain) are mutually exclusive, but voice edges are a subset of forward edges, and structural edges often coincide with forward/voice edges. Thus, surface level edges take precedence over any overlapping structural edges and voice and voice edges take precedence over forward edges.

Forward Process: A graph $G = (\mathbf{X}, \mathbf{E})$ is comprised of node embedding matrix $\mathbf{X} \in \{0,1\}^{n \times |\mathcal{X}|}$, where each row is a one-hot encoding $\mathbf{x}_i \in \{0,1\}^{|\mathcal{X}|}$ for graph nodes $i = 1, \ldots, n$, and edge embedding tensor $\mathbf{E} \in \{0,1\}^{n \times n \times |\mathcal{E}|}$, which describes each edge $\mathbf{e}_{i,j} \in \{0,1\}^{|\mathcal{E}|}$ from node i to node j as a one-hot encoding. Discrete graph diffusion applies noise independently to each node and edge, similar to pixels in image diffusion. At each forward diffusion step $1, \ldots, t, \ldots, T$, node and edge class transition probability matrices are defined as $\mathbf{Q}_X^t \in [0,1]^{|\mathcal{X}| \times |\mathcal{X}|}$ and $\mathbf{Q}_E^t \in [0,1]^{|\mathcal{E}| \times |\mathcal{E}|}$ respectively. In both matrices, each row describes the transition probability from category i to all other categories j such that $\sum_j [\mathbf{Q}_X^t]_{i,j} = \sum_j [\mathbf{Q}_E^t]_{i,j} = 1$ for all i. We can then sample each node and edge at time t (forming graph G^t) given graph G^{t-1} using the transition probability $q(G^t \mid G^{t-1})$, taken as the product of the node-specific transition probabilities $\mathbf{X}^{t-1}\mathbf{Q}_X^t$ and the edge-specific probabilities $\mathbf{E}^{t-1}\mathbf{Q}_E^t$. Furthermore, we can determine the distribution at any time directly from the original graph G^0 using the well-known Chapman-Kolmogorov equation, notated here as $\prod_{\tau=1}^{t} \mathbf{Q}_X^\tau =: \bar{\mathbf{Q}}_X^t$ and $\prod_{\tau=1}^{t} \mathbf{Q}_E^\tau =: \bar{\mathbf{Q}}_E^t$

Reverse Process: The denoising process is estimated using a model ϕ_θ parameterized by θ. This model is trained to directly estimate a graph representing a piece of music $G^0 = (\mathbf{X}^0, \mathbf{E}^0)$ given a noisy graph at any time step $G^t = (\mathbf{X}^t, \mathbf{E}^t)$. We denote the predicted probabilities for each node in the original graph G^0 as $\hat{p}_\mathbf{X} \in [0,1]^{n \times |\mathcal{X}|}$.

In our implementation, edges are predefined and static based on the rhythmic framework of sampled musical material (Section 3.1). This assumption simplifies the diffusion problem considerably, as we are able to set the edge transition matrix to the identity $\mathbf{Q}_E^t = \bar{\mathbf{Q}}_E^t = \mathbf{I}_{|\mathcal{E}|}$. Following [41], we set $\bar{\mathbf{Q}}_\mathbf{X}^t = \bar{\alpha}^t \mathbf{I}_{|\mathcal{X}|} + (1 - \bar{\alpha}^t)\mathbf{1}[\mathbf{m}_\mathbf{X}]'$, where $\mathbf{m}_\mathbf{X}$ is the marginal distribution vector for node types, $[\cdot]'$ is the transpose, and $\mathbf{1}$ is a ones vector. Here, $\bar{\alpha}^t = \prod_{\tau=1}^{t} \alpha^t$ is the noise schedule hyperparameter that goes from 1 to 0 (true data to complete noise) according to the cosine schedule, $[\alpha^t]^2 = f^t / f^0$, where $f^t = \cos(((t/T + s)/(1+s)) \cdot (\pi/2))^2$, and s is a small number (e.g. 0.008) [30]. By freezing the edges of the graph, the reverse diffusion objective is simplified considerably. The DiGress loss (eq. (1) in Appendix A) is reduced to $\mathcal{L}(\hat{p}_\mathbf{X}, \mathbf{X}) = \sum_{i=1}^{n} \text{cross-entropy}(\mathbf{x}_i, [\hat{p}_\mathbf{X}]_i)$, only attending to the predictions for nodes. Further, we only require $\hat{p}_\mathbf{X}$ to estimate reverse diffusion transitions $p_\theta(G^{t-1} \mid G^t) = \prod_{i=1}^{n} p_\theta(\mathbf{x}_i^{t-1} \mid \mathbf{x}_i^t)$ (compare with eqs. (2) and (3) in Appendix A).

The DiGress framework expects nodes with only discrete, one-hot encoded embeddings. However, beyond discrete scale degrees, we include discrete and continuous rhythmic features, bundled in a matrix $\mathbf{R} \in \mathbb{R}^{n \times |\mathcal{R}|}$, where $\mathcal{R}$ represents the set of rhythmic features. Because $\mathbf{R}$ is determined and unchanging from the beginning of the process, it can be incorporated in every denoising iteration to model ϕ_θ (recall Figure 2). More specifically, the input of the denoising model ϕ_θ should be $G^t = \big([\mathbf{X}^t \mid\mid \mathbf{R}], \mathbf{E}\big)$, where $[\cdot \mid\mid \cdot]$ denotes column concatenation. When performing the reverse iterations during inference, we implement Stochastic Control Guidance [20] to avoid certain harmonic intervals, undesired contrapuntal motions, and repetitive melodic lines. Depending on the genre, any quantifiable rules may be added to guide the diffusion process.

3.3 Inference and Phrase Fusion

Music Realization: Because we limited the classes of individual nodes to the global scale degrees (e.g. $\hat{1}$, $\hat{4}$, or $\flat\hat{3}$), they cannot be directly interpreted as music. Rather, they must be mapped to

specific pitches in specific octaves (e.g. C4, F2, or E♭4). The simplest approach is to follow the path of *smoothest voice leading* for each string of nodes connected by forward edges: i.e., starting in a register common to the dataset, for each scale degree we place it according to the smallest interval between the previous note and itself. However, this approach often leads to melodies that go extremely high or low. Instead, we define a central pitch for each voice, which serves as a fall back if a voice gets too far away. If it is possible for consecutive notes to be a step away, they will always follow step-wise motion. If there is a larger interval between consecutive notes, the voice will find the closest note to the central pitch. This approach ensures smooth voice leading is achieved while constraining the voice range.

Rejection Sampling and Musical Analysis: Through diffusion, we generate B musical phrases. Once all phrases are generated (which may be done in parallel), we impose a rule-based rejection sampling to discard poor quality musical phrases. Similar to the Stochastic Control Guidance mentioned in Section 3.2, we reject samples with improper harmonic intervals or contrapuntal motions. Additionally, we analyze the phrase for possible harmonic progressions based on the desired genre. If no harmonic progression can make sense of the phrase, it is discarded.

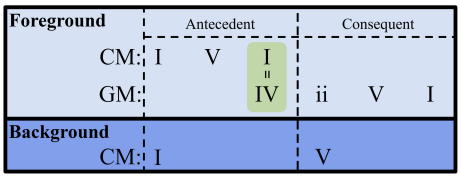

Figure 3: Example phrase fusion via pivot chord modulation from C Major (CM) to G Major (GM). The light and dark blue represent foreground and background analysis, respectively. The *antecedent* is in CM, leading to GM in the *consequent* by reinterpreting the final antecedent "I" as "IV" in the new key.

During the analysis process, we keep track of possible starting and end harmonies and melodic tones. Because important structural events tend to happen at the beginnings and endings of phrases according to Schenkerian theory [11], we can fuse phrases together to match a common Schenkerian structure such as the one found in the purple box of Figure 2. By incorporating such Schenkerian structure, we ensure the generated music has meaningful local and global harmonic variation with direction.

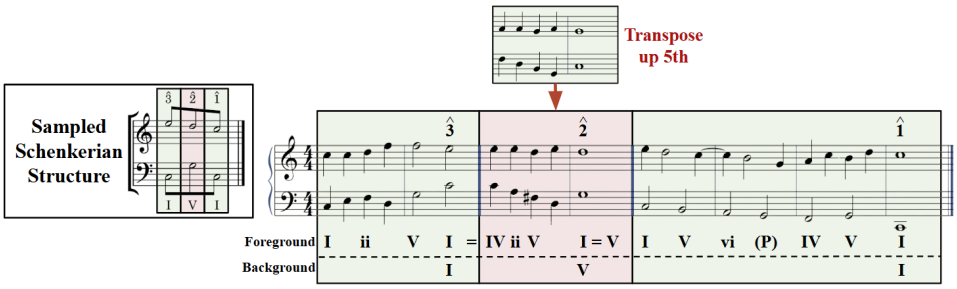

Figure 4: A common Schenkerian structure as three phrases of generated music. Green and red represent music in the home and dominant key, respectively. Here, the 2nd phrase was originally generated in the home key, but is transposed to the dominant via our fusion method in Section 3.3.

Phrase Fusion: To create a smooth transition between phrases, we employ a pivot chord modulation scheme. Say we want to "modulate" from the tonic key "I" in a *antecedent* phrase to the dominant key "V" in a *consequent* phrase (see Figure 3 for example). We first assume all phrases are based in a particular key (e.g. C Major/Minor). If the antecedent ends on a tonic "I" harmony, the consequent can reinterpret the tonic harmony as a surface level subdominant "IV" in the deeper level motion to the dominant "V." Therefore, we search our dictionary of organized phrases for a phrase that begins on a local harmony that typically comes after a "IV" harmony. The sampled phrase can then be transposed to the desired key (dominant "V" in our example here) and appended to the antecedent as the consequent phrase. Similar transitions can move from one key to any other.

Sampling Schenkerian Structure: To determine the overall structure of our generated music, we first gather common Schenkerian structures from the literature. From the set of expert SchAs, we extract the deep middleground structural harmonic progressions and their associated bass and treble notes.

For instance, the most famous structure in SchA is a 3-line *Ursatz* (depicted in Figure 4). The harmonic progression follows a tonic-dominant-tonic ($I - V - I$) structure with root position bass notes and a stepwise descending third in the treble ($\hat{3} - \hat{2} - \hat{1}$). One realization of this structure would involve three phrases. The first phrase would end in the home key with an authentic cadence ($V - I$)

and $\hat{3}$ in the treble voice. The second phrase would end with an authentic cadence in the dominant key (V) with global $\hat{2}$ (local $\hat{5}$) in the treble voice. Finally, phrase three would end with a perfect authentic cadence in the home key; it would end with $V - I$ in the bass and $\hat{1}$ in the treble.

4 Experiments

We ran several experiments, including a human survey, ablation studies, and genre flexibility demonstrations. Full survey results, ablation studies, and implementation details can be found in Appendices B–D. Musical samples and genre flexibility demonstrations may be found on our Huggingface page[2]. Our model was trained on all individual phrases of the Bach chorales that are based in their respective global tonics. We provide the full survey instrument and excerpts in the Supplemental Materials. Reproducible code will be made available pending acceptance.

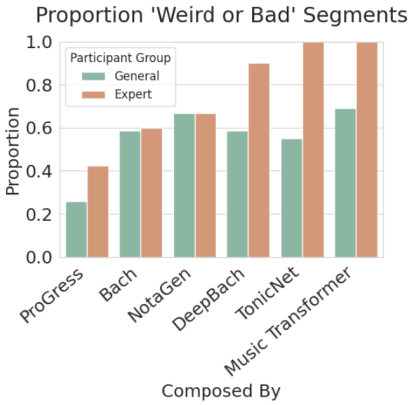

Figure 5: "Weird or bad" survey results.

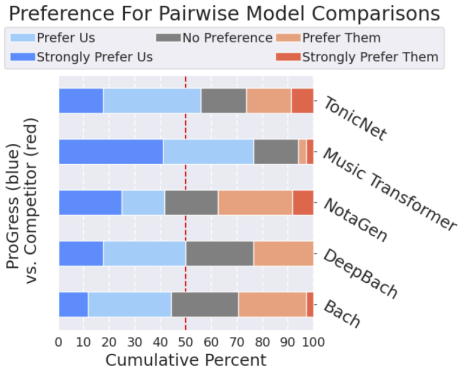

Figure 6: Pairwise model preference for all survey participants.

Survey Design: For our subjective experiments, we use the same survey instrument as [15] and [16]. We compare against Bach and several models specialized for Bach chorale generation: DeepBach [14], NotaGen [42], Music Transformer [19], and TonicNet [33]. For each pair of chorales we asked: **1)** On a scale of 0 (not enjoyable) to 10 (very enjoyable), how would you rate Chorale X? **2)** On a scale of 0 (certain it's by a computer) to 10 (certain it's by a human), what is your degree of belief that a human composed Chorale X? **3)** Which Chorale do you prefer? (a) strongly prefer 1, (b) prefer 1, (c) no clear preference, (d) prefer 2, (e) strongly prefer 2. **4)** Were there any parts of Chorale X that stood out as sounding weird or bad to you? (yes=1, no=0).

Results: Our final dataset consists of 45 participants. Of those, 13 *expert participants* reported studying music privately for more than 5 years and gave correct answers to skill screening questions. We found that ProGress outperformed other methods, *and even Bach*, in all qualitative metrics. Observing the "weird or bad" question results seen in Figure 5, we see that ProGress performs substantially better than other models in both the general and expert participant groups. Bach's score lies comfortably in the middle. We believe this is because ProGress is more structured than other deep learning models and less harmonically adventurous than Bach.

In Figure 6, we see that participants generally prefer ProGress over the competitors. NotaGen nearly tied with ProGress, however our model uses substantially fewer parameters than NotaGen (3 million vs 516 million, respectively). While Bach had around double the proportion of perceived "weirdness" in his music, participants did not show a strong preference for our model over Bach. However, we find that ProGress outperforms Bach in "enjoyability" with statistical significance (see Appendix B).

5 Conclusion

We introduce a hybrid approach in which GNNs and music-theoretical structures and principles work together to produce novel, coherent music in various styles. Through our survey experiment, we show that ProGress's careful music-hierarchical composition style outperforms the stream-of-consciousness approach of several deep learning models.

[2]https://huggingface.co/spaces/cpyang/progress-music-generation

References

[1] Beatoven.ai. https://www.beatoven.ai/. Accessed: August 31, 2023.

[2] Endel. https://endel.io/. Accessed: 2025-03-22.

[3] Infinite Album. https://infinitealbum.io/. Accessed: 2025-03-22.

[4] Riffusion. https://riffusion.com. Accessed: 2025-03-22.

[5] Stable Audio. https://stableaudio.com/. Accessed: 2025-03-22.

[6] Suno AI. https://suno.com. Accessed: 2025-03-22.

[7] Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, et al. Musiclm: Generating music from text. 2023. URL https://arxiv.org/abs/2301.11325.

[8] Anonymous. Autoscha: Automatic hierarchical music representations via multi-relational node isolation. *Under Review*, 2025.

[9] Anonymous. Novel graph link prediction methodology for human-in-the-loop hierarchical music analysis. *Under Review*, 2025.

[10] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023. URL https://arxiv.org/abs/2107.03006.

[11] Allen Clayton Cadwallader, David Gagné, and Frank Samarotto. *Analysis of tonal music: a Schenkerian approach*. Oxford University Press, 1998.

[12] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation, 2020. URL https://arxiv.org/abs/2009.00713.

[13] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[14] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. Deepbach: a steerable model for bach chorales generation. In *International conference on machine learning*, pages 1362–1371. PMLR, 2017.

[15] Stephen Hahn, Rico Zhu, Simon Mak, Cynthia Rudin, and Yue Jiang. An interpretable, flexible, and interactive probabilistic framework for melody generation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 4089–4099. Association for Computing Machinery, 2023.

[16] Stephen Hahn, Jerry Yin, Rico Zhu, Weihan Xu, Yue Jiang, Simon Mak, and Cynthia Rudin. Senthymnent: An interpretable and sentiment-driven model for algorithmic melody harmonization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5050–5060, 2024.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

[18] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions, 2021. URL https://arxiv.org/abs/2102.05379.

[19] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. 2018. URL https://arxiv.org/abs/1809.04281.

[20] Yujia Huang, Adishree Ghatare, Yuanzhe Liu, Ziniu Hu, Qinsheng Zhang, Chandramouli S Sastry, Siddharth Gururani, Sageev Oore, and Yisong Yue. Symbolic music generation with non-differentiable rule guided diffusion, 2024. URL https://arxiv.org/abs/2402.14285.

[21] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Graph neural network for music score data and modeling expressive piano performance. In *International Conference on Machine Learning (ICML)*, pages 3060–3070. PMLR, 2019.

[22] Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations, 2022. URL https://arxiv.org/abs/2202.02514.

[23] Daniel D. Johnson, Jacob Austin, Rianne van den Berg, and Daniel Tarlow. Beyond in-place corruption: Insertion and deletion in denoising probabilistic models, 2021. URL https://arxiv.org/abs/2107.07675.

[24] Nicolas Jonason, Luca Casini, and Bob LT Sturm. Symplex: Controllable symbolic music generation using simplex diffusion with vocabulary priors. 2024. URL https://arxiv.org/abs/2405.12666.

[25] Edward Lee. Ai and the sound of music. *Yale LJF*, 134:187, 2024.

[26] Jing Luo, Xinyu Yang, and Dorien Herremans. Bandcontrolnet: parallel transformers-based steerable popular music generation with fine-grained spatiotemporal features. 2024. URL https://arxiv.org/abs/2407.10462.

[27] Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. 2021. URL https://arxiv.org/abs/2103.16091.

[28] Ed Newton-Rex. Suno is a music ai company aiming to generate $120 billion per year. but is it trained on copyrighted recordings? *Music Business Worldwide*, 2024.

[29] Stephen Ni-Hahn, Weihan Xu, Jerry Yin, Rico Zhu, Simon Mak, Yue Jiang, and Cynthia Rudin. A new dataset, notation software, and representation for computational schenkerian analysis. *arXiv preprint arXiv:2408.07184*, 2024.

[30] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models, 2021. URL https://arxiv.org/abs/2102.09672.

[31] Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling, 2020. URL https://arxiv.org/abs/2003.00638.

[32] Drew F. Nobile. *A structural approach to the analysis of rock music*. PhD thesis, 2014.

[33] Omar A Peracha. Improving polyphonic music models with feature-rich encoding. *arXiv preprint arXiv:1911.11775*, 2020. doi: 10.5281/ZENODO.4245396. URL https://zenodo.org/record/4245396.

[34] Matthias Plasser, Silvan Peter, and Gerhard Widmer. Discrete diffusion probabilistic models for symbolic music generation. 2023. URL https://arxiv.org/abs/2305.09489.

[35] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *International Conference on Machine Learning*, pages 4364–4373. PMLR, 2018.

[36] Felix Salzer. Tonality in early medieval polyphony. *The Music Forum I*, pages 35–98, 1967.

[37] Heinrich Schenker. *Free Composition: Volume III of new musical theories and fantasies*, volume 1. Pendragon Press, 1935.

[38] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. pmlr, 2015.

[39] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

[40] Jonathan Stock. The application of Schenkerian Analysis to Ethnomusicology: problems and possibilities. *Music Analysis*, 12(2):215–240, 1993.

[41] Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation, 2023. URL https://arxiv.org/abs/2209.14734.

[42] Yashan Wang, Shangda Wu, Jianhuai Hu, Xingjian Du, Yueqi Peng, Yongxin Huang, Shuai Fan, Xiaobing Li, Feng Yu, and Maosong Sun. Notagen: Advancing musicality in symbolic music generation with large language model training paradigms, 2025. URL https://arxiv.org/abs/2502.18008.

[43] Jian Wu, Changran Hu, Yulong Wang, Xiaolin Hu, and Jun Zhu. A hierarchical recurrent neural network for symbolic melody generation. *IEEE Transactions on Cybernetics*, 50(6):2749–2757, 2019.

[44] Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. Diffsound: Discrete diffusion model for text-to-sound generation, 2023. URL https://arxiv.org/abs/2207.09983.

[45] Yi Zou, Pei Zou, Yi Zhao, Kaixiang Zhang, Ran Zhang, and Xiaorui Wang. Melons: generating melody with long-term structure using transformers and structure graph. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 191–195. IEEE, 2022.

Appendix

A. Diffusion Preliminaries and DiGress Details

Denoising Diffusion Probabilistic Models

Denoising Diffusion Probabilistic Models (DDPMs; introduced by [38]) aim to generate meaningful data (e.g. images or audio) by *denoising* corrupted data. There are two main processes involved in DDPMs that assume a Markov process: the forward (encoding) process $q(\mathbf{X}^{1:T} \mid \mathbf{X}^0) = \prod_{t=1}^{T} q(\mathbf{X}^t \mid \mathbf{X}^{t-1})$, where $\mathbf{X}^t$ is the data after $t = 1, \ldots, T$ steps of corruption or noise addition, and the reverse (decoding) process $p(\mathbf{X}^{0:T}) = p(\mathbf{X}^T) \prod_{t=1}^{T} p(\mathbf{X}^{t-1} \mid \mathbf{X}^t)$, which aims to undo the data corruption process or find novel clean data from noise (Figure 7).

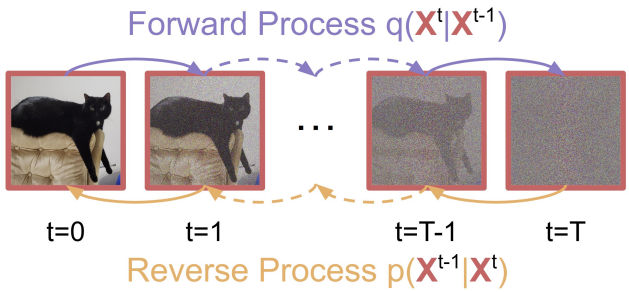

Figure 7: Example of the diffusion process on image data of the first author's cat.

Most work in continuous spaces defines the distributions for forward and reverse precesses to be Gaussian [17, 39, 12, 30]. Even when dealing with categorical data, Gaussian noise is common; categories are treated as one-hot encodings with continuous values [31, 22]. Many works have adapted diffusion for discrete spaces [18, 23, 44, 10].

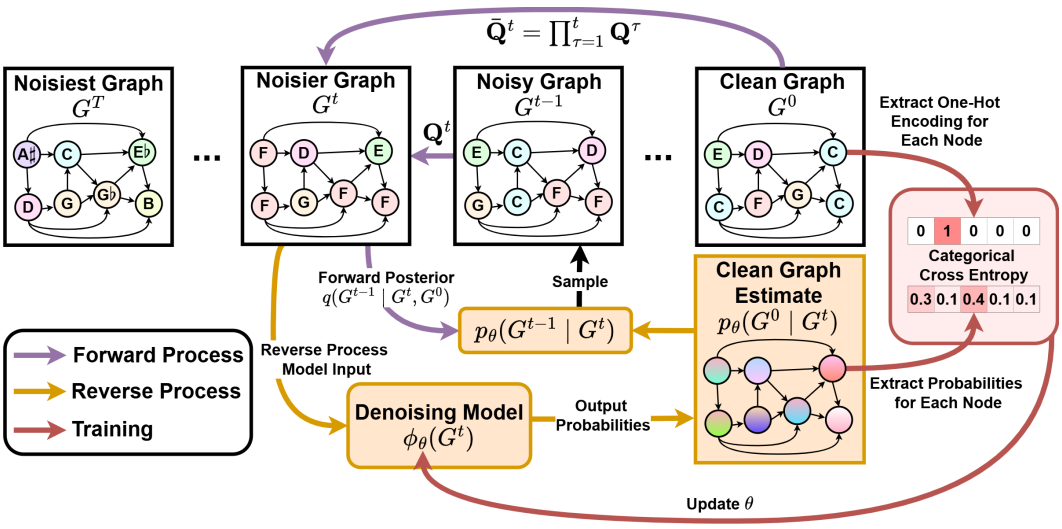

Figure 8: Overview of the DiGress model for our music application adapted from Figure 1 of [41].

Our model follows the setting of [10] and [41], where a data point $\mathbf{x}^0 \in \{0,1\}^d$ is a one-hot encoding of d categories and the noise is represented by a series of transition matrices $(\mathbf{Q}^1, \ldots, \mathbf{Q}^T)$. These transition matrices are defined such that $[\mathbf{Q}^t]_{i,j}$ represents the probability of moving from state i to state j and $q(\mathbf{x}^t \mid \mathbf{x}^{t-1}) = \mathbf{x}^{t-1}\mathbf{Q}^t \in [0,1]^d$.

Discrete Diffusion with Graph Neural Networks

Graphs are a natural medium to represent hierarchy in music [21, 29]. Methods to extract meaningful features from graphs are thus of critical interest, and in the context of deep learning, Graph Neural Networks (GNNs) stand out for their effectiveness. GNNs generalize the discrete convolutions of Convolutional Neural Networks (CNNs) to graphs, where filters perform local neighborhood aggregation over the node space. Following the work of [21, 29], we consider GNNs that operate over heterogeneous, directed graphs.

We are particularly interested in the discrete graph diffusion setting introduced by [41], visualized in Figure 8. Given a set of node categories $\mathcal{X}$ and edge categories $\mathcal{E}$, a graph $G = (\mathbf{X}, \mathbf{E})$ is comprised of node embedding matrix $\mathbf{X} \in \{0,1\}^{n \times |\mathcal{X}|}$, where each row is a one-hot encoding $\mathbf{x}_i \in \{0,1\}^{|\mathcal{X}|}$ for graph nodes $i = 1, \ldots, n$, and edge embedding tensor $\mathbf{E} \in \{0,1\}^{n \times n \times |\mathcal{E}|}$, which describes each edge $\mathbf{e}_{i,j} \in \{0,1\}^{|\mathcal{E}|}$ from node i to node j as a one-hot encoding. Note that the absence of an edge or node is represented as a particular class. Thus, all $\mathbf{x}_i$ and $\mathbf{e}_{i,j}$ are non-empty and have one entry indicating its category.

Forward Process

Discrete graph diffusion applies noise independently to each node and edge (like pixels in image diffusion). At each forward diffusion step $1, \ldots, t, \ldots, T$, node and edge class transition probability matrices are defined as $\mathbf{Q}_X^t \in [0,1]^{|\mathcal{X}| \times |\mathcal{X}|}$ and $\mathbf{Q}_E^t \in [0,1]^{|\mathcal{E}| \times |\mathcal{E}|}$ respectively. In both matrices, each row describes the transition probability from category i to all other categories j such that $\sum_j [\mathbf{Q}_X^t]_{i,j} = \sum_j [\mathbf{Q}_E^t]_{i,j} = 1$ for all i. We can then sample each node and edge at time t (forming graph G^t) given graph G^{t-1} using the following categorical distribution:

$$q(G^t \mid G^{t-1}) = \left(\mathbf{X}^{t-1}\mathbf{Q}_X^t, \mathbf{E}^{t-1}\mathbf{Q}_E^t\right).$$

Furthermore, we can determine the distribution at any time directly from the original graph G^0 using the well-known Chapman-Kolmogorov equation:

$$q(G^t \mid G^0) = \left(\mathbf{X}^0 \prod_{\tau=1}^{t} \mathbf{Q}_X^\tau, \;\; \mathbf{E}^0 \prod_{\tau=1}^{t} \mathbf{Q}_E^\tau\right) =: \left(\mathbf{X}^0\, \bar{\mathbf{Q}}_X^t, \mathbf{E}^0\, \bar{\mathbf{Q}}_E^t\right).$$

Reverse Process

The denoising process is estimated using a model ϕ_θ parameterized by θ. This model is trained to directly estimate a graph representing a piece of music G^0 given a noisy graph at any time step G^t. We denote the predicted probabilities for each node in the original graph G^0 as $\hat{p}_{\mathbf{X}} \in [0,1]^{n \times |\mathcal{X}|}$. To avoid clutter, the time superscript 0 (indicating variables without noise) is implicit in our notation for $\mathbf{X}$, $\mathbf{E}$, $\mathbf{x}$, and $\mathbf{e}$ when no superscript is written. The model is optimized using the cross-entropy loss,

$$\mathcal{L}(\hat{p}_G, G) = \sum_{i=1}^{n} \text{cross-entropy}\Big(\mathbf{x}_i, \big[\hat{p}_{\mathbf{X}}\big]_i\Big) + \lambda \sum_{i=1}^{n}\sum_{j=1}^{n} \text{cross-entropy}\Big(\mathbf{e}_{i,j}, [\hat{p}_{\mathbf{E}}]_{i,j}\Big), \tag{1}$$

where λ controls the attention balance between edge and node predictions.

The trained denoising model can then be used to sample new graphs, using its predictions $\hat{p}_{\mathbf{X}}$ to estimate reverse diffusion iterations. We model the problem as

$$p_\theta(G^{t-1}|G^t) = \prod_{i=1}^{n} p_\theta(\mathbf{x}_i^{t-1}|\mathbf{x}_i^t) \prod_{i=1}^{n}\prod_{j=1}^{n} p_\theta(\mathbf{e}_{i,j}^{t-1}|G^t). \tag{2}$$

Each term is computed by marginalizing over network predictions,

$$p_\theta(\mathbf{x}_i^{t-1}|\mathbf{x}_i^t) = \sum_{c=1}^{|\mathcal{X}|} p_\theta\Big(\mathbf{x}_i^{t-1} \mid \mathbf{x}_i^0 = \mathbb{1}_c, \mathbf{x}_i^t\Big)\Big[\hat{p}_{\mathbf{X}}\Big]_{i,c} \tag{3}$$

where $\mathbb{1}_c$ is the one-hot encoding for class c and we choose

$$p_\theta\Big(\mathbf{x}_i^{t-1} \mid \mathbf{x}_i^0 = \mathbb{1}_c, \mathbf{x}_i^t\Big) = \begin{cases} q(\mathbf{x}_i^{t-1} \mid \mathbf{x}_i^0 = \mathbb{1}_c, \mathbf{x}_i^t) & \text{if } q(\mathbf{x}_i^t \mid \mathbf{x}_i^0 = \mathbb{1}_c) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Edge transitions are computed in a similar fashion. Graphs are then iteratively sampled using these distributions, where the new graph is used as input for the denoising model ϕ_θ at the next time step.

B. Full Qualitative Survey Results

Our survey[3] begins with the following instructions: "For the following survey, you will be presented with several pairs of Chorales that aim to imitate the style of J.S. Bach. For each pair of Chorales, you will first be asked to listen to them completely, then answer a series of simple questions. There are 4 total comparisons. Thank you for your time!"

Our survey includes a few screening questions: 1) how often do you listen to music, 2) Have you ever studied music with a private teacher? If so, for how long, 3) What meter best fits [an excerpt of *Ah! Vous dirai-je, maman*], and 4) What is the name of the melodic interval of [two melodic notes]? Self-reported results for experience and skill questions may be found in Table 1. Skill question responses are divided between a "nonsense," "wrong," and "correct" answers, where "nonsense" answers use terminology that is not used in music theory. For weekly music listening, 1 reported less than an hour, 33 reported between 1 and 15 hours, and 11 reported more than 15 hours.

Table 1: Screening question results broken down by reported experience.

	Meter			Interval		
Experience	Nonsense	Wrong	Correct	Nonsense	Wrong	Correct
0 years	4	2	10	6	2	8
$<$ 5 years	0	1	8	1	1	7
$\geq$ 5 years	2	0	18	3	3	14

[3]Full preview of survey instrument here: https://duke.yul1.qualtrics.com/jfe/preview/previewId/baea6f01-5f30-47b4-b056-f4a9abdb30df/SV_1zVCXYMgF4KDZS6?Q_CHL=preview&Q_SurveyVersionID=current

For model comparisons, we note that an official implementation of Music Transformer [19] is not publicly available, so we trained a model based on https://github.com/gwinndr/MusicTransformer-Pytorch, which has been used for experiments by [20].

For enjoyability, we compare the mean of each competing excerpt vs. ProGress using a paired t-test (Table 3). Similarly, we evaluate mean confidence that each excerpt was composed by a human compared to the actual human-composed excerpt using a paired t-test (Table 4). We determine binomial confidence intervals for the proportion of participants that strictly preferred ProGress compared to the competitors, excluding "no preference" responses from being counted in favor of ProGress (Table 2). Finally, we evaluate whether there is evidence for a difference in the proportion of respondents that identified a "weird or bad" sounding excerpt for each competing excerpt vs. ProGress using a chi-square test.

Table 2: Proportion of respondents strictly preferring ProGress (higher is better).

Method	Proportion	95% CI
vs. TonicNet	0.56	(0.29, 0.61)
vs. Music Transformer	0.76	(0.60, 0.88)
vs. DeepBach	0.50	(0.34, 0.66)
vs. NotaGen	0.42	(0.24, 0.61)
vs. Bach	0.44	(0.29, 0.61)

In Table 2 we show the Clopper-Pearson binomial confidence intervals for the proportion of participants that strictly preferred ProGress over competitors. Note that we exclude "no preference" participants being counted in favor of ProGress, handicapping our score.

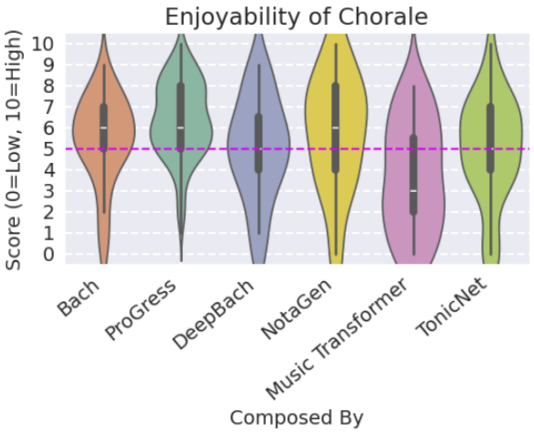

Figure 9: Enjoyability survey results.

Figure 9 shows that participants found all models generally enjoyable except Music Transformer. Table 3 shows that within a general population sample, ProGress is statistically more enjoyable than Bach and all other models except NotaGen.

Table 3: Enjoyability (higher is better).

Method	Mean	95% CI	p-value
ProGress	6.37	(6.08, 6.66)	ref.
Bach	5.47	(4.80, 6.14)	0.011
DeepBach	5.00	(4.21, 5.79)	<0.001
NotaGen	5.75	(4.59, 6.91)	0.152
Music Transformer	3.68	(2.81, 4.54)	<0.001
TonicNet	5.12	(4.27, 5.96)	0.001

Figure 10 and Table 4 show that participants are generally uncertain about whether the excerpts are written by a human or not. Still, ProGress clearly outperforms other models.

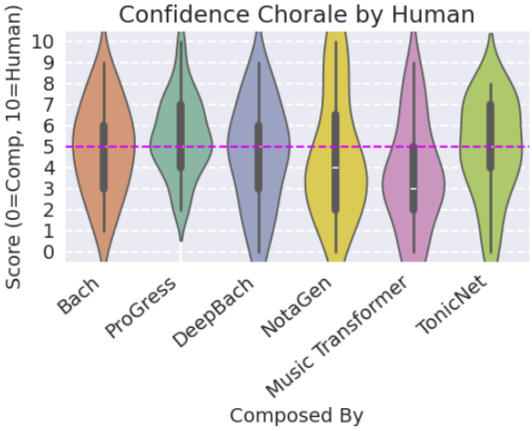

Figure 10: Turing test survey results.

Table 4: Confidence of being composed by human (higher is better).

Method	Mean	95% CI	p-value
Bach	4.76	(4.06, 5.47)	ref.
ProGress	5.68	(5.34, 6.03)	0.162
DeepBach	4.09	(3.25, 4.92)	0.212
NotaGen	4.63	(3.38, 5.87)	0.664
Music Transformer	2.76	(1.98, 3.55)	<0.001
TonicNet	4.06	(3.21, 4.90)	0.196

C. Ablation Study

For our ablation studies, we experiment by removing various features within the $\mathbf{R}$ matrix, plus removing the $\mathbf{R}$ matrix altogether. Indeed, we find that the $\mathbf{R}$ matrix is vital to the performance of the network, improving validation loss by approximately 14%. We report the minimum validation loss for ablated models over 3 runs in Table 5.

Table 5: Minimum validation loss for ablated models

	Full	No $\mathbf{R}$	No metric strength	No duration	No offset
Validation Loss	21.48	25.92	21.80	23.57	24.08

We also experiment ablating the stochastic control guidance during diffusion inference. Unfortunately, rule guidance did not significantly improve our strict rule-based rejection rate when applied to Bach chorales. The rate when generating 40 samples with and without rule guidance went from 75% to 77.5% respectively (lower is better). We hypothesize that larger improvements may be accomplished in other genres with more flexible rules, but leave this to future work.

D. Implementation Details

Our model code is in available on Github[4]. In our experiments, our denoising diffusion model consisted of 4 convolutional layers with hidden dimension 256, 8 attention heads, and ran through 100 diffusion steps. It was trained for up to 150 epochs with a batch size 8, using the Adam optimizer. We used a training/validation split of 90/10. These hyperparameters were chosen based on empirical

[4]https://github.com/stephenHahn88/ProGress_Supplement

performance on the Bach chorales. We used a single RTX 3060 6gb GPU, which was able to train a full ProGress model in approximately 47 minutes.

For inference, we generated several hundred phrases and rejected samples that did not follow strict contrapuntal rules based on music theoretical principles of Bach's time. This process took under a minute. These rules included avoiding parallel 5ths and 8ves, avoiding dissonant harmonic intervals (2nds and 4ths) on strong beats, and avoiding improbable harmonic progressions (e.g. V -> IV). These rules may be loosened or adapted for various genres.