# PDMX: A Large-Scale Public Domain MusicXML Dataset for Symbolic Music Processing

**Phillip Long** **Zachary Novack** **Taylor Berg-Kirkpatrick** **Julian McAuley**
University of California – San Diego
{p1long, znovack, tberg, jmcauley}@ucsd.edu

## Abstract

The recent explosion of generative AI-Music systems has raised numerous concerns over data copyright, licensing music from musicians, and the conflict between open-source AI and large prestige companies. As these issues highlight the need for publicly available, copyright-free musical data, we present **PDMX**: a large-scale open-source dataset of over 250K public domain MusicXML scores collected from the score-sharing forum MuseScore, making it the largest available copyright-free symbolic music dataset to our knowledge. PDMX additionally includes a wealth of both tag and user interaction metadata, allowing us to efficiently analyze the dataset and filter for high quality user-generated scores. We conduct multitrack music generation experiments evaluating how different representative subsets of PDMX lead to different behaviors in downstream models, and how user-rating statistics can be used as an effective measure of data quality.

## 1 Introduction

The current explosion in generative music systems (1; 2; 3; 4; 5; 6) has increased debate over the legality of training on unlicenced data, and in particular, how such models may replace artists and interact with modern copyright law (7; 8; 9; 10; 11). Such concerns over replacing artists have renewed interest in *symbolic* music processing, which allows for greater artist-in-the-loop interaction (5). While a natural alternative to the intense music licensing process is to train on public domain data, this remains complex, as many symbolic music datasets fail to adequately vet for copyrighted music (12; 13; 14; 15; 16), let alone explicitly filter for public domain works (17; 18; 19) (see App. A.1).

We thus introduce **PDMX**: a large-scale dataset of over 250K **P**ublic **D**omain **M**usic**X**ML files, the largest publicly available, copyright-free MusicXML dataset in existence to our knowledge. PDMX includes genre, tag, description, and popularity metadata for every file. As a test bed, we use an unconditional multitrack music generation task to understand how metadata-based data filtering affects downstream modeling, highlighting that the user rating data in PDMX can be a powerful measure of data quality. PDMX is released on Zenodo[1] under a CC-BY license as a *fully commercially viable* dataset for symbolic music processing.

## 2 Dataset

PDMX was created using data scraped from the online score-sharing forum MuseScore[2] (20), explored further in App. A.2. We additionally scraped metadata for each score, which includes both descriptions of the score (i.e. composer, genre), community content in the form of ratings and comments, and most importantly the overall license for each score. We then selected songs with specifically Public Domain Mark or CC-0 licenses. In total, our initial scrape yielded 254,077 files, the largest dataset of MusicXML-like data to our knowledge, with a total 6,250.37 hours of music (see App. A.6).

[1] https://zenodo.org/records/14004826
[2] https://musescore.com

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Two issues arise from the crowd-sourced nature of the MuseScore platform: we cannot assume that all songs in the base PDMX dataset are either high quality or unique. To address data quality, we employ the metadata's "rating" attribute, which represents the average of users' five-star ratings (while a rating of zero indicates a song is unrated). An analysis of ratings can be found in Sec A.3. Meanwhile, because heavy duplication within a dataset can introduce bias towards high frequency data points (21), for each song, we encode a combination of title and artist using a pre-trained text embedding model, Sentence-BERT (22), to deduplicate and ensure uniqueness. Furthermore, for each group of duplicates, we include the best unique instrumentations (e.g. solo piano vs. string quartet) and arrangements (e.g. easy vs. advanced versions), for which the underlying algorithm is detailed in App. A.4.

Given the metadata and filtering/deduplication process, we thus focus on four main subsets of the data: (1) **All (A)**: The entirety of PDMX. (2) **Deduplicated (D)**: The collection of each song's best unique arrangements. (3) **Rated (R)**: Songs with non-zero ratings. (4) **Rated and Deduplicated (R∩D)**: The intersection of the previous two subsets.

## 3 Experiments

In order to demo PDMX as a dataset for symbolic music processing, we follow previous work (3) and focus on an unconditional multitrack music generation task, explained further in App. A.8. We focus on our four main subsets of PDMX (detailed in Tab. 2), as well as a **Random** subset of songs sampled from the full dataset at the size of the **R∩D** subset. We additionally reserve the top 50% (in terms of rating, specifically >4.74 stars) from the **R∩D** subset as a potentially "high quality" subset for fine-tuning. In this setup, we seek to answer two questions: (1) Do data quality and deduplication affect symbolic modeling? and (2) Does fine-tuning on small but high quality data meaningfully change behavior? We report objective results in App. A.9. Examples can be found in our demo.[3]

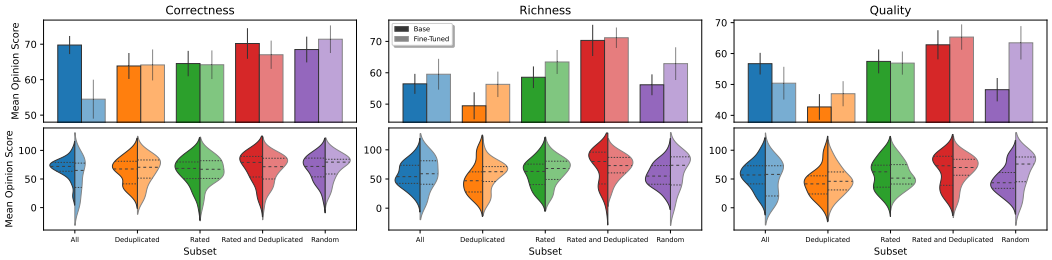


Figure 1: Results from the subjective listening test. For each subset and rating axis, the mean opinion scores of the base and fine-tuned models are displayed on the left and right, respectively. The **R∩D** (red) subset performs the best, and fine-tuning on >50% rated data improves richness and quality.

In Fig. 1, we report the mean opinion scores along three axes for each model from a subjective listening test, elaborated on in App. A.10. Comparing different modeling subsets (ignoring fine-tuning), we observe consistent musical correctness across all subsets, but we also find that the **R∩D** subset (i.e. our most filtered subset, in red), consistently displays the highest scores for both richness and quality. Regarding fine-tuning, we find that this process increases richness in *all* models, and improves quality in three (**D**, **R∩D**, **Random**), with a particularly strong effect on the **Random** model. These results together demonstrate our deduplication and filtering strategies' capacity for significant dataset distillation (as the **R∩D** model performs best despite seeing only 15% of the data).

## 4 Conclusion

We present **PDMX**: the largest dataset of public domain MusicXML files to our knowledge, showing promising results for unconditional multitrack generation that indicate improved performance when filtering for high-quality scores. In the future, we hope to investigate ways to use the large dataset as an effective pretraining mechanism for symbolic generation models, and to utilize PDMX's wealth of extra performance directive annotations for fine-grained time-located text-to-music generation. We also plan to explore PDMX's capacity for discriminative MIR tasks.

[3] `https://pnlong.github.io/PDMX.demo`

## References


[1] S. Forsgren and H. Martiros, "Riffusion: Stable diffusion for real-time music generation," 2022. [Online]. Available: https://riffusion.com/about

[2] StabilityAI, "Stable audio: Fast timing-conditioned latent audio diffusion," Nov 2023. [Online]. Available: https://stability.ai/research/stable-audio-efficient-timing-latent-diffusion

[3] H.-W. Dong, K. Chen, S. Dubnov, J. McAuley, and T. Berg-Kirkpatrick, "Multitrack music transformer," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[4] A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi *et al.*, "MusicLM: Generating music from text," *arXiv:2301.11325*, 2023.

[5] J. Thickstun, D. Hall, C. Donahue, and P. Liang, "Anticipatory music transformer," *arXiv preprint arXiv:2306.08620*, 2023.

[6] K. Chen, Y. Wu, H. Liu, M. Nezhurina, T. Berg-Kirkpatrick, and S. Dubnov, "MusicLDM: Enhancing novelty in text-to-music generation using beat-synchronous mixup strategies," *arXiv:2308.01546*, 2023.

[7] A. Majumdar, "Facing the music: The future of copyright law and artificial intelligence in music industry," *Available at SSRN 4754032*, 2023.

[8] O. Bulayenko, J. P. Quintais, D. J. Gervais, and J. Poort, "Ai music outputs: Challenges to the copyright legal framework," *Available at SSRN 4072806*, 2022.

[9] E. Drott, "Copyright, compensation, and commons in the music ai industry," *Creative Industries Journal*, vol. 14, no. 2, pp. 190–207, 2021.

[10] P.-H. Chen and P.-L. P. Rau, "Using artificial intelligence in music creation? a survey based on copyright consciousness," in *International Conference on Human-Computer Interaction*. Springer, 2024, pp. 210–225.

[11] B. L. Sturm, M. Iglesias, O. Ben-Tal, M. Miron, and E. Gómez, "Artificial intelligence and music: open questions of copyright law and engineering praxis," in *Arts*, vol. 8, no. 3. MDPI, 2019, p. 115.

[12] J. Liu, Y. Dong, Z. Cheng, X. Zhang, X. Li, F. Yu, and M. Sun, "Symphony generation with permutation invariant language model," *arXiv preprint arXiv:2205.05448*, 2022.

[13] Wikifonia. [Online]. Available: http://www.wikifonia.org/

[14] Z. Wang, K. Chen, J. Jiang, Y. Zhang, M. Xu, S. Dai, X. Gu, and G. Xia, "Pop909: A pop-song dataset for music arrangement generation," *arXiv preprint arXiv:2008.07142*, 2020.

[15] M. Zeng, X. Tan, R. Wang, Z. Ju, T. Qin, and T.-Y. Liu, "Musicbert: Symbolic music understanding with large-scale pre-training," *arXiv preprint arXiv:2106.05630*, 2021.

[16] J. Ens and P. Pasquier, "Building the metamidi dataset: Linking symbolic and audio musical data." in *ISMIR*, 2021, pp. 182–188.

[17] C. Raffel, *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. Columbia University, 2016.

[18] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C.-Z. A. Huang, S. Dieleman, E. Elsen *et al.*, "Enabling factorized piano music modeling and generation with the MAESTRO dataset," in *International Conference on Learning Representations (ICLR)*, 2019.

[19] H.-T. Hung, J. Ching, S. Doh, N. Kim, J. Nam, and Y.-H. Yang, "Emopia: A multi-modal pop piano dataset for emotion recognition and emotion-based music generation," *arXiv preprint arXiv:2108.01374*, 2021.

[20] W. Xu, J. McAuley, T. Berg-Kirkpatrick, S. Dubnov, and H.-W. Dong, "Generating symbolic music from natural language prompts using an llm-enhanced dataset," *arXiv preprint arXiv:2410.02084*, 2024.

[21] K. Lee, D. Ippolito, A. Nystrom, C. Zhang, D. Eck, C. Callison-Burch, and N. Carlini, "Deduplicating training data makes language models better," *arXiv preprint arXiv:2107.06499*, 2021.

[22] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[23] H.-W. Dong, K. Chen, J. McAuley, and T. Berg-Kirkpatrick, "Muspy: A toolkit for symbolic music generation," *arXiv preprint arXiv:2008.01951*, 2020.

[24] M. S. Cuthbert and C. Ariza, "music21: A toolkit for computer-aided musicology and symbolic music data," 2010.

[25] S.-L. Wu and Y.-H. Yang, "The jazz transformer on the front line: Exploring the shortcomings of ai-composed music through quantitative measures," *arXiv preprint arXiv:2008.01307*, 2020.

[26] R. Pandey and S. Taruna, "Prevalent exact string-matching algorithms in natural language processing: a review," in *Journal of Physics: Conference Series*, vol. 1854, no. 1. IOP Publishing, 2021, p. 012042.

[27] H.-C. Chen and A. L. Chen, "A music recommendation system based on music and user grouping," *Journal of Intelligent Information Systems*, vol. 24, pp. 113–132, 2005.

[28] P. Lisena, A. Meroño-Peñuela, and R. Troncy, "Midi2vec: Learning midi embeddings for reliable prediction of symbolic music metadata," *Semantic Web*, vol. 13, no. 3, pp. 357–377, 2022.

[29] Z. Novack, J. McAuley, T. Berg-Kirkpatrick, and N. J. Bryan, "DITTO: Diffusion inference-time t-optimization for music generation," 2024.

[30] S. Ding, Z. Liu, X. Dong, P. Zhang, R. Qian, C. He, D. Lin, and J. Wang, "Songcomposer: A large language model for lyric and melody composition in song generation," *arXiv preprint arXiv:2402.17645*, 2024.

[31] Z. Sheng, K. Song, X. Tan, Y. Ren, W. Ye, S. Zhang, and T. Qin, "Songmass: Automatic song writing with pre-training and alignment constraint," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 15, 2021, pp. 13 798–13 805.

[32] I. Ogawa and M. Morise, "Tohoku kiritan singing database: A singing database for statistical parametric singing synthesis using japanese pop songs," *Acoustical Science and Technology*, vol. 42, no. 3, pp. 140–145, 2021.

[33] J. Gardner, I. Simon, E. Manilow, C. Hawthorne, and J. Engel, "Mt3: Multi-task multitrack music transcription," *arXiv preprint arXiv:2111.03017*, 2021.

[34] D. von Rütte, L. Biggio, Y. Kilcher, and T. Hofmann, "Figaro: Generating symbolic music with fine-grained artistic control," *arXiv preprint arXiv:2201.10936*, 2022.

[35] Y.-S. Huang and Y.-H. Yang, "Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions," in *Proceedings of the 28th ACM international conference on multimedia*, 2020, pp. 1180–1188.

[36] S.-L. Wu and Y.-H. Yang, "Compose & embellish: Well-structured piano performance generation via a two-stage approach," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

# A Appendix

## A.1 Related Works

Table 1: Summary of Existing Datasets. PDMX is the largest MusicXML dataset, competitive with existing large scale MIDI datasets, and the largest dataset of public domain data.

| Dataset | Format | Hours | Size | Multitrack | License |
|---|---|---|---|---|---|
| Lakh MIDI (LMD) (17) | MIDI | >9,000 | 174,533 | ✓ | CC-BY 4.0† |
| SymphonyNet (12) | MIDI | >3,200 | 46,359 | ✓ | |
| MAESTRO (18) | MIDI | 201.21 | 1,282 | | CC BY-NC-SA 4.0 |
| Wikifonia Lead Sheet Dataset (13) | MusicXML | 198.40 | 6,405 | | |
| POP909 (14) | MIDI | 60.00 | 909 | | |
| EMOPIA (19) | MIDI | 11.0 | 1,078 | | CC BY-NC-SA 4.0 |
| MMD (15) | MIDI | - | 1,524,557 | | |
| MetaMidi (16) | MIDI | - | 612,088 | ✓ | |
| PDMX | MusicXML | 6,250 | 254,077 | ✓ | CC-0 / Public Domain |

† While LMD is released under CC-BY 4.0, it has been publically documented that many copyrighted works exist in the dataset (5).

There exist a number of publicly available datasets for modeling symbolic music (see Tab. 1), with a wide variety of licensing scenarios and sizes. Most datasets, such as the moderate-sized SymphonyNet (12), the large MMD (15) and MetaMidi (16), and the smaller Wikifonia (13) and POP909 (14) datasets offer no clear licensing information, being thus unsafe for copyright issues. While some existing datasets do have explicit licenses, these are either specifically *non-commercial*, such as for MAESTRO (18) and EMOPIA(19), or have documented licensing violations as in LMD (17). Additionally, few large datasets contain diverse *multitrack* music, and fewer still present data in the MusicXML format, thus ommiting the notational information present in MusicXML but unsupported in datatypes like MIDI. Unlike past datasets, PDMX is a large-scale set comprised entirely of CC-0 / Public Domain MusicXML files, containing diverse multitrack symbolic music with song metadata.

## A.2 Parsing MuseScore Data

Unlike MIDI, MusicXML files are meant for rendering sheet music *as it should be read by a musician* (rather than how it directly sounds), thus containing a wealth of notational information that is common in western music scores. Existing software capable of parsing these files, namely the Python library MusPy (23), is not optimized for the task. Given a piece of symbolic music, the resulting MusPy *Music* object primarily extracts the note value information, as well as limited metadata like time/key signatures and tempo markings. However, MusPy fails to parse many MusicXML-specific score objects, including performance directives (e.g. dynamics, articulations), lyrics, and phrase boundaries, stripping out such content from the notes themselves into a catchall “annotation" structure.

We design our own data structure, *MusicRender*, as an extension of the MusPy *Music* object to specifically account for these flaws. *MusicRender* supports expanded functionality for parsing MusicXML-specific score attributes using multiple of distinct Python objects for each type of non-note score object (such as rehearsal marks or dynamic text). In particular, MusicRender uses such annotations, along with a number of heuristics, to parse MusicXML files such that its symbolic domain outputs reflect the real perceptual rendering of the notes (i.e. how notes should be *performed*). For instance, note durations for a slurred section extend to the next note, and accented notes are louder. Tempo-related annotations, like *ritardandos*, and volume-related features, like *crescendos*, are realized through changes in tempo and velocity.

Similar to MusPy, a *MusicRender* object supports full data I/O to a number of file types (JSON, MIDI), MIR modeling representations (e.g. piano-roll, music21 (24)), and programmatic audio synthesis via Fluidsynth. We thus store PDMX as *MusicRender* JSON files, which are readily loaded into Python environments as *MusicRender* objects with *no* information loss.

## A.3 Data Quality

A past problem in existing symbolic music datasets is overall dataset quality, as very limited works exist to assess the “quality" of symbolic music, and existing high-quality datasets are much smaller (14). Due to the crowd-sourced nature of the MuseScore platform, we cannot assume that all songs

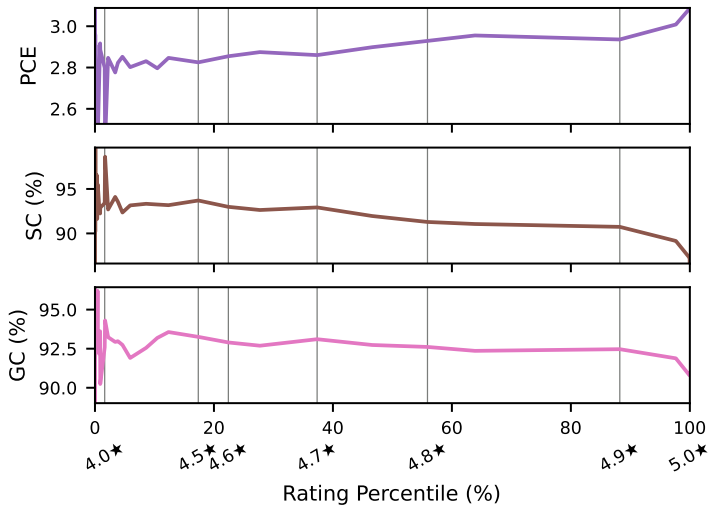


Figure 2: Musical statistics vs. rating percentile in PDMX (specific ratings shown as vertical lines). Higher-rated songs seem to be more harmonically dynamic (higher PCE, lower SC), yet rating has little effect on rhythm.

in the base PDMX dataset are of high quality. While we could use various proxies for quality such as number of views or whether a song has a paywall, we can instead harness MuseScore's crowd-sourcing to our benefit. MuseScore allows individual users to rate a song out of five stars, accessible through the metadata's "rating" attribute. We can use a song's five-star rating average as our metric for data quality.

We refer to the 14,182 rated songs (5.6% of all songs) as the rated subset (with a rating of zero indicating that a song is unrated). In practice, ratings actually range from 2.83 to 4.98 stars, In Tab. 2, where we display the Pitch Class Entropy (PCE), Scale Consistency (SC), and Groove Consistency (GC) (following past work (3; 25; 5)), we see that rated songs display more harmonic variation (higher PCE, lower SC) than their unrated counterparts, though a song's rating appears to have little effect on rhythm. Additionally in Fig. 2, we show these metrics as a function of the rating percentile, displaying that this trend of increasing PCE/decreasing SC holds as the rating increases.

### A.4 Deduplication

Deduplication is an important part of dataset construction, as heavy duplication within a dataset may bias the dataset towards particular high frequency data points and thus degrade downstream modeling tasks (21). However, unlike in more traditional domains like text processing (26), string matching techniques using the textual metadata are insufficient for detecting duplicates in PDMX. This is because many of the same pieces having different titles, such as "Pachelbel's Canon in D" and "Canon by Pachelbel", despite being the same piece. Additionally, the notion of "duplicate" is ill-defined in the context of sheet music, as even two scores of the same song may be arranged differently and for distinct instrumentations.

To address these issue, we first use a pre-trained text embedding model, Sentence-BERT (22), to encode song titles as fixed dimension embedding vectors. Formally, we embed a song "descriptor" – a combination of a song's title, subtitle (if applicable), artist, and composer (if different from the artist) – to combat the scenario where different artists compose works of the same name. We then use cosine similarity to compare embedding vectors and scale those values between zero and one to obtain similarity scores. We set a duplicate threshold of 80%; that is, we cluster a given song with all other songs $\geq$80% similar. This threshold was manually verified to capture a reasonable range of duplicate songs. However, deduplicating by song descriptor alone ignores a variety of possible different arrangements, all valuable from a music generation perspective.

For each cluster of duplicate songs, we next group by instrumentation. For example, solo piano arrangements of "Pachelbel's Canon" fall into separate clusters from their string quartet counterparts.

However, within a single instrumentation cluster can exist multiple unique arrangements (e.g. a beginner versus advanced version). To address this, we use a simple heuristic of total note count to cluster songs (as note count is an easy proxy for large differences in song arrangement). We set a uniqueness threshold of 5% (i.e. songs with a >5% difference in note count fall into different clusters). Like the duplicate threshold, we obtained this value through trial and error, feeling that 5% well-captured overly similar arrangements. Within each similar note count cluster, we finally select the "best" arrangement of a song by considering each score's rating and, in the event of a tie, number of notes (the more notes, the better). Therefore, while a score could have tens of duplicates when considering song descriptor alone, within this grouping can exist many unique arrangements.

Using this deduplication strategy, we remove 151,442 songs (nearly 60% of the base dataset), leaving 102,635 unique arrangements remaining as part of the deduplicated subset. Notably, while around 85% of all songs have no duplicates, over 95% of *rated* songs are unique. Unlike rating, deduplication appears to have little effect on either harmonic or rhythmic variation.

### A.5 Dataset Subsets

Table 2: Subsets of PDMX, including filtering based on deduplication and rating information. We follow past works (3; 25; 5) and report the average Pitch Class Entropy (PCE), Scale Consistency (SC), and Groove Consistency (GC) for each subset.

| **Subset** | **Hours / Size** | **PCE** | **SC** (%) | **GC** (%) |
|---|---|---|---|---|
| **A** | 6,250 / 254K | $2.69 \pm 0.00$ | $97.12 \pm 0.01$ | $93.75 \pm 0.01$ |
| **D** | 3,756 / 102K | $2.77 \pm 0.00$ | $95.15 \pm 0.02$ | $93.79 \pm 0.01$ |
| **R** | 1,001 / 14K | $2.91 \pm 0.00$ | $91.59 \pm 0.07$ | $92.59 \pm 0.05$ |
| **R∩D** | 941 / 13K | $2.90 \pm 0.00$ | $91.70 \pm 0.07$ | $92.65 \pm 0.05$ |
| Fine-Tuning | 595 / 6K | $2.95 \pm 0.00$ | $90.54 \pm 0.11$ | $92.39 \pm 0.07$ |

### A.6 Analysis

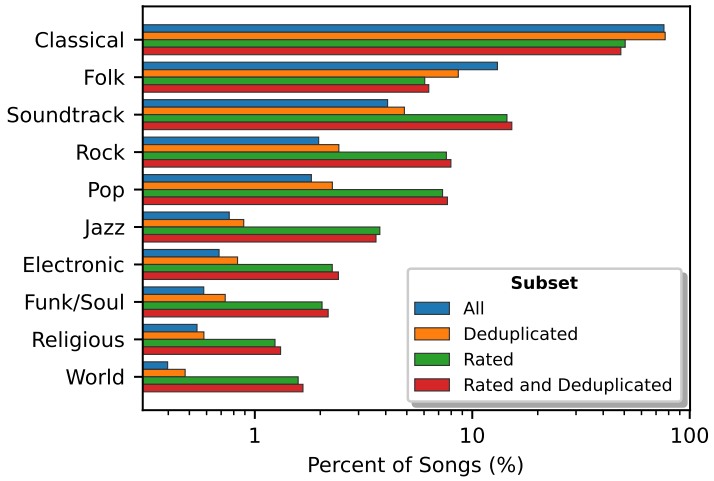


Figure 3: Top-10 Genre distribution in PDMX. 67% of songs lack a genre tag. All genres besides the two most common, classical and folk music, display notably higher frequencies in the rated subsets.

As PDMX is a *multitrack* dataset, we first analyze the density of tracks (i.e. single instrument parts) within a given song. Over 90% of songs in PDMX contain less than five tracks, while over half the dataset consists of solo works. We note that only ≈3% of songs in PDMX have more than five tracks, possibly due to larger multitrack scores (common in orchestral or marching band repertoires) not in the public domain. PDMX encompasses 20 different genres, the most common being classical and folk music (see Fig. 3). More modern genres, like hip-hop and electronic music, are comparatively

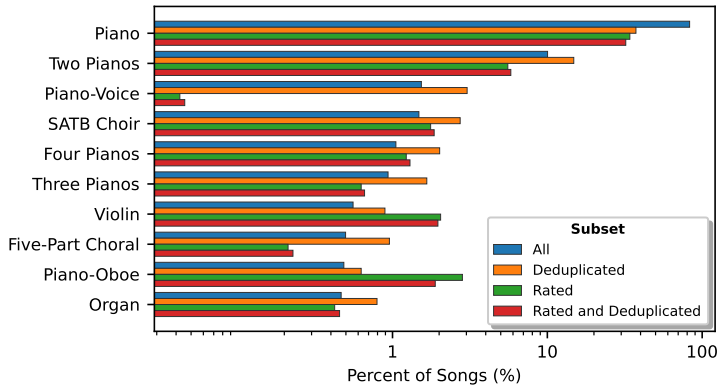


Figure 4: Top-10 most common instrumentations in PDMX. Piano arrangements dominate the dataset, while Piano-Voice arrangements are markedly less frequent in the rated subsets.

much fewer in number, likely due to limited public domain content for more recent works. Although genre tags are absent from 67% of songs, only 50% of *notes* lack a genre, suggesting that genre-labeled works are more dense than unlabelled ones. Additionally, when breaking down the genre distribution by subset (see Fig. 3), we find that the rated (and subsequently rated and deduplicated) subsets of PDMX contain a significantly longer "tail" of genres than the full dataset (with ≈40% coming from non classical/folk genres), denoting a large amount of unrated *classical* music present in PDMX. We also analyze the instrumentation of PDMX (see Fig. 4) and observe that the dataset includes >10K unique instrumentations. Despite a substantial frequency of piano music, PDMX also contains a notable amount of works for choir and string sections.

### A.7 Additional Features

While we have focused on highlighting PDMX's note-based content and parts of its metadata, we note that PDMX includes a number of extra features. Regarding metadata, PDMX includes rich rating and user comment information, enabling work on symbolic music recommendation (27; 28) and preference modeling. Notably, as PDMX is a MusicXML dataset, it includes 12M time-aligned performance directives such as tempo text (e.g. *adagio*), dynamic hairpins (e.g. *crescendo*), note articulations (e.g. *staccato*), and section markings, which are parsed by the *MusicRender* framework and may be used both for discriminative tagging tasks, expressive music rendering, and even as conditions for controllable music generation (5; 29). PDMX also includes over 10M lyric tokens, opening the door for further research on lyric-to-score and score-to-lyric tasks (30; 31; 32). We plan to fully release PDMX as 44.1kHz synthetic audio rendered through Fluidsynth, creating a large corpus for music transcription (33) and audio-domain music generation (6; 29).

### A.8 Experimental Setup and Metrics

In this work, we view each score in PDMX as a sequence of notes $\boldsymbol{n} = \boldsymbol{n}_1, \ldots, \boldsymbol{n}_N$, from which our goal is to learn an autoregressive model $p_\theta(\boldsymbol{n}_i \mid \boldsymbol{n}_{1:i-1})$ that can generate symbolic music. To accomplish this, we use the tokenization scheme REMI+ (34), an extension to the REMI input representation that allows for multitrack music (35). Each note is represented by a tuple of five variables: beat, position, pitch, duration, instrument. We employ a metrical timing system (i.e. time is represented in beats rather than seconds).

For all experiments, we use a REMI+ style decoder-only transformer, with 6 layers, 8 attention heads, and a hidden dimension of 512, totaling at ≈20M parameters. We use an absolute positional embedding with a maximum sequence length of 1024. We train each model on a single A6000 GPU for 100K steps with a batch size of 12, learning rate of 5e-4, and the Adam optimizer with default hyperparameters. For fine-tuning, we employ a smaller learning rate of 5e-5, and only train for 5K steps.

Following past work (3; 25; 5), we report the PCE, SC, and GC across 1,200 generations per model, which measure how well the model captures the underlying musical patterns of the data.

### A.9 Objective Metrics

Table 3: Quantitative Results. For all metrics, performance is determined by closeness to the fine-tuning subset's statistics.

| Subset | Fine-Tuned | PCE | SC (%) | GC (%) |
|---|---|---|---|---|
| **A** | | $2.66 \pm 0.01$ | $97.65 \pm 0.13$ | $93.70 \pm 0.11$ |
| **D** | | $2.66 \pm 0.01$ | $96.32 \pm 0.16$ | $93.83 \pm 0.15$ |
| **R** | | $\mathbf{2.80 \pm 0.01}$ | $\underline{93.89 \pm 0.20}$ | $\underline{91.83 \pm 0.22}$ |
| **R∩D** | | $\underline{2.80 \pm 0.01}$ | $\mathbf{93.32 \pm 0.21}$ | $\mathbf{91.90 \pm 0.21}$ |
| **Random** | | $2.68 \pm 0.01$ | $97.57 \pm 0.12$ | $93.46 \pm 0.12$ |
| **A** | ✓ | $2.81 \pm 0.01$ | $\mathbf{92.42 \pm 0.23}$ | $90.75 \pm 0.25$ |
| **D** | ✓ | $2.79 \pm 0.01$ | $93.17 \pm 0.22$ | $\mathbf{90.93 \pm 0.24}$ |
| **R** | ✓ | $\underline{2.84 \pm 0.01}$ | $92.77 \pm 0.22$ | $90.68 \pm 0.24$ |
| **R∩D** | ✓ | $\mathbf{2.85 \pm 0.01}$ | $\underline{92.51 \pm 0.22}$ | $90.38 \pm 0.26$ |
| **Random** | ✓ | $2.79 \pm 0.01$ | $93.37 \pm 0.21$ | $\underline{90.83 \pm 0.27}$ |

In Tab. 3, we observe that between the five base models, those trained on the rated subsets display greater harmonic and rhythmic diversity than those not, and have the closest statistics to the fine-tuning subset. However, once all models fine-tuned on the $50\%$ best rated scores, this distinction goes away, with all five fine-tuned models showing more similar metrics.

### A.10 Subjective Listening Test

To measure music quality across our ten models (five base, five fine-tuned), we conducted a listening test with 12 participants. In the questionnaire, each participant listened to 30 different samples randomly chosen from a pool of 10 samples per model. Following past work (36), for each sample users were asked to rate the generation (from 0-100) along three axes:

- **Correctness:** Is the music free of inharmonious notes, unnatural rhythms, and awkward phrasing?
- **Richness:** Is the sample musically / harmonically interesting?
- **Quality:** Subjectively, how much do you like the generation?

In Fig. 1, we show the average values (top) and violin plots (bottom) for each model along each rating axis. In comparing different modeling subsets (ignoring fine-tuning), we find that correctness is consistent across all subsets. For richness and quality however, the **R∩D** subset (i.e. our most filtered subset, in red) consistently shows the highest scores, followed by **R** (green), **A** (blue), **Random** (purple), and **D** (orange), suggesting that our deduplication strategy is primarily useful for rated songs (as we pick the highest rated duplicate) rather than unrated (which is chosen at random).

Regarding fine-tuning, we find that this process increases richness in *all* models, and improves quality in three (**D**, **R∩D**, **Random**), with a particularly strong effect on the Random (purple) model, converting it into our second-highest rated in quality overall. There is also a noticeable negative effect on correctness in the All (blue) model, suggesting that the model may have overfit to overly simple examples in the full dataset. These results together show the strength of our deduplication and filtering strategies as a way to perform significant dataset distillation (as the **R∩D** model performs best despite seeing only 15% of the data).