THE NES MUSIC DATABASE: A MULTI-INSTRUMENTAL DATASET WITH EXPRESSIVE PERFORMANCE ATTRIBUTES

Chris Donahue  
UC San Diego  
cdonahue@ucsd.edu

Huanru Henry Mao  
UC San Diego  
hhmao@ucsd.edu

Julian McAuley  
UC San Diego  
jmcauley@ucsd.edu

ABSTRACT

Existing research on music generation focuses on composition, but often ignores the expressive performance characteristics required for plausible renditions of resultant pieces. In this paper, we introduce the Nintendo Entertainment System Music Database (NES-MDB), a large corpus allowing for separate examination of the tasks of composition and performance. NES-MDB contains thousands of multi-instrumental songs composed for playback by the compositionally-constrained NES audio synthesizer. For each song, the dataset contains a musical score for four instrument voices as well as expressive attributes for the dynamics and timbre of each voice. Unlike datasets comprised of General MIDI files, NES-MDB includes all of the information needed to render exact acoustic performances of the original compositions. Alongside the dataset, we provide a tool that renders generated compositions as NES-style audio by emulating the device’s audio processor. Additionally, we establish baselines for the tasks of composition, which consists of learning the semantics of composing for the NES synthesizer, and performance, which involves finding a mapping between a composition and realistic expressive attributes.

1. INTRODUCTION

The problem of automating music composition is a challenging pursuit with the potential for substantial cultural impact. While early systems were hand-crafted by musicians to encode musical rules and structure [25], recent attempts view composition as a statistical modeling problem using machine learning [3]. A major challenge to casting this problem in terms of modern machine learning methods is building representative datasets for training. So far, most datasets only contain information necessary to model the semantics of music composition, and lack details about how to translate these pieces into nuanced performances. As a result, demonstrations of machine learning systems trained on these datasets sound rigid and deadpan. The datasets that do contain expressive performance characteristics predominantly focus on solo piano [10, 27, 32] rather than multi-instrumental music.

A promising source of multi-instrumental music that contains both compositional and expressive characteristics is music from early videogames. There are nearly 1400 unique games licensed for the Nintendo Entertainment System (NES), all of which include a musical soundtrack. The technical constraints of the system’s audio processing unit (APU) impose a maximum of four simultaneously monophonic instruments. The machine code for the games preserves the exact expressive characteristics needed to perform each piece of music as intended by the composer. All of the music was composed in a limited time period and, as a result, is more stylistically cohesive than other large datasets of multi-instrumental music. Moreover, NES music is celebrated by enthusiasts who continue to listen to and compose music for the system [6], appreciating the creativity that arises from resource limitations.

In this work, we introduce NES-MDB, and formalize two primary tasks for which the dataset serves as a large test bed. The first task consists of learning the semantics of composition on a separated score, where individual instrument voices are explicitly represented. This is in contrast to the common blended score approach for modeling polyphonic music, which examines reductions of full scores. The second task consists of mapping compositions onto sets of expressive performance characteristics. Combining strategies for separated composition and expressive performance yields an effective pipeline for generating NES music de novo. We establish baseline results and reproducible evaluation methodology for both tasks. A further contribution of this work is a library that converts between NES machine code (allowing for realistic playback) and representations suitable for machine learning.1

2. BACKGROUND AND TASK DESCRIPTIONS

Statistical modeling of music seeks to learn the distribution \( P(\text{music}) \) from human compositions \( c \sim P(\text{music}) \) in a dataset \( M \). If this distribution could be estimated accurately, a new piece could be composed simply by sampling. Since the space of potential compositions is exponentially large, to make sampling tractable, one usually assumes a factorized distribution. For monophonic sequences, which consist of no more than one note at a time, the probability

1 Including games released only on the Japanese version of the console

2 https://github.com/chrisdonahue/nesmdb
2.2 Separated composition

Given the shortcomings of the blended score, we might prefer models which operate on a separated score representation (Fig. 1b). A separated score $S$ is a matrix of size $V \times T$, where $V$ is the number of instrument voices, and $S[v, t] = n$, the note $n$ played by voice $v$ at timestep $t$. In other words, the format encodes a monophonic sequence for each instrument voice. Statistical approaches to this representation can explicitly model the relationships between various instrument voices by

$$P(c) = \prod_{t=1}^{T} \prod_{v=1}^{V} P(S[v, t] \mid S_{v, t\neq t}, S_{v\neq v, \forall t}).$$

This formulation explicitly models the dependencies between $S[v, t]$, voice $v$ at time $t$, and every other note in the score. For this reason, Eq. 3 more closely resembles the process by which human composers write multi-instrumental music, incorporating temporal and contrapuntal information. Another benefit is that resultant models can be used to harmonize with existing musical material, adding voices conditioned on existing ones. However, any non-trivial amount of temporal context introduces high-dimensional interdependencies, meaning that such a formulation would be challenging to sample from. As a consequence, solutions are often restricted to only take past temporal context into account, allowing for simple and efficient ancestral sampling (though Gibbs sampling can also be used to sample from Eq. 3 [13, 16]).

Most existing datasets of multi-instrumental music have uninhibited polyphony, causing a separated score representation to be inappropriate. However, the hardware constraints of the NES APU impose a strict limit on the number of voices, making the format ideal for NES-MDB.

2.3 Expressive performance

Given a piece of a music, a skilled performer will embellish the piece with expressive characteristics, altering the timing and dynamics to deliver a compelling rendition. While a few instruments have been augmented to capture this type of information symbolically (e.g. a Disklavier), it is rarely available for examination in datasets of multi-instrumental music. Because NES music is comprised of instructions that recreate an exact rendition of each piece, expressive characteristics controlling the velocity and timbre of each voice are available in NES-MDB (details in Section 3.1). Thus, each piece can be represented as an expressive score (Fig. 1c), the union of its separated score and expressive characteristics.

We consider the task of mapping a composition $c$ onto expressive characteristics $e$. Hence, we would like to model $P(e \mid c)$, and the probability of a piece of music $P(m)$ can be expressed as $P(e \mid c) \cdot P(c)$, where $P(c)$ is from Eq. 3. This allows for a convenient pipeline for music generation where a piece of music is first composed with binary amplitudes and then mapped to realistic dynamics, as if interpreted by a performer.
3.1 Extracting expressive scores

Given the VGM files, we emulate the functionality of the APU to yield an expressive score (Fig. 1c) at a temporal discretization of 44.1 kHz. This rate is unnecessarily high for symbolic music, so we subsequently downsample the scores. Because the music has no explicit tempo markings, we accommodate a variety of implicit tempos by choosing a permissive downsampling rate of 24 Hz. By removing dynamics, timbre, and voicing at each timestep, we derive separated score (Fig. 1b) and blended score (Fig. 1a) versions of the dataset.

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Note</th>
<th>Velocity</th>
<th>Timbre</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse 1 (P1)</td>
<td>({0, 32, \ldots, 108})</td>
<td>([0, 15])</td>
<td>([0, 3])</td>
</tr>
<tr>
<td>Pulse 2 (P2)</td>
<td>({0, 32, \ldots, 108})</td>
<td>([0, 15])</td>
<td>([0, 3])</td>
</tr>
<tr>
<td>Triangle (TR)</td>
<td>({0, 21, \ldots, 108})</td>
<td>([0, 15])</td>
<td>([0, 1])</td>
</tr>
<tr>
<td>Noise (NO)</td>
<td>({0, 1, \ldots, 16})</td>
<td>([0, 15])</td>
<td>([0, 1])</td>
</tr>
</tbody>
</table>

Table 2: Dimensionality for each timestep of the expressive score representation (Fig. 1c) in NES-MDB.

In Table 2, we show the dimensionality of the instrument states at each timestep of an expressive score in NES-MDB. We constrain the frequency ranges of the melodic voices (pulse and triangle generators) to the MIDI notes on an 88-key piano keyboard (21 through 108 inclusive, though the pulse generators cannot produce pitches below MIDI note 32). The percussive noise voice has 16 possible “notes” (these do not correspond to MIDI note numbers) where higher values have more high-frequency noise. For all instruments, a note value of 0 indicates that the instrument is not sounding (and the corresponding velocity will be 0). When sounding, the pulse and noise generators have 15 non-linear velocity values, while the triangle generator has no velocity control beyond on or off.

Additionally, the pulse wave generators have 4 possible duty cycles (affecting timbre), and the noise generator has a rarely-used mode where it instead produces metallic tones. Unlike for velocity, a timbre value of 0 corresponds to an actual timbre setting and does not indicate that an instrument is muted. In total, the pulse, triangle and noise generators have state spaces of sizes 4621, 89, and 481 respectively—around 40 bits of information per timestep for the full ensemble.

4. EXPERIMENTS AND DISCUSSION

Below, we describe our evaluation criteria for experiments in separated composition and expressive performance. We present these results only as statistical baselines for comparison; results do not necessarily reflect a model’s ability to generate compelling musical examples.

**Negative log-likelihood and Accuracy** Negative log-likelihood (NLL) is the (log of the) likelihood that a model assigns to unseen real data (as per Eq. 3). A low NLL averaged across unseen data may indicate that a model captures

---

5. We also release NES-MDB in MIDI format with no downsampling
semantics of the data distribution. Accuracy is defined as the proportion of timesteps where a model’s prediction is equal to the actual composition. We report both measures for each voice, as well as aggregations across all voices by summing (for NLL) and averaging (for accuracy).

**Points of Interest (POI).** Unlike other datasets of symbolic music, NES-MDB is temporally-discretized at a high, fixed rate (24 Hz), rather than at a variable rate depending on the tempo of the music. As a consequence, any given voice has around an 83% chance of playing the same note as that voice at the previous timestep. Accordingly, our primary evaluation criteria focuses on musically-salient points of interest (POIs), timesteps at which a voice deviates from the previous timestep (the beginning or end of a note). This evaluation criterion is mostly invariant to the rate of temporal discretization.

### 4.1 Separated composition experiments

For separated composition, we evaluate the performance of several baselines and compare them to a cutting edge method. Our simplest baselines are unigram and additive-smoothed bigram distributions for each instrument. The predictions of such models are trivial: the unigram model always predicts “no note” and the bigram model always predicts “last note”. The respective accuracy of these models, 37% and 83%, reflect the proportion of the timesteps that are silent (unigram) or identical to the last timestep (bigram). However, if we evaluate these models only at POIs, their performance is substantially worse (4% and 0%).

We also measure performance of recurrent neural networks (RNNs) at modeling the voices independently. We train a separate RNN (either a basic RNN cell or an LSTM cell [15]) on each voice to form our RNN Soloists and LSTM Soloists baselines. We compare these to LSTM Quartet, a model consisting of a single LSTM that processes all four voices and outputs an independent softmax over each note category, giving the model full context of the composition in progress. All RNNs have 2 layers and 256 units, except for soloists which have 64 units each, and we train them with 512 steps of unrolling for backpropagation through time. We train all models to minimize NLL using the Adam optimizer [19] and employ early stopping based on the NLL of the validation set.

While the DeepBach model [13] was designed for modeling the chorales of J.S. Bach, the four-voice structure of those chorales is shared by NES-MDB, making the model appropriate for evaluation in our setting. DeepBach embeds each timestep of the four-voice score and then processes these embeddings with a bidirectional LSTM to aggregate past and future musical context. For each voice, the activations of the bidirectional LSTM are concatenated with an embedding of all of the other voices, providing the model with a mechanism to alter its predictions for any voice in context of the others at that timestep. Finally, these merged representations are concatenated to an independent softmax for each of the four voices. Results for DeepBach and our baselines appear in Table 3.

As expected, the performance of all models at POIs is worse than the global performance. DeepBach achieves substantially better performance at POIs than the other models, likely due to its bidirectional processing which allows the model to “peek” at future notes. The LSTM Quartet model is attractive because, unlike DeepBach, it permits efficient ancestral sampling. However, we observe qualitatively that samples from this model are musically unsatisfying. While the performance of the soloists is worse than the models which examine all voices, the superior performance of the LSTM Soloists to the RNN Soloists suggests that LSTMs may be beneficial in this context.

We also experimented with artificially emphasizing POIs during training, however we found that resultant models produced unrealistically sporadic music. Based on this observation, we recommend that researchers who study NES-MDB always train models with unbiased emphasis, in order to effectively capture the semantics of the particular temporal discretization.

### 4.2 Expressive performance experiments

The expressive performance task consists of learning a mapping from a separated score to suitable expressive characteristics. Each timestep of a separated score in NES-MDB has note information (random variable N) for the four instrument voices. An expressive score additionally has velocity (V) and timbre (T) information for P1, P2, and NO but not TR. We can express the distribution of performance characteristics given the composition as $P(V, T | N)$. Some of our proposed solutions factorize this further into a conditional autoregressive formulation $\prod_{t=1}^{T} P(V_t, T_t | N, V_{t<1}, T_{t<1})$, where the model has explicit knowledge of its decisions for velocity and timbre at earlier timesteps.

![Figure 2: LSTM Note+Auto expressive performance model that observes both the score and its prior output.](image)

Unlike for separated composition, there are no well-established baselines for multi-instrumental expressive performance, and thus we design several approaches. For the autoregressive formulation, our most-sophisticated model (Fig. 2) uses a bidirectional LSTM to process the separated score, and a forward-directional LSTM for the autoregressive expressive characteristics. The represen-
Table 3: Results for separated composition experiments. For each instrument, negative log-likelihood and accuracy are calculated at points of interest (POIs). We also calculate aggregate statistics at POIs and globally (All). While DeepBach [13] achieves the best statistical performance, it uses future context and hence is more expensive to sample from.

Table 4: Results for expressive performance experiments evaluated at points of interest (POI). Results are broken down by expression category (e.g., $V_{NO}$ is noise velocity, $T_P$ is pulse 1 timbre) and aggregated at POIs and globally (All).

Table 5: Negative log-likelihoods for various models on the blended score format (Fig. 1a, Eq. 2) of NES-MDB. We also show results for Piano-midi.de (PM), Nottingham (NH), MuseData (MD), and the chorales of J.S. Bach (BC).

what value to output next. Interestingly, the MultiReg Note model has better performance at POIs than the MultiReg Note+Auto model. The latter overfits more quickly which may explain its inferior performance despite the fact that it sees strictly more information than the note-only model.

4.3 Blended composition experiments

In Table 5, we report the performance of several models on the blended composition task (Eq. 2). In NES-MDB, blended scores consist of 88 possible notes with a maximum of three simultaneous voices (noise generator is discarded). This task, standardized in [2], does not preserve the voicing of the score, and thus it is not immediately useful for generating NES music. Nevertheless, modeling blended scores of polyphonic music has become a standard benchmark for sequential models [5, 18], and NES-MDB
may be useful as a larger dataset in the same format.

In general, models assign higher likelihood to NES-MDB than the four other datasets after training. As with our other two tasks, this is likely due to the fact that NES-MDB is sampled at a higher temporal rate, and thus the average deviation across timesteps is lower. Due to its larger size, a benefit of examining NES-MDB in this context is that sequential models tend to take longer to overfit the dataset than they do for the other four. We note that our implementations of these models may deviate slightly from those of the original authors, though our models achieve comparable results to those reported in [2,17] when trained on the original datasets.

5. RELATED WORK

There are several popular datasets commonly used in statistical music composition. A dataset consisting of the entirety of J.S. Bach’s four-voice chorales has been extensively studied under the lenses of algorithmic composition and reharmonization [1, 2, 13, 14]. Like NES-MDB, this dataset has a fixed number of voices and can be represented as a separated score (Fig. 1b), however it is small in size (389 chorales) and lacks expressive information. Another popular dataset is Piano-midi.de, a corpus of classical piano from various composers [27]. This dataset has expressive timing and dynamics information but has heterogeneous time periods and only features solo piano music. Alongside Bach’s chorales and the Piano-midi.de dataset, Boulanger-Lewandowski et al. (2012) standardized the Nottingham collection of folk tunes and MuseData library of orchestral and piano classical music into blended score format (Fig. 1a).

Several other symbolic datasets exist containing both compositional and expressive characteristics. The Magaloff Corpus [10] consists of Disklavier recordings of a professional pianist playing the entirety of Chopin’s solo piano works. The Lakh MIDI dataset [28] is the largest corpus of symbolic music assembled to date with nearly 200k songs. While substantially larger than NES-MDB, the dataset has unconstrained polyphony, inconsistent expressive characteristics, and encompasses a wide variety of genres, instruments and time periods. Another paper trains neural networks on transcriptions of video game music [9], though their dataset only includes a handful of songs.

5.1 Statistical composition

While most of the early research in algorithmic music composition focused on expert systems [25], statistical approaches have since become the predominant approach. Mozer (1994) trained RNNs on monophonic melodies using a formulation similar to Eq. 1, finding the composed results to compare favorably to those from a trigram model. Others have also explored monophonic melody generation with RNNs [8, 26]. Boulanger-Lewandowski et al. (2012) standardize the polyphonic prediction task for blended scores (Eq. 2), measuring the polyphonic prediction task for a multitude of classical baselines against RNNs [30], restricted Boltzmann machines [34], and NADEs [21] on polyphonic music datasets. Several papers [5, 17, 35] directly compare to their results. Statistical models of music have also been employed as symbolic priors to assist music transcription algorithms [2, 4, 24].

Progressing towards models that assist humans in composition, many researchers study models to create new harmonizations for existing musical material. Allan and Williams (2005) train HMMs to create new harmonizations for Bach chorales [1]. Hadjeres et al. (2017) train a bidirectional RNN model to consider past and future temporal context (Eq. 3) [13]. Along with [16, 31], they advocate for the usage of Gibbs sampling to generate music from complex graphical models.

5.2 Statistical performance

Musicians perform music expressively by interpreting a performance with appropriate dynamics, timing and articulation. Computational models of expressive music performance seek to automatically assign such attributes to a score [36]. We point to several extensive surveys for information about the long history of rule-based systems [7, 12, 20, 36].

Several statistical models of expressive performance have also been proposed. Raphael (2010) learns a graphical model that automates an accompanying orchestra for a soloist, operating on acoustic features rather than symbolic [29]. Flossmann et al. (2013) build a system to control velocity, articulation and timing of piano performances by learning a graphical model from a large symbolic corpus of human performances [11]. Xia et al. (2015) model the expressive timing and dynamics of piano duet performances using spectral methods [37]. Two end-to-end systems attempt to jointly learn the semantics of composition and expressive performance using RNNs [23, 33]. Malik and Ek (2017) train an RNN to generate velocity information given a musical score [22]. These approaches differ from our own in that they focus on piano performances rather than multi-instrumental music.

6. CONCLUSION

The NES Music Database is a large corpus for examining multi-instrumental polyphonic composition and expressive performance generation. Compared to existing datasets, NES-MDB allows for examination of the “full pipeline” of music composition and performance. We parse the machine code of NES music into familiar formats (e.g. MIDI), eliminating the need for researchers to understand low-level details of the game system. We also provide an open-source tool which converts between the simpler formats and machine code, allowing researchers to audit their generated results as waveforms rendered by the NES. We hope that this dataset will facilitate a new paradigm of research on music generation—one that emphasizes the importance of expressive performance. To this end, we establish several baselines with reproducible evaluation methodology to encourage further investigation.
7. ACKNOWLEDGEMENTS

We would like to thank Louis Pisha for invaluable advice on the technical details of this project. Additionally, we would like to thank Nicolas Boulanger-Lewandowski, Eunjung Stella Koh, Steven Merity, Miller Puckette, and Cheng-i Wang for helpful conversations throughout this work. This work was supported by UC San Diego’s Chancellor’s Research Excellence Scholarship program. GPUs used in this research were donated by NVIDIA.

8. REFERENCES


