THE JAZZ TRANSFORMER ON THE FRONT LINE: EXPLORING THE SHORTCOMINGS OF AI-COMPOSED MUSIC THROUGH QUANTITATIVE MEASURES

Shih-Lun Wu[1,2] **and Yi-Hsuan Yang**[2,3]
[1] National Taiwan University, [2] Taiwan AI Labs, [3] Academia Sinica, Taiwan
`b06902080@csie.ntu.edu.tw, yang@citi.sinica.edu.tw`

ABSTRACT

This paper presents the Jazz Transformer, a generative model that utilizes a neural sequence model called the Transformer-XL for modeling lead sheets of Jazz music. Moreover, the model endeavors to incorporate structural events present in the Weimar Jazz Database (WJazzD) for inducing structures in the generated music. While we are able to reduce the training loss to a low value, our listening test suggests however a clear gap between the ratings of the generated and real compositions. We therefore go one step further and conduct a series of computational analysis of the generated compositions from different perspectives. This includes analyzing the statistics of the pitch class, grooving, and chord progression, assessing the structureness of the music with the help of the fitness scape plot, and evaluating the model's understanding of Jazz music through a MIREX-like continuation prediction task. Our work presents in an analytical manner why machine-generated music to date still falls short of the artwork of humanity, and sets some goals for future work on automatic composition to further pursue.

1. INTRODUCTION

Music is a heart-touching form of art that strikes a chord with people's emotions, joyful or sorrowful; intense or relieved, through the twists and turns of notes. Despite its ubiquity in our everyday lives, the composition and arrangement of music often requires substantial human effort. This is a major reason why automatic music composition is such a fascinating field of study. Over the years, researchers have sought strenuously ways for machines to generate well-formed music; such methods include meticulously designed non deep learning-based algorithms like the Markov chains [6] and formal grammars [19]; and, a proliferation of deep learning-based solutions in the past decade [8]. In this work, we exclusively study the extension and evaluation of Transformer-based models [43] for its claimed successes in natural language processing and music generation in recent years [12, 13, 20, 38].

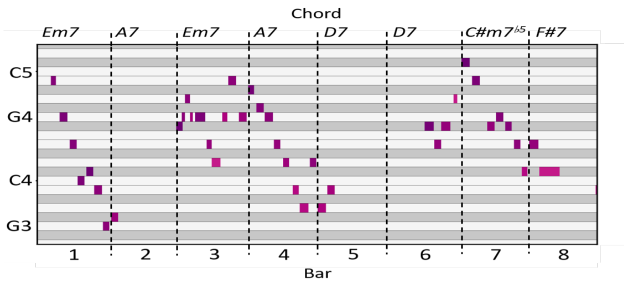

Figure 1. The first 8 bars of a piece (filename `sample_B01.mp3` in Google Drive) composed by the Jazz Transformer, exhibiting clear rests between phrases.

The dataset chosen for our work is the Weimar Jazz Database (WJazzD) [2, 37]. As opposed to the commonly used piano MIDIs in recent works [20, 21], the choice of this dataset represents a fresh endeavor to train the Transformer on Jazz music, and grants us the unique opportunity to integrate structure-related events, precisely annotated in the WJazzD, to the model. However, such an attempt involves no short of technical challenges, including the quantization of the numerous short notes in Jazz improvisations; and, dealing with the complex chord representations used in the WJazzD. In Section 3, we will elaborate on how these difficulties are tackled in a detailed manner.

Furthermore, while recent works in Transformer-based music generation often praised the model's capabilities, like being able to compose "compelling" music, or generate pieces with "expressiveness, coherence, and clear structures" as claimed in [20] and [21] respectively, rarely do we admit that the machine is still far behind humans, as shown in our user study (Section 4), and take a step back to "face the music", in other words, to identify what exactly goes wrong in the model's compositions.

Therefore, the goal of the paper is two-fold. First, to deploy Transformers to a new, more complex music genre, Jazz, asking the model to compose melody lines, chord progression, and structures all at once. Second, to develop a set of objective metrics (Section 5) that evaluate the generated music's pitch usages, rhythmicity, consistency in chord progression, and structureness (see Sec. 5.4 for definition), to discover the culprits behind the model's incompetence (Section 6).

Figure 1 shows an example of a composition generated by our model, in which we may find reasonable combinations of chords and melody; and, clear rests between phrases. Audio samples can be found in a Google Drive folder, [1] which we encourage readers to listen to. We have also open-sourced our implementation of the Jazz Transformer [2] and the proposed objective metrics. [3]

2. RELATED WORK

There has been a great body of research work on computational analysis of human performance of Jazz [3, 4, 15, 18, 44]. One prominent example is the Jazzomat Project [5], which established the WJazzD [2] to study the creative processes underpinning Jazz solo improvisations [37]. Weiss *et al.* [44], for instance, used the dataset to explore the evolution of tonal complexity of Jazz improvisations in the past century. See Sec. 3.1 for more details of the dataset.

The use of Transformer-like architectures for training music composition models has drawn increasing attention recently. These works enhanced the Transformer's capability in modeling music through relative positional encoding schemes [20, 36], cross-domain pre-training [13], and event token design [13, 21]. To the best of our knowledge, this work represents the first attempt in the literature to employ Transformers to compose exclusively Jazz music.

Automatic composition of general lead sheets has been investigated lately, mostly based on recurrent neural network (RNN) models [7, 29, 30]. As for inducing structures in the generated music, several RNN-based solutions have also been proposed [24, 31, 39]. Since Transformers have been shown to outperform RNNs in various tasks [9, 20, 26], we strive to be the forerunner in bringing them to these realms of research.

Relatively little work has been done to train a model for Jazz composition. JazzGAN [42] is a model employing a generative adversarial network (GAN) architecture for chord-conditioned melody composition, using a dataset of only 44 lead sheets, approximately 1,700 bars. Another model presented in [22] explores the use of recurrent variational auto-encoders for generating both the melody and chords of a lead sheet from scratch.

A number of objective metrics have been employed for measuring the performance of deep learning for music composition [10, 14, 41, 45]. However, most of them focused on surface-level statistics only (e.g., pitch class histograms, note onset intervals, etc.). The introduction of structureness indicators and the MIREX-like metric (see Sec. 5.4–5.5) in this paper provide new insights into assessing music's quality at piece level, and evaluating the model's overall understanding of a certain music genre.

3. THE JAZZ TRANSFORMER

Transformers use self-attention modules to aggregate information from the past events when predicting the next events [11, 27, 43]. Accordingly, it is natural that we model music as a language, namely, to represent each composition by a sequence of event tokens. In this section, we will explain in detail how we break down the components of the WJazzD [37] to construct the vocabulary of events, and how the pieces are converted into sequences that can be fed into a Transformer-like model for training.

	# solos	Total duration	Total # events	Avg. # events per solo
Train	409	11h 19m	1,220 K	2,983
Val.	22	33m	56 K	2,548

Table 1. Statistics of the dataset we compile from the WJazzD [37]. See Section 3.2 for details of the "events".

3.1 Dataset

The WJazzD dataset [2, 37] comprises of 456 monophonic Jazz solos. Each solo is arranged in the lead sheet style and comes with two tracks: the melody track and the beat track. The melody track contains every note's pitch, onset time and duration (in seconds), with additional information on loudness (in decibels), phrase IDs and "midlevel units" (MLUs) [17], a structure of finer granularity than a phrase to capture the distinctive short-time ideas in Jazz improvisations. The beat track contains the beat onsets (in seconds), chord progressions and form parts (or sections, e.g., `A1`, `B1`). The highlight of this dataset is that all the contents, including the notes, chords, metrical and structural markings, are human-annotated and cross-checked by the annotators [37], ensuring the data cleanliness that is often crucial for machine learning tasks. To simplify the subsequent processings, we retain only the pieces marked solely with 4/4 time signature, resulting in 431 solos. For objective analysis, we leave 5% of the solos as the held-out validation data. See Table 1 for the statistics of the data.

3.2 Data Representation

The event representation adopted here is a modified version of the "REvamped MIDI-derived event representation" recently proposed in [21], extended to integrate the chord system and structural events of WJazzD. The resulting event encodings can be broken down into the following 4 categories: **note-related**—NOTE-VELOCITY, NOTE-ON, NOTE-DURATION; **metric-related**—BAR, POSITION, TEMPO-CLASS, TEMPO; **chord-related**—CHORD-TONE, CHORD-TYPE, CHORD-SLASH; and **structure-related**—PHRASE, MLU, PART, REPETITION.

3.2.1 Note-related Events

Each note in the melody is represented by three events, i.e., NOTE-VELOCITY, NOTE-ON, and NOTE-DURATION.

The NOTE-VELOCITY event decides how hard the note should be played. We derive it according to the estimated loudness (in decibels) provided by the dataset, and quantize it into 32 bins, corresponding to MIDI velocities

[1] `https://drive.google.com/drive/folders/1-09SoxumYPdYetsUWHIHSugK99E2tNYD?usp=sharing`
[2] `https://github.com/slSeanWU/jazz_transformer`
[3] `https://github.com/slSeanWU/MusDr`

$[3, 7, \ldots, 127]$, through $v = \lfloor (80 + 3 \cdot (dB - 65))/4 \rfloor$, where dB is the decibel value of the note, and v, clipped such that $v \in [1, 32]$, is the resulting NOTE-VELOCITY(v) event. This mapping scheme comes in handy in the process of converting the model's compositions to MIDIs.

The NOTE-ON events, ranging from 0 to 127, correspond directly to the MIDI numbers, indicating the note's pitch. The NOTE-DURATION events represent the note's length in 64th note multiples, ranging from 1 to 32, obtained by taking the ratio of the note's duration (in seconds) to the duration of the beat (also in seconds) where the note situates. The reason why we use such a fine-grained quantum, while previous work mainly consider only 16th note multiples (e.g., [20, 21]), is as follows. Most notes in WJazzD are quite short, with a significant portion being 32th and 64th notes (12.9% and 2.7% respectively). The quantum is chosen such that the coverage of the 32 NOTE-DURATION events encompasses the most notes, which is 99.6% with our choice of the 64th note. [4]

3.2.2 *Metric-related Events*

To model the progression of time, we use a combination of BAR and POSITION events; as demonstrated in [21], this combination leads to clearer rhythmic structure in the generated music compared to using TIME-SHIFT events introduced in [20]. In addition, the pace the music should be played at is set by TEMPO-CLASS and TEMPO events.

A BAR event is added at the beginning of each bar, and a bar is quantized into 64 subunits, each represented by a POSITION event; for example, POSITION(16) marks the start of the 2nd beat in a bar. A POSITION event occurs whenever there is a note onset, chord change, or tempo change. It is worth mentioning that to minimize the quantization error, a note's onset position is justified with the beat it is in through the formula:

$$p_n = p_b + 16 \cdot (t_n - t_b)/d_b \,, \qquad (1)$$

where p_b, t_b, d_b are the beat's position (note that $p_b \in \{0, 16, 32, 48\}$), onset time, and duration; and t_n is the note's onset time. The resulting p_n is then rounded to the nearest integer to determine the note's onset position.

The TEMPO-CLASS and TEMPO events always co-occur at every beat position. The 5 TEMPO-CLASS events represent the general "feeling" of speed (i.e. fast, or slow) with interval boundaries of $[50, 80, 110, 140, 180, 320]$ beats per minute (bpm), while the 12 TEMPO events assigned to each tempo class in evenly-spaced steps (within the interval, e.g., $50, 52.5, 55$ bpm...) determine the exact pace. The events can be derived simply by taking the reciprocal of a beat's duration (provided by WJazzD). The frequent appearance of these tempo events facilitates smooth local tempo changes common in Jazz performances.

3.2.3 *Chord-related Events*

Chord progressions serve as the harmonic foundation of Jazz improvisations [25], hence a complex chord representation system is used in the WJazzD dataset. If we were to treat each of the 418 unique chord representations present in the WJazzD as a token, the majority of chord tokens will have very few occurrences—in fact, 287 (69%) of them appear in less than 5 solos, making it hard for the model to learn the meaning of those chords well; plus, the process of translating chords to individual notes during the conversion to MIDIs would be extremely cumbersome.

Fortunately, thanks to the detailed clarification provided in [37], we are able to decompose each chord into 3 events, namely, the CHORD-TONE, CHORD-TYPE, and CHORD-SLASH events, with the help of regular expressions (regex) and some rule-based exception handling.

The 12 CHORD-TONE events, one for each note on the chromatic scale (i.e. `C`, `C#`, `D`, ...), determine the root note, hence the tonality, of the chord. The 47 CHORD-TYPE events affect the chord's quality and emotion by the different combination of notes played relative to the root note (or, *key template* as we call it); e.g., the key template of a Dominant 7th chord (CHORD-TYPE(7)) is $[0, 4, 7, 10]$. Finally, the 12 CHORD-SLASH events allow the freedom to alter the bass note to slightly tweak the chord's quality. If a chord contains no slash, its CHORD-SLASH event will share the same key as its CHORD-TONE. For instance, the chord `C7/G`, a C Dominant 7th over G, is represented by [CHORD-TONE(C), CHORD-TYPE(7), CHORD-SLASH(G)].

Note that after our decomposition, the number of unique chord-related events is greatly reduced to 71; and, the resulting set of events is still able to represent all 418 chords in WJazzD. It is easy to use the manually-constructed key template accompanying each CHORD-TYPE, together with the CHORD-TONE and CHORD-SLASH events to map each chord to notes during the conversion to MIDIs.

3.2.4 *Structure-related Events*

For the melodies, we prepend a PHRASE event to the notes marked as the start of a phrase. The presence of phrases may be important as it informs the model to "take a breath" between streams of notes. And, we retain several common types and subtypes of midlevel units (e.g., *line, rhythm, lick* etc.) as MLU events [17], likewise prepended to the starting note of each MLU, hoping that the model could capture the short-term note patterns described by the MLU types. PART and REPETITION events are added to each beginning and end of a form part, [5] guiding the model to generate repetitive chord progression and coherent melody lines for the parts marked with the same letter.

3.3 Model and Training Setups

Due to the large number of events per solo (check Table 1), it is hard to feed the entire pieces into a Transformer at once because of memory constraint. Therefore, we choose as the backbone sequence model the Transformer-XL [11], an improved variant of the Transformer which introduces recurrence to the architecture. It remedies the memory constraint and the resulting context fragmentation issue by caching the computation record of the last segment, and

[4] All notes shorter than a 64th note are discarded and those longer than a half note are clipped.

[5] For example, the entire `A1` part is represented as [PART-START(A), REPETITION-START(1), ... other events ..., REPETITION-END(1), PART-END(A)].

allowing the current segment to attend to the cache in the self-attention process. This allows information to flow across the otherwise separated segments, inducing better coherence in the generated music.

To evaluate the effectiveness of adding the structure-related events (cf. Section 3.2.4), we consider the following two variants in our objective analysis:

- **Model (A)**: trained with no structure-related events.
- **Model (B)**: trained with the complete set of events.

They both consist of 12 layers, 8 attention heads and about 41 million learnable parameters. We train them on a single NVIDIA GTX 1080-Ti GPU (with 11 GB memory) with Adam optimizer, learning rate 1e$-$4, batch size 8, segment length 512 and 100% teacher forcing. Besides, following the Music Transformer's data augmentation setting [20], we randomly transpose each solo in the range of -3 to $+3$ keys in every epoch. It takes roughly a full day for the negative log-likelihood losses of the models to drop to 0.25, a level at which they are able to produce music of distinctive Jazz feeling (see Section 6 for justifications).

4. SUBJECTIVE STUDY

To discover how users feel about the Jazz Transformer's compositions, we set up a blind listening test in which test-takers listen to four one-minute long pieces, two from the Model (B)'s compositions (at loss level 0.25), and two from real data. We do not include Model (A) here to reduce the burden on the test-takers, assuming that Model (B) is better. We inform them that the pieces are independent of one another, and they will be asked the same set of questions after listening to each piece, namely, to rate it in a five-point Likert scale on the following aspects:

- **Overall Quality (O):** Does it sound good overall?
- **Impression (I):** Can you remember a certain part or the melody?
- **Structureness (S):** Does it involve recurring music ideas, clear phrases, and coherent sections?
- **Richness (R):** Is the music diverse and interesting?

We distribute five test suites to our social circles and collect responses from 59 anonymized subjects, of which 27 are classified as "pros" for they rate their musical background (in general, not restricted to Jazz) as $4/5$ or $5/5$ (i.e., also on a five-point scale). The result shown in Figure 2 indicates that the Jazz Transformer receives mediocre scores and falls short of humans in every aspect, especially in *overall quality* **(O)** and *structureness* **(S)**. Moreover, performed *one-tailed Z-tests for the difference of means* also suggests the significance of the gaps ($p < 0.05$ for all aspects), providing concrete evidence of the model's defeat.

5. OBJECTIVE EVALUATION METRICS

The result of our subjective study poses to us an intriguing question: If the machine is still inferior to humans in creating music, then what exactly are the causes? To unravel the mystery, we develop a set of objective metrics which enables us to scrutinize the Jazz Transformer's compositions from various perspectives, and make comparisons with real data. These metrics include the analyses of event distributions, namely, the pitch class histogram, the grooving pattern, and the chord progressions; assessing the structureness with the help of the fitness scape plot; and, judging the model's performance on a discriminative task through the MIREX-like continuation prediction challenge.

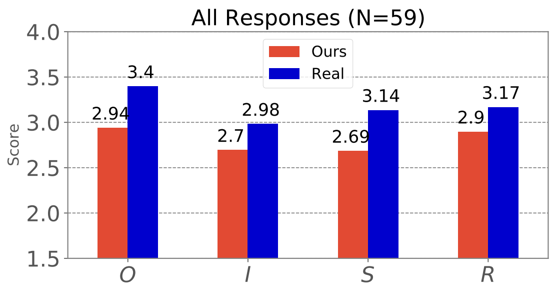

Figure 2. Result of subjective study (**O:** Overall Quality, **I:** Impression, **S:** Structureness, **R:** Richness), comparing from-scratch compositions created by the proposed model with structure-related events (i.e., 'Model (B)') against the real pieces from the WJazzD. We note that the gaps in all aspects are statistically significant ($p < 0.05$).

5.1 Pitch Class Histogram Entropy

To gain insight into the usage of different pitches, we first collect the notes appeared in a certain period (e.g., a bar) and construct the 12-dimensional pitch class histogram $\overrightarrow{\mathbf{h}}$, according to the notes' pitch classes (i.e. `C`, `C#`, ..., `A#`, `B`), normalized by the total note count in the period such that $\sum_i h_i = 1$. Then, we calculate the entropy of $\overrightarrow{\mathbf{h}}$:

$$\mathcal{H}(\overrightarrow{\mathbf{h}}) = -\sum_{i=0}^{11} h_i \log_2(h_i)\,. \tag{2}$$

The entropy, in information theory, is a measure of "uncertainty" of a probability distribution [40], hence we adopt it here as a metric to help assessing the music's quality in tonality. If a piece's tonality is clear, several pitch classes should dominate the pitch histogram (e.g., the tonic and the dominant), resulting in a low-entropy $\overrightarrow{\mathbf{h}}$; on the contrary, if the tonality is unstable, the usage of pitch classes is likely scattered, giving rise to an $\overrightarrow{\mathbf{h}}$ with high entropy.

5.2 Grooving Pattern Similarity

The grooving pattern represents the positions in a bar at which there is at least a note onset, denoted by $\overrightarrow{\mathbf{g}}$, a 64-dimensional binary vector in our setting.[6] We define the similarity between a pair of grooving patterns $\overrightarrow{\mathbf{g}}^a$, $\overrightarrow{\mathbf{g}}^b$ as:

$$\mathcal{GS}(\overrightarrow{\mathbf{g}}^a, \overrightarrow{\mathbf{g}}^b) = 1 - \frac{1}{Q}\sum_{i=0}^{Q-1} \text{XOR}(g_i^a, g_i^b)\,, \tag{3}$$

[6] For example, if a bar contains only two note onsets, at the beginning of the 1st beat and 2nd beat respectively, then the corresponding $\overrightarrow{\mathbf{g}}$ will have $g_0, g_{16} = 1$, and the rest dimensions 0.

where Q is the dimensionality of $\vec{g}^a$, $\vec{g}^b$, and XOR$(\cdot,\cdot)$ is the exclusive OR operation. Note that the value of $\mathcal{GS}(\cdot,\cdot)$ would always lie in between 0 and 1.

The grooving pattern similarity helps in measuring the music's rhythmicity. If a piece possesses a clear sense of rhythm, the grooving patterns between pairs of bars should be similar, thereby producing high $\mathcal{GS}$ scores; on the other hand, if the rhythm feels unsteady, the grooving patterns across bars should be erratic, resulting in low $\mathcal{GS}$ scores.

5.3 Chord Progression Irregularity

To measure the irregularity of a chord progression, we begin by introducing the term *chord trigram*, which is a triple composed of 3 consecutive chords in a chord progression; for example, (`Dm7`, `G7`, `CM7`). Then, *the chord progression irregularity* ($\mathcal{CPI}$) is defined as the percentage of *unique chord trigrams* in the chord progression of an entire piece. Please note that 2 chord trigrams are considered different if any of their elements does not match.

It is common for Jazz compositions to make use of 8- or 12-bar-long templates of chord progressions (known as the *8-*, or *12-bar blues*), which themselves can be broken down into similar substructures [25, 35], as the foundation of a section, and more or less "copy-paste" them to form the complete song with, say, `AABA` parts. Therefore, a well-composed Jazz piece should have a chord progression irregularity that is not too high.

5.4 Structureness Indicators

The *structureness* of music is induced by the repetitive musical content in the composition. It can involve multiple granularities, ranging from an instant musical idea to an entire section. From a psychological perspective, the appearance of repeated structures is the essence of the catchiness and the emotion-provoking nature of music [28].

The fitness scape plot algorithm [32, 33] and the associated SM Toolbox [34] offer an aesthetic way of detecting and visualizing the presence of repeating structures in music. The fitness scape plot is a matrix $S_{N \times N}$,[7] where $s_{ij} \in [0,1]$ is the *fitness*, namely, the degree of repeat in the piece derived from the *self-similarity matrix* (SSM) [16], of the segment specified by (i,j).

Our *structureness indicator* is based on the fitness scape plot and designed to capture the most salient repeat within a certain duration interval. For brevity of the mathematical representation, we assume the sampling frame rate of S is 1 Hz (hence N will be the piece's duration in seconds), and define the structureness indicator as follows:

$$\mathcal{SI}_l^u(S) = \max_{\substack{l \le i \le u \\ 1 \le j \le N}} S \,, \tag{4}$$

where l, u[8] are the lower and upper bounds of the duration interval (in seconds) one is interested in. In our experiments, we choose the structureness indicators of $\mathcal{SI}_3^8$, $\mathcal{SI}_8^{15}$, and $\mathcal{SI}_{15}$ to examine the short-, medium-, and long-term structureness respectively.

[7] N is the number of frames sampled from the audio of a piece, the 1st axis represents the segment duration (in frames), and the 2nd axis represents the center of segment (in frames).

[8] If present, otherwise l defaults to 1, and u defaults to N.

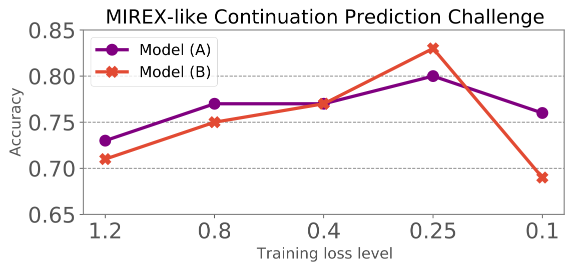

Figure 3. Result of the MIREX-like continuation prediction challenge, each checkpoint is asked 100 questions. Notice that the accuracy of both Model (A) and (B) peaks at the loss level of 0.25, at 80% and 83% respectively.

5.5 MIREX-like Continuation Prediction Challenge

Being inspired by the "Patterns for Prediction Challenge" held as part of the Music Information Retrieval Evaluation eXchange (MIREX) 2019 [1, 23], we developed a method to test the model's capability to predict the correct continuation given a musical prompt. The challenge is carried out as follows: First, the model is fed with the beginning 8 bars of a piece, denoted by $\vec{s}$; then, it is presented with a set of four 8-bar continuations $\mathcal{X} = \{\vec{x}^0, \vec{x}^1, \vec{x}^2, \vec{x}^3\}$, in which one is the true continuation, and the rest are wrong answers randomly drawn from other pieces. The way the model attempts to answer the multiple choice question is by calculating the average probability of generating the events of each continuation:

$$\mathcal{P}(\vec{x}^i) = \frac{1}{L} \sum_{j=0}^{L-1} p(x_j^i \mid \tilde{x}_{j-1}, \ldots, \tilde{x}_0; \vec{s}), \; i \in \{0,1,2,3\} \,, \tag{5}$$

where L is the length of the shortest given continuation (in # events) in $\mathcal{X}$, x_j^i is the j-th event token in $\vec{x}^i$, and $\tilde{x}_{j-1}, \ldots, \tilde{x}_0$ are the events sampled from the model's output, hence the conditional probability $p(x_j^i)$ at each timestep can be obtained straightforward. Finally, the model returns $\arg\max_i \mathcal{P}(\vec{x}^i)$ as its answer, of which the correctness we can check.

If the model can achieve high accuracy on this continuation prediction task, we may say it possesses a good overall understanding of Jazz music, enough for it to tell right from wrong when given multiple choices.

6. EXPERIMENT RESULTS AND DISCUSSIONS

We begin with the evaluation on the MIREX-like challenge (Section 5.5). We pick 5 checkpoints of both Model (A) and Model (B) at different training loss levels to ask each of them 100 multiple choice questions (the prompt and continuation choices of each question are randomly drawn from the held-out validation data). The result shown in Figure 3 indicates that, similarly for both models, the accuracy steadily goes up as the training loss decreases, peaks at the loss level of 0.25, and drops afterwards. This shows that the models are gradually gaining knowledge about Jazz music along the training process until a certain point, where they potentially start to overfit.

	Model (A)		Model (B)			Real
loss	*0.80*	*0.25*	*0.80*	*0.25*	*0.10*	- -
$\mathcal{H}_1$	2.29	2.45	2.26	2.20	**2.17**	1.94
$\mathcal{H}_4$	3.12	3.05	3.04	**2.91**	2.94	2.87
$\mathcal{GS}$	**0.76**	0.69	0.75	**0.76**	**0.76**	0.86
$\mathcal{CPI}$	81.2	77.6	79.2	**72.6**	75.9	40.4
$\mathcal{SI}_3^8$	0.18	0.22	0.25	**0.27**	0.26	0.36
$\mathcal{SI}_8^{15}$	0.15	0.17	**0.18**	**0.18**	0.17	0.36
$\mathcal{SI}_{15}$	0.11	**0.14**	0.10	0.12	0.11	0.35

Table 2. Results of objective evaluations. $\mathcal{H}_1$, $\mathcal{H}_4$ are the 1-, and 4-bar pitch class histogram entropy (see Sec. 5.1); $\mathcal{GS}$ is the grooving pattern similarity (Sec. 5.2) measured on all pairs of bars within a piece; $\mathcal{CPI}$ is the chord progression irregularity (in %; Sec. 5.3); finally, $\mathcal{SI}_3^8$, $\mathcal{SI}_8^{15}$, and $\mathcal{SI}_{15}$ are the short-, medium-, and long-term structureness indicators (Sec. 5.4). **Bold** texts indicate the model checkpoint performing the closest to real data, which is considered to be the best. It is observed that Model (B) (i.e., the model trained with structure-related events) with a loss of 0.25 outperforms its counterparts at other loss levels and Model (A) on most of the metrics. Moreover, consistent with the result of the MIREX-like challenge (Fig. 3), the performance of Model (B) plunges when the loss goes too low (0.1 in this case).

Following the MIREX-like challenge, we pick several checkpoints of both Models (A) and (B) for objective evaluations described in Sections 5.1–5.4. The chosen checkpoints are at loss levels 0.8, 0.25, and 0.1 (for Model (B) only, since in the MIREX-like challenge (Fig. 3), its accuracy drastically drops when the loss reduces from 0.25 to 0.1). In the experiments, 50 32-bar-long from-scratch compositions from each checkpointed model are compared against the 409 pieces in the training dataset.

From the results (Table 2), we can summarize the model's deficiencies as follows: 1) the *erraticity* of the generated musical events; and, 2) the *absence* of medium- and long-term repetitive structures. Comparing with the real data, the first argument can be justified by the higher $\mathcal{H}_1$ and $\mathcal{H}_4$, manifesting the unstable usage of pitches at local scale; and, the lower $\mathcal{GS}$ and higher $\mathcal{CPI}$ of the entire pieces, marking the lack of consistency in rhythm and harmony from a global point of view; meanwhile, the second argument can be explained directly by the significantly lower values of structureness indicators $\mathcal{SI}_8^{15}$ and $\mathcal{SI}_{15}$, suggesting that while the model might be able to repeat some short fragments of music, creating structures of a longer time span is still beyond its capability.

Much to our delight, the introduction of structure-related events seems to be functional to some extent, noticeable from the numbers that Model (B) at 0.25 loss level is for most of the time the closest competitor to humans, with a substantial lead on the metrics focusing on shorter timespans (i.e., $\mathcal{H}_1$, $\mathcal{H}_4$, and $\mathcal{SI}_3^8$) when placed in comparison with Model (A). This suggests that the use of PHRASE and MLU events provides some assistance to the model in modeling music. Furthermore, resonating with the accuracy trend in the MIREX-like challenge, the performance worsens when the loss is reduced to an overly low level.

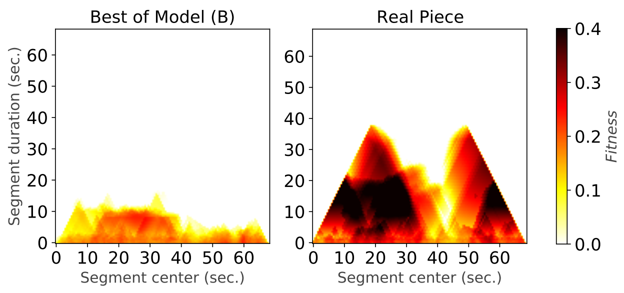

Figure 4. The fitness scape plots of Model (B)'s best composition (according to the *structureness* **(S)** score in our subjective study, see Sec. 4) versus a human composition in the WJazzD. Note that the piece by Model (B) contains almost no signs of repetition longer than 10 seconds, while the real piece's repetitive structures extend well into the 20–30 seconds range.

To visualize the deficiency in structureness of the model's compositions, we choose the piece which scores the highest, 3.14, in the *structureness* **(S)** aspect in our subjective study; and, a human composition of the same duration, receiving 3.54 in the aspect **S**, for a head-to-head comparison of their fitness scape plots. The rivalry (see Figure 4) reveals the stark contrast between their fitness values across all timescales. In the model's work, all traces of repetitive structures disappear at the timescale of 10 seconds; whereas in the human composition, not only do the fitness values stay high in longer timespans, but a clear sense of section is also present, as manifested by the 2 large, dark "triangles" in its scape plot.

7. CONCLUSION AND FUTURE WORK

In this paper, we have presented the Jazz Transformer, whose incorporation of structure-related events has been shown useful here in enhancing the quality of machine-generated music. Moreover, we have proposed a series of objective metrics that shed light on the shortcomings of machine-composed pieces, including the erratic usage of pitch classes, inconsistent grooving pattern and chord progression; and, the absence of repetitive structures. These metrics not only show that the Transformer is in fact not that good a music composer, but also serve as effective quantitative measures for future efforts in automatic music composition to assess their models' performance, which by now still relies heavily on human evaluation.

In the future, we plan to carry out larger-scale studies to explicate the correlations between those quantitative metrics and the aspects of subjective evaluation; and, to continue working on inducing structures in machine-composed music; such endeavors may not stay on revamping events that fit into Transformers as done, but involve a complete redesign of the Transformer architecture, enabling it to read the structural information directly computable from data, say, the fitness scape plot, to grasp the blueprint of a piece before composing music at finer scales.

8. ACKNOWLEDGEMENTS

The authors would like to express the utmost gratitude to **the Jazzomat Research Project** (University of Music FRANZ LISZT Weimar), for compiling the WJazzD dataset and making it publicly available; **Wen-Yi Hsiao** (Taiwan AI Labs), for rendering the MIDIs to audios for subjective study; and **Yi-Jen Shih** (National Taiwan University), for the help in arranging our open-source codes.

9. REFERENCES

[1] The "Patterns for Prediction Challenge" of Music Information Retrieval Evaluation eXchange. [Online] `https://www.music-ir.org/mirex/wiki/2019:Patterns_for_Prediction`.

[2] The Weimar Jazz Database. [Online] `https://jazzomat.hfm-weimar.de/`.

[3] Jakob Abeßer, Stefan Balke, Klaus Frieler, Martin Pfleiderer, and Meinard Müller. Deep learning for Jazz walking bass transcription. In *Proc. AES International Conference on Semantic Audio*, 2017.

[4] Jakob Abeßer, Estefanía Cano, Klaus Frieler, Martin Pfleiderer, and Wolf-Georg Zaddach. Score-informed analysis of intonation and pitch modulation in Jazz solos. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pages 823–829, 2015.

[5] Jakob Abeßer, Klaus Frieler, Martin Pfleiderer, and Wolf-Georg Zaddach. Introducing the Jazzomat project – Jazz solo analysis using music information retrieval methods. In *Proc. International Symposium on Computer Music Multidisciplinary Research (CMMR)*, 2013.

[6] Christopher Anderson, Arne Eigenfeldt, and Philippe Pasquier. The generative electronic dance music algorithmic system (GEDMAS). In *Proc. Artificial Intelligence and Interactive Digital Entertainment Conference*, 2013.

[7] Cedric De Boom, Stephanie Van Laere, Tim Verbelen, and Bart Dhoedt. Rhythm, chord and melody generation for lead sheets using recurrent neural networks. *arXiv preprint arXiv:2002.10266*, 2020.

[8] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. *Deep Learning Techniques for Music Generation, Computational Synthesis and Creative Systems*. Springer, 2019.

[9] Tsung-Ping Chen and Li Su. Harmony Transformer: Incorporating chord segmentation into harmony recognition. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pages 259–267, 2019.

[10] Ching-Hua Chuan and Dorien Herremans. Modeling temporal tonal relations in polyphonic music through deep networks with a novel image-based representation. In *Proc. AAAI Conference on Artificial Intelligence*, 2018.

[11] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2978–2988, 2019.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] Chris Donahue, Huanru Henry Mao, Yiting Ethan Li, Garrison W. Cottrell, and Julian McAuley. LakhNES: Improving multi-instrumental music generation with cross-domain pre-training. In *Proc. International Society for Music Information Retrieval (ISMIR)*, pages 685–692, 2019.

[14] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proc. AAAI Conference on Artificial Intelligence*, pages 34–41, 2018.

[15] Vsevolod Eremenko, Emir Demirel, Baris Bozkurt, and Xavier Serra. Audio-aligned Jazz harmony dataset for automatic chord transcription and corpus-based research. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, pages 483–490, 2018.

[16] Jonathan Foote. Visualizing music and audio using self-similarity. In *Proc. ACM International Conference on Multimedia*, pages 77–80, 1999.

[17] Klaus Frieler, Martin Pfleiderer, Wolf-Georg Zaddach, and Jakob Abeßer. Midlevel analysis of monophonic Jazz solos: A new approach to the study of improvisation. *Musicae Scientiae*, 20:143–162, 2016.

[18] Jeff Gregorio and Youngmoo Kim. Phrase-level audio segmentation of jazz improvisations informed by symbolic data. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[19] Ryan Groves. Automatic melodic reduction using a supervised probabilistic context-free grammar. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[20] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music Transformer: Generating music with long-term structure. In *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[21] Yu-Siang Huang and Yi-Hsuan Yang. Pop Music Transformer: Beat-based modeling and generation of expressive Pop piano compositions. In *Proc. ACM International Conference on Multimedia*, 2020.

[22] Hsiao-Tzu Hung, Chung-Yang Wang, Yi-Hsuan Yang, and Hsin-Min Wang. Improving automatic jazz melody generation by transfer learning techniques. In *Proc. Asia Pacific Signal and Information Processing Association Annual Summit and Conf. (APSIPA ASC)*, 2019.

[23] Berit Janssen, Tom Collins, and Iris Ren. Algorithmic ability to predict the musical future: Datasets and evaluation. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pages 208–215, 2019.

[24] Harsh Jhamtani and Taylor Berg-Kirkpatrick. Modeling self-repetition in music generation using structured adversaries. In *Proc. Machine Learning for Media Discovery Workshop, extended abstract*, 2019.

[25] Philip Johnson-Laird. How Jazz musicians improvise. *Music Perception — MUSIC PERCEPT*, 19:415–442, 2002.

[26] Shigeki Karita et al. A comparative study on Transformer vs RNN in speech applications. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*, pages 449–456, 2019.

[27] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proc. International Conference on Machine Learning*, 2020.

[28] Daniel J. Levitin. *This is Your Brain on Music: The Science of a Human Obsession*. Dutton, 2006.

[29] Hyungui Lim, Seungyeon Rhyu, and Kyogu Lee. Chord generation from symbolic melody using BLSTM networks. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pages 621–627, 2017.

[30] Hao-Min Liu, Meng-Hsuan Wu, and Yi-Hsuan Yang. Lead sheet generation and arrangement via a hybrid generative model. In *Proc. International Society for Music Information Retrieval Conference (ISMIR), late-breaking demo*, 2018.

[31] Gabriele Medeot, Srikanth Cherla, Katerina Kosta, Matt McVicar, Samer Abdallah, Marco Selvi, Ed Newton-Rex, and Kevin Webster. StructureNet: Inducing structure in generated melodies. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pages 725–731, 2018.

[32] Meinard Müller, Peter Grosche, and Nanzhu Jiang. A segment-based fitness measure for capturing repetitive structures of music recordings. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, pages 615–620, 2011.

[33] Meinard Müller and Nanzhu Jiang. A scape plot representation for visualizing repetitive structures of music recordings. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, pages 97–102, Porto, Portugal, 2012.

[34] Meinard Müller, Nanzhu Jiang, and Harald G. Grohganz. SM Toolbox: MATLAB implementations for computing and enhancing similarity matrices. In *Proc. Audio Engineering Society (AES)*, 2014.

[35] Simon John Nelson. *Melodic improvisation on a twelve bar blues model: an investigation of physical and historical aspects and their contribution to performance*. PhD thesis, City University London, 2001.

[36] Christine McLeavy Payne. MuseNet. *OpenAI Blog*, 2019.

[37] Martin Pfleiderer, Klaus Frieler, Jakob Abeßer, Wolf-Georg Zaddach, and Benjamin Burkhart, editors. *Inside the Jazzomat — New Perspectives for Jazz Research*. Schott Campus, 2017.

[38] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *Open AI Blog*, 1(8), 2019.

[39] Shakeel Raja. Music generation with temporal structure augmentation. *arXiv preprint arXiv:2004.10246*, 2020.

[40] Claude E Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[41] Bob L. Sturm and Oded Ben-Tal. Taking the models back to music practice: Evaluating generative transcription models built using deep learning. *Journal of Creative Music Systems*, 2(1), 2017.

[42] Nicholas Trieu and Robert M. Keller. JazzGAN: Improvising with generative adversarial networks. In *Proc. International Workshop on Musical Metacreation*, 2018.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017.

[44] Christof Weiss, Stefan Balke, Jakob Abeßer, and Meinard Müller. Computational corpus analysis: A case study on Jazz solos. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, pages 416–423, 2018.

[45] Li-Chia Yang and Alexander Lerch. On the evaluation of generative models in music. *Neural Computing and Applications*, 2018.