



Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions

Yu-Siang Huang

Taiwan AI Labs & Academia Sinica
Taipei, Taiwan
yshuang@ailabs.tw

Yi-Hsuan Yang

Taiwan AI Labs & Academia Sinica
Taipei, Taiwan
yhyang@ailabs.tw

ABSTRACT

A great number of deep learning based models have been recently proposed for automatic music composition. Among these models, the Transformer stands out as a prominent approach for generating expressive classical piano performance with a coherent structure of up to one minute. The model is powerful in that it learns abstractions of data on its own, without much human-imposed domain knowledge or constraints. In contrast with this general approach, this paper shows that Transformers can do even better for music modeling, when we improve the way a musical score is converted into the data fed to a Transformer model. In particular, we seek to impose a metrical structure in the input data, so that Transformers can be more easily aware of the beat-bar-phrase hierarchical structure in music. The new data representation maintains the flexibility of local tempo changes, and provides hurdles to control the rhythmic and harmonic structure of music. With this approach, we build a Pop Music Transformer that composes Pop piano music with better rhythmic structure than existing Transformer models.

CCS CONCEPTS

• Applied computing → Sound and music computing.

KEYWORDS

Automatic music composition, transformer, neural sequence model

ACM Reference Format:

Yu-Siang Huang and Yi-Hsuan Yang. 2020. Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions. In *28th ACM International Conference on Multimedia (MM '20)*, October 12–16, 2020, Seattle, WA, USA.. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3394171.3413671>

1 INTRODUCTION

Music is sound that's organized on purpose on many time and frequency levels to express different ideas and emotions. For example, the organization of musical notes of different fundamental frequencies (from low to high) influences the melody, harmony and texture of music. The placement of strong and weak beats over time, on the other hand, gives rise to the perception of rhythm [28].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MM '20, October 12–16, 2020, Seattle, WA, USA.

© 2020 Association for Computing Machinery.

ACM ISBN 978-1-4503-7988-5/20/10...\$15.00

<https://doi.org/10.1145/3394171.3413671>

	MIDI-like [30]	REMI (this paper)
Note onset	NOTE-ON (0–127)	NOTE-ON (0–127)
Note offset	NOTE-OFF (0–127)	NOTE DURATION (32th note multiples; 1–64)
Time grid	TIME-SHIFT (10–1000ms)	POSITION (16 bins; 1–16) & BAR (1)
Tempo changes	✗	TEMPO (30–209 BPM)
Chord	✗	CHORD (60 types)

Table 1: The commonly-used MIDI-like event representation [23, 30] versus the proposed beat-based one, REMI. In the brackets, we show the ranges of the type of event.

Repetition and long-term structure are also important factors that make a musical piece coherent and understandable.

Building machines that can compose music like human beings is one of the most exciting tasks in multimedia and artificial intelligence [5, 7, 13, 18, 26, 27, 31, 44, 45]. Among the approaches that have been studied, neural sequence models, which consider music as a *language*, stand out in recent work [9, 23, 30, 32] as a prominent approach with great potential.¹ In doing so, a digital representation of a musical score is converted into a time-ordered sequence of discrete tokens such as the NOTE-ON events. Sequence models such as the Transformer [39] can then be applied to model the probability distribution of the event sequences, and to sample from the distribution to generate new music compositions. This approach has been shown to work well for composing minute-long pieces of classical piano music with expressive variations in note density and velocity [23], in the format of a MIDI file.²

We note that there are two critical elements in the aforementioned approach—the way music is converted into discrete tokens for language modeling, and the machine learning algorithm used to build the model. Recent years have witnessed great progress regarding the second element, for example, by improving the self-attention in Transformers with “sparse attention” [8], or introducing a recurrence mechanism as in Transformer-XL for learning longer-range

¹However, whether music and language are related is actually debatable from a musicology point of view. The computational approach of modeling music in the same way as modeling natural language therefore may have limitations.

²Therefore, the model generates the pitch, velocity (dynamics), onset and offset time of each note, including those involved in the melody line, the underlying chord progression, and the bass line, etc in an expressive piano performance.

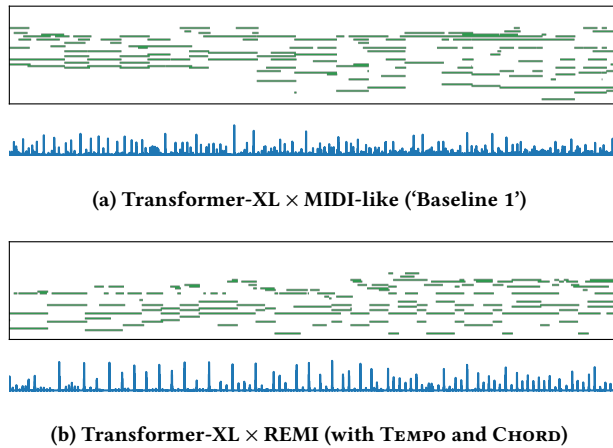


Figure 1: Examples of piano rolls and ‘downbeat probability curves’ (cf. Section 5.3) for music generated by an adaptation of the state-of-the-art model Music Transformer [23], and the proposed model. We can see clearer presence of regularly-spaced downbeats in the probability curve of (b).

dependency [11, 12]. However, no much has been done for the first one. Most work simply follows the *MIDI-like* event representation proposed by [30] to set up the “vocabulary” for music.

As shown in Table 1, the MIDI-like representation, for the case of modeling classical piano music [9, 23], uses NOTE-ON events to indicate the action of hitting a specific key of the piano keyboard, and NOTE-OFF for the release of the key. This representation has its roots in MIDI [22], a communication protocol that also uses NOTE-ON and NOTE-OFF messages to transmit data of real-time musical performance. Unlike *words* in human language, note messages in MIDI are associated with *time*. To convert a score into a sequence, [30] uses additionally the TIME-SHIFT (ΔT) events to indicate the relative time gap between events (rather than the absolute time of the events), thereby representing the advance in time. This MIDI-like representation is general and represents keyboard-style music (such as piano music) fairly faithfully, therefore a good choice as the format for the input data fed to Transformers. A general idea here is that a deep network can learn abstractions of the MIDI data that contribute to the generative musical modeling on its own. Hence, high-level (semantic) information of music [4], such as downbeat, tempo and chord, are not included in this MIDI-like representation.

However, we note that when humans compose music, we tend to organize regularly recurring patterns and accents over a *metrical structure* defined in terms of sub-beats, beats, and bars [10]. Such a structure is made clear on a score sheet or a MIDI file with notation of the time signature and bar lines, but is *implicit* in the MIDI-like event representation. A sequence model has to recover the metrical structure on its own from the provided sequence of tokens. When it comes to modeling music genres that feature the use of steady beats, we observe that the Transformer [23] has a hard time learning the regularity of beats over time, as shown in Figure 1a.

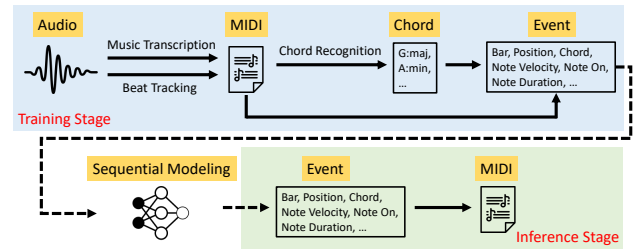


Figure 2: The diagram of the proposed beat-based music modeling and generation framework. The training stage entails the use of music information retrieval (MIR) techniques, such as automatic transcription, to convert an audio signal into an event sequence, which is then fed to a sequence model for training. At inference time, the model generates an original event sequence, which can then be converted into a MIDI file for playback.

To remedy this, and to study whether Transformer-based composition models can benefit from the addition of some human knowledge of music, we propose *REMI*, which stands for *revamped MIDI*-derived events, to represent MIDI data following the way humans read them. Specifically, we introduce the BAR event to indicate the beginning of a bar, and the POSITION events to point to certain locations within a bar. For example, POSITION (9/16) indicates that we are pointing to the middle of a bar, which is quantized to 16 regions in this implementation (see Section 3 for details). The combination of POSITION and BAR therefore provides an *explicit* metrical grid to model music, in a beat-based manner. This new data representation informs models the presence of a beat-bar hierarchical structure in music and empirically leads to better rhythmic regularity in the case of modeling Pop music, as shown in Figure 1b.

We note that, while the incorporation of the bars is not new in the literature of automatic music composition [6, 20, 38], it represents a brand new attempt in the context of Transformer-based music modeling. This facilitates extensions of the Transformer, such as the Compressive Transformer [33], Insertion Transformer [37], and Tree Transformer [40], to be studied for this task in the future, since the models have now clearer ideas of segment boundaries in music.

Moreover, we further explore adding other supportive musical tokens capturing higher-level information of music. Specifically, to model the expressive rhythmic freedom in music (e.g., tempo rubato), we add a set of TEMPO events to allow for local tempo changes per beat. To have control over the chord progression underlying the music being composed, we introduce the CHORD events to make the harmonic structure of music explicit, in the same vein of adding the POSITION and BAR events for rhythmic structure.

In achieving all these, the proposed framework for automatic music composition has to integrate audio- and symbolic-domain **music information retrieval** (MIR) techniques such as downbeat tracking [2, 3, 16] and chord recognition [17, 35], which differentiates it from existing approaches. Moreover, we use automatic music transcription [1, 21, 25] to prepare the MIDI data, so the model learns from **expressive piano performances**. We show in Figure 2 an illustration of the overall framework. Table 1 compares REMI

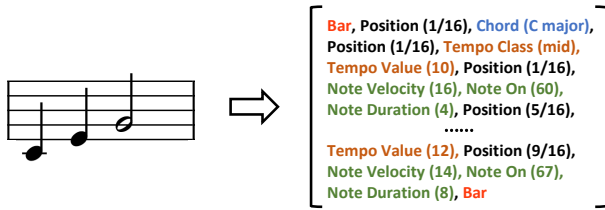


Figure 3: An example of a REMI event sequence and the corresponding music fragment in staff notation. The BAR, POSITION and TEMPO-related events entail the use of audio-domain downbeat and beat tracking algorithms [3] (see Section 3.3), and the CHORD events the use of a symbolic-domain chord recognition algorithm (see Section 3.5).

with the commonly-adopted MIDI-like representation, and Figure 3 gives an example of a REMI event sequence.

In our experiment (see Section 5), we use Transformer-XL [11] to learn to compose Pop piano music using REMI as the underlying data representation. We conduct both objective and subjective evaluation comparing the proposed model against the Music Transformer [23], showing that we are able to improve the state-of-the-art with simple yet major modifications in the data presentation, rather than with sophisticated re-design of the neural network architecture.

For reproducibility, the code, data and pre-trained model are made publicly available.³

2 RELATED WORK

Recent neural network-based approaches for automatic music composition can be broadly categorized into two groups. *Image-based* approaches such as MidiNet [41] and MuseGAN [13] use an image-like representation such as the piano roll to represent a score as a matrix of time steps and MIDI pitches, and then use convolution-based operations to generate music. It is easier for such approaches to learn the rhythmic structure of music, as the time steps corresponding to the beats and bar lines are clear in such matrices. *Language-based* approaches such as the Music Transformer [23], on the other hand, may learn the temporal dependency between musical events such as NOTE-ONS better. Yet, due to the limits of the MIDI-like representation outlined in Section 1, existing work may fall short in the rhythmic aspect. The proposed approach combines the advantage of the image- and language-based approaches by embedding a metrical grid in the event representation.

The idea of designing events for music metadata is similar to CTRL [24], which provided more explicit controls for text generation. However, recent work in neural sequence modeling of music mostly adopt the same MIDI-like event representation proposed by [30], or its extensions. For example, extra events denoting the composer, instrumentation and global tempo are used in MuseNet [32] for conditioned generation, and events specifying the NOTE-ON and NOTE-OFF of different instruments are used to achieve multi-instrument music composition [12]. However, to our best knowledge, the use of TIME-SHIFT to mark the advance in time has

been taken for granted thus far. While this approach has its own merits, we endeavor to explore alternative representations in this paper, again in the specific context of Transformer-based modeling.

3 NEW EVENT-BASED REPRESENTATION OF MUSIC

In this section, we discuss at length how ‘REMI’ is different from the commonly-adopted ‘MIDI-like’ representation (cf. Table 1), and how we design the proposed events.

As discussed in [30], a score to be converted into events can be either a *MIDI score* with no expressive dynamics and timing, or a *MIDI performance* that has been converted from an expressive audio recording by means of, for example, a MIDI keyboard, or automatic music transcription [1]. We consider the latter case in this paper, but the discussion below can also be generalized to the former case.

3.1 NOTE-ON and NOTE VELOCITY

The collection of 128 NOTE-ON events indicates the onset of MIDI pitches from 0 (C-1) to 127 (G9), and NOTE VELOCITY indicates the level of dynamics (which correspond to perceptual loudness) of the note event.⁴ Both the MIDI-like and REMI representations have these two types of events.

3.2 NOTE-OFF versus NOTE DURATION

In REMI, we use NOTE DURATION events in replacement of the NOTE-OFF events. Specifically, we represent each note in a given score with the following three consecutive tokens: a NOTE VELOCITY event, a NOTE-ON event, and a NOTE DURATION event. There are advantages in doing so:

- In MIDI-like, the duration of a note has to be inferred from the time gap between a NOTE-ON and the corresponding NOTE-OFF, by accumulating the time span of the TIME SHIFT events in between. In REMI, note duration is made explicit, facilitating modeling the rhythm of music.
- In MIDI-like, a NOTE-ON event and the corresponding NOTE-OFF are usually several events apart.⁵ As a result, a sequence model may find it difficult to learn that NOTE-ON and NOTE-OFF must appear in pairs, generating dangling NOTE-ON events without the corresponding NOTE-OFF. We would then have to use some heuristics (e.g., maximal note duration) to turn off a note in post-processing. With REMI, we are free of such an issue, because the sequence model can easily learn from the training data that a NOTE VELOCITY event has to be followed by a NOTE-ON event, and then right away a NOTE DURATION event.

3.3 TIME-SHIFT versus POSITION & BAR

Empirically, we find that it is not easy for a model to generate music with steady beats using the TIME-SHIFT events. When listening to the music generated, the intended bar lines drift over time and the rhythm feels unstable. We attribute this to the absence of a metrical structure in the MIDI-like representation. Mistakes in TIME-SHIFT

⁴Following [30], in our implementation we quantize note velocity into 32 levels, giving rise to 32 different NOTE VELOCITY events.

⁵For example, in our implementation, there are on average 21.7 ± 15.3 events between a pair of NOTE-ON and NOTE-OFF, when we adopt the MIDI-like event representation.

³<https://github.com/YatingMusic/remi>

would lead to accumulative error of timing in the inference phase, which is not obvious in the training phase due to the common use of teacher forcing strategy [14] in training recurrent models.

To address this issue, we propose to use the combination of BAR and POSITION events instead. Both of them are readily available from the musical scores, or from the result of automatic music transcription (with the help of a beat and downbeat tracker [3]); they are simply discarded in the MIDI-like representation. While BAR marks the bar lines, POSITION points to one of the Q possible discrete locations in a bar. Here Q is an integer that denotes the time resolution adopted to represent a bar. For example, if we consider a 16-th note time grid as [23, 34], $Q = 16$. Adding BAR and POSITION therefore provides a metrical context for models to “count the beats” and to compose music bar-after-bar.⁶⁷

We note that there are extra benefits in using POSITION & BAR, including 1) we can more easily learn the dependency (e.g., repetition) of note events occurring at the same POSITION (\star/Q) across bars; 2) we can add bar-level conditions to condition the generation process if we want; 3) we have time reference to coordinate the generation of different tracks for the case of multi-instrument music.

3.4 TEMPO

In an expressive musical performance, the temporal length (in seconds) of each bar may not be the same. To account for such local changes in tempo (i.e., beats per minute; BPM), we add TEMPO events every beat (i.e., at POSITION $(1/Q)$, POSITION $((Q/4 + 1)/Q)$, etc). In this way, we have a flexible time grid for expressive rhythm.⁸ Unlike the POSITION events, information regrading the TEMPO events may not be always available in a MIDI score. But, for the case of MIDI performances, one can derive such TEMPO events with the use of an audio-domain tempo estimation function [3].

3.5 CHORD

As another set of supportive musical tokens, we propose to encode the chord information into input events. Specifically, chords are defined as any harmonic set of pitches consisting of multiple notes sounding together or one after another. A chord consists of a root note and a chord quality [29]. Here, we consider 12 chord roots (C, C#, D, D#, E, F, F#, G, G#, A, A#, B) and five chord qualities (major, minor, diminished, augmented, dominant), yielding 60 possible CHORD events, covering the triad and seventh chords. However, the set of CHORD events can be further expanded as long as there is a way to get these chords from a MIDI score or a MIDI performance in the data preparation process. We note that the CHORD events are just “symbols”—the notes are still generated with the NOTE-ON events after them.

Following the time grid of REMI, each TEMPO or CHORD event is preceded by a POSITION event.

⁶We find in our implementation that models learn the meaning of BAR and POSITION quickly—right after a few epochs the model knows that, e.g., POSITION $(9/Q)$ cannot go before POSITION $(3/Q)$, unless there is a BAR in between. No postprocessing is needed to correct errors. See Figure 4 for the learning curve seen in our implementation.

⁷We note that the idea of using BAR and POSITION has been independently proposed before [19], though in the context of RNN-based not Transformer-based models.

⁸As exemplified in Figure 3, we use a combination of TEMPO CLASS events (low, mid, high) and TEMPO VALUE events to represent local tempo values of 30–209 BPM.

We note that music composed by a model using the MIDI-like representation also exhibits the use of chords, as such note combinations can be found in the training data by the sequence model itself. However, by explicitly generating TEMPO and CHORD events, tempo and chord become controllable.

4 PROPOSED FRAMEWORK

Again, a diagram of the proposed beat-based music modeling and generation framework can be found in Figure 2. We provide details of some of the major components below.

4.1 Backbone Sequence Model

The Transformer [39] is a neural sequence model that uses self-attention to bias its prediction of the current token based on a subset of the past tokens. This design has been shown effective for modeling the structure of music. For example, with the help of a relative-positional encoding method [36], Music Transformer is claimed in [23] to be able to compose expressive classical piano music with a coherent structure of up to one minute.

Transformer-XL [11] extends Transformer by introducing the notion of recurrence and revising the positional encoding scheme. The recurrence mechanism enables the model to leverage the information of past tokens beyond the current training segment, thereby looking further into the history. Theoretically, Transformer-XL can encode arbitrarily long context into a fixed-length representation. Therefore, we adopt Transformer-XL as our backbone model architecture.⁹

In the training process, each input sequence is divided into “segments” of a specific length (set to 512 events in our implementation). Given such segments, the overall computational procedure for an N -layer Transformer-XL with M heads can be summarized as:

$$\tilde{\mathbf{h}}_{\tau}^{n-1} = [\text{stop_gradient}(\mathbf{h}_{\tau-1}^{n-1}) \circ \mathbf{h}_{\tau}^{n-1}], \quad (1)$$

$$\mathbf{q}_{\tau}^n, \mathbf{k}_{\tau}^n, \mathbf{v}_{\tau}^n = \mathbf{h}_{\tau}^{n-1} \mathbf{W}_q^T, \tilde{\mathbf{h}}_{\tau}^{n-1} \mathbf{W}_k^T, \tilde{\mathbf{h}}_{\tau}^{n-1} \mathbf{W}_v^T, \quad (2)$$

$$\mathbf{a}_{\tau,i}^n = \text{masked_softmax}(\mathbf{q}_{\tau,i}^n \mathbf{k}_{\tau,i}^n + \mathbf{R}) \mathbf{v}_{\tau,i}^n, \quad (3)$$

$$\mathbf{a}_{\tau}^n = [\mathbf{a}_{\tau,1}^n \circ \mathbf{a}_{\tau,2}^n \circ \dots \circ \mathbf{a}_{\tau,m}^n]^T \mathbf{W}_o^n, \quad (4)$$

$$\mathbf{o}_{\tau}^n = \text{layernorm}(\mathbf{a}_{\tau}^n + \mathbf{h}_{\tau}^{n-1}), \quad (5)$$

$$\mathbf{h}_{\tau}^n = \max(0, \mathbf{o}_{\tau}^n \mathbf{W}_1^n + \mathbf{b}_1^n) \mathbf{W}_2^n + \mathbf{b}_2^n, \quad (6)$$

where \mathbf{h}_{τ}^n denotes the n -th layer hidden features for the τ -th segment, \mathbf{R} denotes the relative positional encodings designed for the Transformer-XL [11], $\mathbf{a}_{\tau,i}^n$ indicates the attention features from the i -th head, and \mathbf{q} , \mathbf{k} , \mathbf{v} denote the query, key and values in the computation of self-attention [39], respectively. The main difference between Transformer and Transformer-XL lies in the usage of the features from the segments prior to the current segment (i.e., the segments are from the same input MIDI sequence of a music piece) when updating the model parameters based on that current segment. This mechanism brings in longer temporal context and speeds up both the training and inference processes.

⁹We have implemented both Transformer- and Transformer-XL based models and found the latter composes Pop piano music with better temporal coherence perceptually. However, as the use of Transformer-XL for automatic music composition has been reported elsewhere [12], we omit such an empirical performance comparison between the XL and non-XL versions.

Chord	Required	Gain 1 point	Deduct 1 point	Deduct 2 points
Major	0, 4	7	2, 5, 9	1, 3, 6, 8, 10
Minor	0, 3	7	2, 5, 8	1, 4, 6, 9, 11
Diminished	0, 3, 6	9	2, 5, 10	1, 4, 7, 8, 11
Augmented	0, 4, 8	-	2, 5, 9	1, 3, 6, 7, 10
Dominant	0, 4, 7, 10	-	2, 5, 9	1, 3, 6, 8, 11

Table 2: The proposed rule-based scoring criteria for establishing the CHORD events via pitch intervals in the chromatic scale.

Transformer-like sequence models learn the dependency among elements (i.e., events here) of a sequence. Therefore, although we have quite a diverse set of events featuring different characteristics in the proposed REMI representation, we opt for the simple approach of using a single Transformer-XL to model all these events at once. In other words, we do not consider the alternative, possibly more computationally intensive, approach of establishing multiple Transformer models, one for a non-overlapping subset of the events, followed by some mechanisms to communicate between them.

4.2 Beat Tracking and Downbeat Tracking

To create the BAR events, we employ the recurrent neural network model proposed by [3] to estimate from the audio files the position of the ‘downbeats,’ which correspond to the first beat in each bar. The same model is used to track the beat positions to create the TEMPO events [2]. We obtain the tick positions between beats by linear interpolation, and then align the note onsets and offsets to the nearest tick. To eliminate the imprecision of transcription result, we further quantize the onset and offset times according to the assumed time grid (e.g., to the 16-th note when $Q = 16$).

4.3 Chord Recognition

We establish the 60 CHORD events described in Section 3.5 by designing a heuristic rule-based symbolic-domain chord recognition algorithm to the transcribed MIDI files. First, we compute binary-valued “chroma features” [17] for each tick, to represent the activity of 12 different pitch classes ignoring the pitch’s octave. Then, we use a sliding window to assign “likelihood scores” to every active note for each 2-beat and 4-beat segment. After summarizing the chroma features of the current segment, we consider every note in a segment as a candidate root note of the CHORD of that segment, and calculate its pitch intervals to all the other notes in that segment. The look-up table shown in Table 2 is employed to assign likelihood scores to a pair of root note and chord quality based on the pitch intervals. Each chord quality has its required set of pitch intervals, and scoring functions. Finally, we recursively label the segments by the CHORD symbol with the highest likelihood score.

5 EVALUATION

We report both objective and subjective evaluations aiming to validate the effectiveness of the proposed model over the Music Transformer [23] for the case of composing expressive Pop piano music. In what follows, we present the training data we used to train both

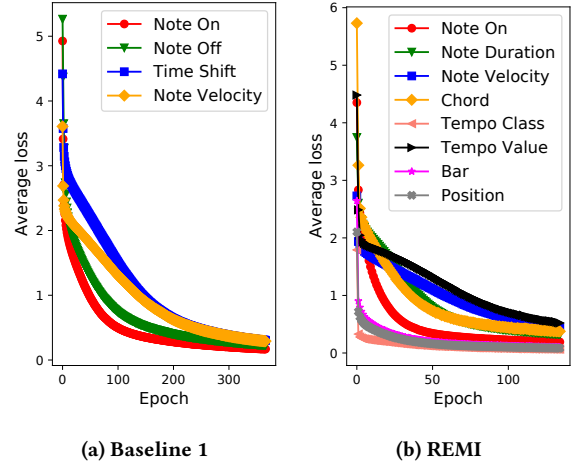


Figure 4: Average cross-entropy loss for different types of events as the training process proceeds. (a) An adapted version of the Music Transformer [23]; (b) the proposed model.

our model and variants of the the Music Transformer, then some implementation details, and finally the performance study.

5.1 Dataset

We intend to evaluate the effectiveness of the proposed approach for Pop piano composition, as Pop is a musical genre that features salient rhythmic structures. In doing so, we collect audio files of Pop piano music from the Internet. A total number of 775 pieces of piano music played by different people is collected, amounting to approximately 48 hours’ worth of data. They are covers¹⁰ of various Japanese anime, Korean popular and Western popular songs, playing only with the piano. We then apply “Onsets and Frames” [21], the state-of-the-art approach for automatic piano transcription, to estimate the pitch, onset time, offset time and velocity of the musical notes of each song, converting the audio recordings into MIDI performances.

We select the Pop piano covers such that they are all in 4/4 time signature, simplifying the modeling task. Accordingly, a bar is composed of four beats. Furthermore, following [23], we consider the 16-th note time grid and quantize each bar into $Q = 16$ intervals.

In our preliminary experiments, we have attempted to use a finer time grid (e.g., 32-th or 64-th note) to reduce quantization errors and to improve the expression of music (e.g., to include triplets, swing, micro-timing variations, etc). However, we found that this seems to go beyond the capability of the adopted Transformer-XL architecture. The generated compositions using a finer time grid tends to be fragmented and not pleasant to listen to. This is an issue that has to be addressed in the future. We stick with $Q = 16$ hereafter.

Due to copyright restrictions, we plan to make the training data publicly available not as audio files but as the transcribed MIDI files and the converted REMI event sequences.

¹⁰A ‘cover’ is a new recording by someone other than the original artist or composer of a commercially released song.

Method	Note offset	Time grid	TEMPO	CHORD	Beat STD	Downbeat STD	Downbeat salience
Baseline 1	NOTE-OFF	TIME-SHIFT (10-1000ms)			0.0968	0.3561	0.1033
Baseline 2	DURATION	TIME-SHIFT (10-1000ms)			0.0394	0.1372	0.1651
Baseline 3	DURATION	TIME-SHIFT (16th-note multiples)			0.0396	0.1383	0.1702
REMI	DURATION	POSITION & BAR	✓	✓	0.0386	0.1376	0.2279
	DURATION	POSITION & BAR	✓		0.0363	0.1265	0.1936
	DURATION	POSITION & BAR		✓	0.0292	0.0932	0.1742
	DURATION	POSITION & BAR			0.0199	0.0595	0.1880
Training data					0.0607	0.2163	0.2055

Table 3: Quantitative comparison of different models for Pop piano composition, evaluating how the composed music exhibit rhythmic structures. We report the average result across songs. For all the three objective metrics, the values are the closer to those of the ‘Training data’ shown in the last row the better—the training data are the transcribed and 16-th note quantized version of the 775 Pop songs. ‘Baseline 1’ represents a Transformer-XL version of the Music Transformer, adopting a MIDI-like event representation. The other two baselines are its improved variants. We highlight the two values closest to those of the training data in bold.

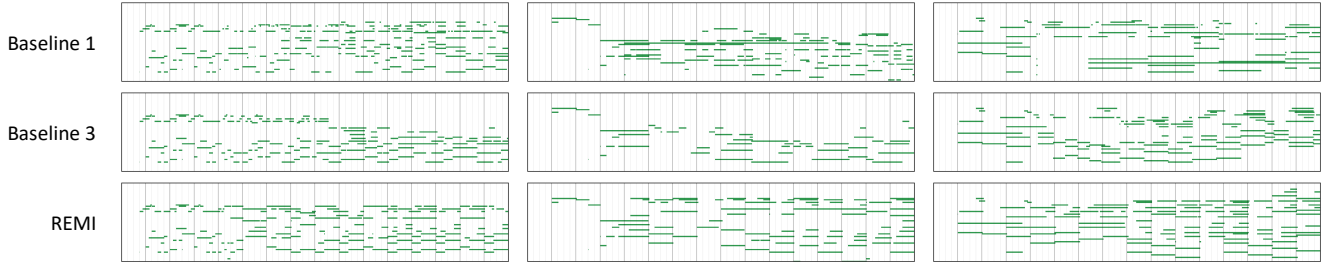


Figure 5: Examples of the generated piano rolls of different models. These are 12 bars generated to continue a given prompt of 4-bar human piano performance. We show the result for three different prompts (from left to right). The thicker vertical lines indicate the bar lines; best viewed in color.

5.2 Baselines & Model Settings

We consider three variants of the Music Transformer [23] as the baselines in our evaluation. The first one, dubbed **Baseline 1**, follows fairly faithfully the model settings of [23], except that we use Transformer-XL instead of Transformer (for fair comparison with our model). The other two differ from the first one only in the adopted event representation. While Baseline 1 employs exactly the MIDI-like representation, **Baseline 2** replaces NOTE-OFF by NOTE DURATION, and **Baseline 3** further modifies the time steps taken in TIME-SHIFT from multiples of 10ms to multiples of the 16-th note. In this way, Baseline 3 has a time grid similar to that of REMI, making Baseline 3 a strong baseline.

For either the baselines or the proposed model adopting the REMI representation, we train a single Transformer-XL with $N = 12$ self-attention layers and $M = 8$ attention heads. The length of the training input events (i.e., the segment length) and the recurrence length (i.e., the length of the segment “cached” [11]) are both set to 512. The total number of learnable parameters is approximately 41M. Training a model with an NVIDIA V100 with mini-batch size of 16 till the training loss (i.e., cross-entropy) reaches a certain

level takes about 9 hours. We find that Baseline 1 needs a smaller learning rate and hence longer training time. For processing the MIDI files, we use the miditoolkit.¹¹

Figure 4 shows the training loss of different event types as the training unfolds. Figure 4a shows that Baseline 1 struggles the most for learning TIME-SHIFT. Moreover, NOTE-OFF has higher loss than NOTE-ON, suggesting that they are not recognized as event pairs. In contrast, Figure 4b shows that the POSITION events in REMI are easy to learn, facilitating learning the rhythm of music.¹²

5.3 Objective Evaluation

The objective evaluation assesses the rhythmic structure of the generated compositions. Specifically, we employ each model to randomly generate 1,000 sequences, each with 4,096 events, using the temperature-controlled stochastic sampling method with top- k [24]. We convert the generated sequences into MIDI and then

¹¹<https://github.com/YatingMusic/miditoolkit>

¹²According to [21, 25], piano transcription models do better in estimating note onsets and pitches, than offsets and velocities. This may be why our model has higher loss for NOTE DURATION and NOTE VELOCITY.

render them into audio via a piano synthesizer.¹³ Then, we apply the joint beat and downbeat tracking model of [3] to the resulting audio clips.¹⁴

The model [3] has two components. The first one estimates the probability, or *salience*, of observing beats and downbeats for each time frame via a recurrent neural network (RNN). The second one applies a dynamic Bayesian network (DBN) to the output of the RNN to make binary decisions of the occurrence of beats and downbeats. We use the output of the RNN to calculate the downbeat salience, and the output of the DBN for the beat STD and downbeat STD. Specifically, given a piece of audio signal, the model [3] returns a time-ordered series of (τ_t^B, s_t^B) , where τ_t^B denotes the estimate time (in seconds) for the t -th beat by the DBN, and $s_t^B \in [0, 1]$ the salience associated with that beat estimated by the RNN. It also returns similarly a series of (τ_t^D, s_t^D) for the downbeats; see Figure 1 for examples of such series. From the tracking results, we calculate the following values for each audio clip:

- **Beat STD:** the standard deviation of $(\tau_t^B - \tau_{t-1}^B)$, which is always positive, over the beat estimates for that clip.
- **Downbeat STD:** similarly the standard deviation of $(\tau_t^D - \tau_{t-1}^D)$ over the downbeat estimates for that clip. Both beat STD and downbeat STD assess the consistency of the rhythm.
- **Downbeat salience:** the mean of s_t^D over the downbeat estimates for that clip, indicating the salience of the rhythm.

We report the average values across the audio clips, assuming that these values are closer to the values calculated from the training data (which have been quantized to the 16-th note grid) the better.¹⁵

Tables 3 shows the result of the baselines and a few variants (ablated versions) of the REMI-based model. From Beat STD and Downbeat STD, we see that the result of Baseline 1, which resembles Music Transformer [23], features fairly inconsistent rhythm, echoing the example shown in Figure 1a. The STD is lower when NOTE DURATION is used in place of NOTE-OFF, highlighting the effect of NOTE DURATION in stabilizing the rhythm. Interestingly, we see that the REMI models without the TEMPO events have much lower STD than the training data, suggesting that TEMPO is important for expressive rhythmic freedom.

From downbeat salience, the REMI models outnumber all the baselines, suggesting the effectiveness of POSITION & BAR. Moreover, the gap between the result of Baseline 1 and Baseline 2 further supports the use of the NOTE DURATION events.

5.4 Subjective Evaluation

The best way to evaluate a music composition model remains today to be via a listening test. To have a common ground to compare different models, we ask the models to *continue* the same given prompt. For doing so, we prepare an additional set of 100 Pop piano performances (that have no overlaps with the training set), process them according to the procedures described in Section 5.1, and take the first four bars from each of them as the prompts. The following three models are asked to generate 16 bars continuing each prompt,

¹³<https://github.com/YatingMusic/ReaRender>

¹⁴We choose to use the model of [3] for it achieved state-of-the-art performance for beat and downbeat tracking for a variety of musical genres, especially for Pop music.

¹⁵Small beat/downbeat STD implies that the music is too rigid, whereas large STD implies that the rhythm is too random. We want the rhythm to be stable yet flexible.

	Baseline 1	Baseline 3	REMI
'pros' group	1.77	2.03	2.20
'non-pros' group	1.66	1.90	2.44
all participants	1.73	1.98	2.28

Table 4: The average scores for the subjective preference test on a three-point scale from 1 (like the least) to 3 (like the most).

Pairs	Wins	Losses	p -value
REMI Baseline 1	103	49	5.623e-5
REMI Baseline 3	92	60	0.0187
Baseline 3 Baseline 1	90	62	0.0440

Table 5: The result of pairwise comparison from the user study, with the p -value of the Wilcoxon signed-rank test.

and got evaluated in an online listening test: 'Baseline 1,' 'Baseline 3' and 'REMI without CHORD.' We do not use CHORD here to make the rhythmic aspect the major point of difference among the models.

We distribute the listening test over our social circles globally and solicit the response from 76 participants. 51 of them understand basic music theory and have the experience of being an amateur musician, so we consider them as professionals (denoted as 'pros'). A participant has to listen to two randomly picked sets of samples and evaluate, for each set, which sample they *like the most* and *the least*, in any evaluation criteria of their choice. Each set contains the result of the three models (in random order) for a given prompt.

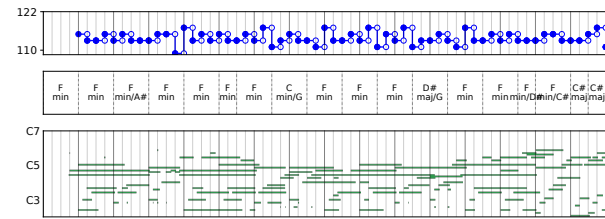
Table 4 shows the aggregated scores, and Table 5 the result of broken-down pairwise comparisons, along with the p -value of the Wilcoxon signed-rank test. We see that REMI is preferred by both pros and non-pros, and that the difference between REMI and Baseline 1 is significant ($p < 0.01$).

Figure 5 provides examples of the generated continuations. From the optional verbal feedbacks of the participants, the music generated by REMI are perceptually more pleasing and are more in line with the prompt. Audio examples can be found at our demo website.¹⁶

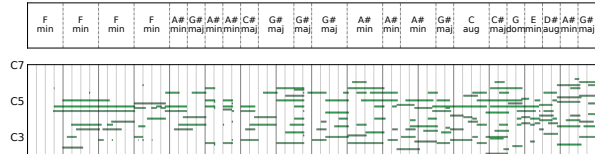
5.5 Controllable CHORD and TEMPO

Finally, we demonstrate the controllability of TEMPO and CHORD of our model. To achieve this, we can simply force the model not to generate specific events by masking out the corresponding probabilities of the model output. Figure 6 shows the piano rolls, (optionally) the TEMPO and CHORD events generated by our model under different conditions. Figure 6b shows that the model selects chords that are harmonically close to F:minor when we prohibit it from generating F:minor. Figure 6c shows controlling the TEMPO CLASS affects not only the TEMPO VALUE but also the note events.

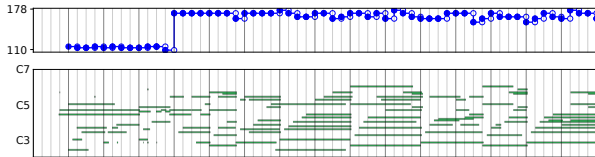
¹⁶<https://drive.google.com/drive/folders/1LzPBjHPip4S0CBOLquk5CNapvXSfys54>



(a) A human-made musical piece containing the prompt used below



(b) The result when we enforce no F:minor chord after the 4th bar



(c) The result when we enforce using TEMPO CLASS (HIGH)

Figure 6: An example illustrating how we can condition the TEMPO and CHORD events (the first four bars is the prompt).

6 CONCLUSION

In this paper, we have presented REMI, a novel MIDI-derived event representation for sequence model-based composition of expressive piano music. Using REMI as the underlying data representation, we have also built a Pop Music Transformer that outperforms that state-of-the-art Music Transformer for composing expressive Pop piano music, as validated in objective and subjective studies. We take this as a proof-of-concept that it is beneficial to embed prior human knowledge of music through including components of music information retrieval (MIR) techniques, such as downbeat estimation and chord recognition, to the overall generative framework. For future work, we plan to incorporate other high-level information of music, such as grooving [15] and musical emotion [42], to learn to compose additional tracks such as the guitar, bass, and drums, to further improve the resolution of the time grid, to condition the generative process with other multimedia input such as the lyrics [43] or a video clip [44], and also to study other model architectures that can model longer sequences (e.g., [33]) and thereby learn better the long-term structure in music.

7 ACKNOWLEDGEMENT

The authors would like to thank **Wen-Yi Hsiao** (Taiwan AI Labs) for helping with processing the MIDI data and with rendering the MIDI to audios for subjective study.

REFERENCES

- [1] Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. 2018. Automatic music transcription: An overview. *IEEE Signal Processing Magazine* 36, 1 (2018), 20–30.
- [2] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. 2016. Madmom: A new python audio and music signal processing library. In *Proc. ACM Multimedia*. 1174–1178.
- [3] Sebastian Böck, Florian Krebs, and Gerhard Widmer. 2016. Joint beat and downbeat tracking with recurrent neural networks. In *Proc. Int. Soc. Music Information Retrieval Conf.* 255–261.
- [4] Dmitry Bogdanov, J. Serrà, Nicolas Wack, and Perfecto Herrera. 2009. From low-level to high-level: Comparative study of music similarity measures. In *Proc. IEEE International Symposium on Multimedia*.
- [5] Paolo Bottoni, Anna Labella, Stefano Faralli, Mario Pierro, and Claudio Scorzafava. 2006. Interactive composition, performance and music generation through iterative structures. In *Proc. ACM Multimedia*. 189–192.
- [6] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. 2019. *Deep Learning Techniques for Music Generation, Computational Synthesis and Creative Systems*. Springer.
- [7] Gong Chen, Yan Liu, Sheng-hua Zhong, and Xiang Zhang. 2018. Musicality-novelty generative adversarial nets for algorithmic composition. In *Proc. ACM Multimedia*. 1607–1615.
- [8] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse Transformers. *arXiv preprint arXiv:1904.10509* (2019).
- [9] Kristy Choi, Curtis Hawthorne, Ian Simon, Monica Dinulescu, and Jesse Engel. 2020. Encoding musical style with transformer autoencoders. In *Proc. Int. Conf. Machine Learning*.
- [10] Grosvenor W Cooper, Grosvenor Cooper, and Leonard B Meyer. 1963. *The Rhythmic Structure of Music*. University of Chicago Press.
- [11] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proc. Annual Meeting of the Association for Computational Linguistics*. 2978–2988.
- [12] Chris Donahue, Huanru Henry Mao, Yiting Ethan Li, Garrison W. Cottrell, and Julian McAuley. 2019. LakhNES: Improving multi-instrumental music generation with cross-domain pre-training. In *Proc. Int. Soc. Music Information Retrieval Conf.* 685–692.
- [13] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. 2018. MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proc. AAAI* 34–41.
- [14] Kenji Doya. 1992. Bifurcations in the learning of recurrent neural networks 3. *learning (RTRL)* 3 (1992), 17.
- [15] Jan Frühauf, Reinhard Kopiez, and Friedrich Platz. 2013. Music on the timing grid: The influence of microtiming on the perceived groove quality of a simple drum pattern performance. *Musicae Scientiae* 17, 2 (2013), 246–260.
- [16] Magdalena Fuentes, Brian McFee, C. Hélène Crayencour, Slim Essid, and Pablo Juan Bello. 2018. Analysis of common design choices in deep learning systems for downbeat tracking. In *Proc. Int. Soc. Music Information Retrieval Conf.* 106–112.
- [17] Takuya Fujishima. 1999. Realtime chord recognition of musical sound: A system using common Lisp. In *Proc. International Computer Music Conf.* 464–467.
- [18] Lorenzo Gatti, Gözde Özbal, Oliviero Stock, and Carlo Strapparava. 2017. Automatic generation of lyrics parodies. In *Proc. ACM Multimedia*. 485–491.
- [19] Benjamin Genchel, Ashis Pati, and Alexander Lerch. 2019. Explicitly conditioned melody generation: A case study with interdependent RNNs. In *Proc. Computer Simulation of Music Creativity Conf.*
- [20] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. 2017. DeepBach: A Steerable Model for Bach chorales generation. In *Proc. Int. Conf. Machine Learning*. 1362–1371.
- [21] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. 2018. Onsets and Frames: Dual-objective piano transcription. In *Proc. Int. Soc. Music Information Retrieval Conf.* 50–57.
- [22] Jim Heckroth. 1998. A tutorial on MIDI and wavetable music synthesis. *Application Note, CRYSTAL a division of CIRRRUS LOGIC* (1998).
- [23] Cheng-Zhi A. Huang et al. 2019. Music Transformer: Generating music with long-term structure. In *Proc. Int. Conf. Learning Representations*.
- [24] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858* (2019).
- [25] Jong Wook Kim and Juan Pablo Bello. 2019. Adversarial learning for improved onsets and frames music transcription. In *Proc. Int. Soc. Music Information Retrieval Conf.* 670–677.
- [26] Chien-Hung Liu and Chuan-Kang Ting. 2017. Computational intelligence in music composition: A survey. *IEEE Transactions on Emerging Topics in Computational Intelligence* 1, 1 (2017), 2–15.

- [27] Qi Lyu, Zhiyong Wu, and Jun Zhu. 2015. Polyphonic music modelling with LSTM-RTRBM. In *Proc. ACM Multimedia*. 991–994.
- [28] Jason Martineau. 2008. *The Elements of Music: Melody, Rhythm, and Harmony*. Bloomsbury Publishing USA.
- [29] B. McFee and J. P. Bello. 2017. Structured training for large-vocabulary chord recognition. In *Proc. Int. Soc. Music Information Retrieval Conf.* 188–194.
- [30] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. 2018. This time with feeling: Learning expressive musical performance. *Neural Computing and Applications* (2018).
- [31] François Pachet. 2016. A Joyful Ode to automatic orchestration. *ACM Transactions on Intelligent Systems and Technology* 8, 2 (2016).
- [32] Christine McLeavy Payne. 2019. MuseNet. *OpenAI Blog* (2019).
- [33] Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive Transformers for long-range sequence modelling. In *Proc. Int. Conf. Learning Representations*.
- [34] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A hierarchical latent vector model for learning long-term structure in music. In *Proc. Int. Conf. Machine Learning*. 4361–4370.
- [35] Ricardo Scholz and Geber Ramalho. 2008. Cochonut: Recognizing complex chords from MIDI guitar sequences. In *Proc. Int. Soc. Music Information Retrieval Conf.* 27–32.
- [36] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proc. Conf. the North American Chapter of the Association for Computational Linguistics*. 464–468.
- [37] Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion Transformer: Flexible sequence generation via insertion operations. In *Proc. Int. Conf. Machine Learning*. 5976–5985.
- [38] Bob L. Sturm, João F. Santos, Oded Ben-Tal, and Iryna Korshunova. 2016. Music transcription modelling and composition using deep learning. In *Proc. Conf. Computer Simulation of Musical Creativity*.
- [39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. Advances in Neural Information Processing Systems*. 5998–6008.
- [40] Yaoshian Wang, Hung-Yi Lee, and Yun-Nung Chen. 2019. Tree Transformer: Integrating tree structures into self-attention. In *Proc. Conf. Empirical Methods in Natural Language Processing*. 1061–1070.
- [41] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. 2017. MidiNet: A convolutional generative adversarial network for symbolic-domain music generation. In *Proc. Int. Soc. Music Information Retrieval Conf.* 324–331.
- [42] Yi-Hsuan Yang and Homer H. Chen. 2011. *Music Emotion Recognition*. CRC Press.
- [43] Yi Yu and Simon Canales. 2019. Conditional LSTM-GAN for melody generation from lyrics. *arXiv preprint arXiv:1908.05551* (2019).
- [44] Yi Yu, Zhijie Shen, and Roger Zimmermann. 2012. Automatic music soundtrack generation for outdoor videos from contextual sensor information. In *Proc. ACM Multimedia*. 1377–1378.
- [45] Hongyuan Zhu et al. 2018. Xiaolce Band: A melody and arrangement generation framework for Pop music. In *Proc. ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*. 2837–2846.