Vol.2019-MUS-125 No.5
2019/11/19

リズム・コード・メロディの三段階深層学習による音楽生成

松村昂輝[†1]　大谷紀子[†2]　木村司[†3]　福井健一[†3]　沼尾正行[†3]

概要：本研究では，深層学習の手法を用いた新しい音楽生成方法を提案する．音楽理論の観点から，コードとメロディは互いに調和する必要がある．つまり，音楽を作る際には、コードを考慮した上でメロディを作る事が望ましい．本研究では，コードを考慮したメロディ生成を行う深層学習モデルを対象とする．既存研究において，音楽の重要な要素であるリズムが考慮されたモデルは存在しない．本研究では，リズム・コード・メロディを一括に考慮して音楽を生成する新しい手法を提案する．本手法では，以下の三段階の手順に従い，複数の深層学習モデルを用いて音楽を生成する．①RhythmGenerator，RhythmTransformer により，大量の音楽データから学習した，コード及びメロディのリズムを生成する．②生成されたリズムを入力とし，RhythmToChordGenerator がコードを生成する．③生成されたリズムとコードから，ChordToMelodyGenerator によりメロディを生成する．提案手法を用いた実験では，リズム・コード・メロディが調和した音楽を生成することができた．

1. はじめに

近年，深層学習による音楽生成モデルが数多く提案されている．Hao-Wen Dong らは，敵対的生成ネットワーク（GANs）を用いたマルチトラック生成モデルを提案している [1]．また Cheng-Zhi Anna Huang らは，Transformer を音楽などの長いシーケンス生成に適応した新しいモデルを提案している [2]. これらは, 音楽を構成する要素であるコードやメロディなどを段階的に考慮した生成手法を用いていない．一方，Gino Brunner らは，コードを考慮してメロディを生成する JamBot [3]を提案した．このモデルで生成されるコード進行は五度圏の特徴を掴んでおり，音楽理論を考慮しながらモデルが音楽を生成していることが分かった．しかし，JamBot では音楽の重要な要素であるリズムが考慮されていないといった問題がある．本研究では，リズム・コード・メロディのそれぞれの関係性を考慮し，音楽を生成する新しいモデルを提案する．提案モデルでは，リズム生成器によって，初めにコードとメロディのリズムが予測される．コードのリズムの情報はコード生成器に入力され，次のコードが予測される．最後に，メロディのリズムと生成されたコードを元にメロディ生成器がメロディを予測する．提案モデルにより，リズムを持ちながらコードとメロディが調和した音楽を生成することが出来た．

次章以降，本論文の構成は以下のようになっている．第二章ではデータセットとデータの処理方法について述べる．次に，提案手法の解説とモデルの学習について第三章で述べた後，第四章で，音楽生成の実験とその結果を示し，結論を第六章で述べる．

2. データの前処理

本章では，使用するデータセットと，学習モデルへ入力する前に行うデータ処理方法について説明する．

2.1 Wikifonia 楽譜データセット

本研究では，Wikifonia.org が提供する約 6 千曲の楽譜データを使用する [4]. 楽譜データは, 西洋音楽の様々なジャンルの楽曲で構成されている．各楽譜データには，楽曲のメロディとそれに対応する歌詞，コード名が記載されている．メロディとコードのデータを抽出し，それらが演奏される位置の情報により，メロディとコードのリズムを抽出する．データ処理の過程で，本モデルで使用可能なデータを抽出できたのは 3, 267 曲であった．

2.2 データ前処理

次にデータの前処理について説明する．メロディ，コード，リズムに対して，それぞれ以下のように処理を行う．また，データ処理には Python の Music21 ライブラリ [5]を使用した．

- メロディ

メロディは，Midi 形式で扱われる 128 種類のピッチ番号（0 から 127 までの数値）で表す．3,267 曲を解析すると，全ての楽曲で使用されるメロディのピッチ番号は 34 から 96 までの 62 種類であった．

- コード

コード名から，そのコードに含まれる各音符のピッチを 1 オクターブ分のタプル，つまり 0 から 11 までの数字の組み合わせで表す. 例えば, C コードに含まれる音符は（ド，ミ，ソ）であり，これは（0，3，5）のタプルで表わされる．

- リズム

本研究では，リズムを「音符またはコードが持つ拍の長さ」と「同じ長さの要素が続く回数」のペアと定義する．図 1 にリズムの抽出方法の例を示す．これらの音符のリズムは[(0.5, 4), (1, 1), …]のように表され，(0.5, 4)は「8 分音符

1 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University
2 東京都市大学メディア情報学部
Faculty of Informatics, Tokyo City University
3 大阪大学産業科学研究所
Institute of Scientific and Industrial Research, Osaka University

が 4 回」を，(1, 1)は「4 分音符が 1 回」であることを意味する．コードのリズム抽出方法も同様に行う．

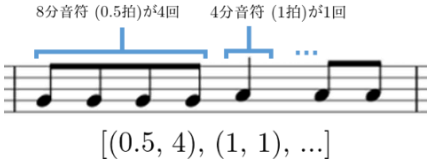

図 1　リズムの抽出方法

3. 提案手法

提案手法のモデル概要を図 2 に示す．提案するモデルは，以下の三つの生成器で構成される．

① リズム生成器

RhythmGenerator: コードリズムのシーケンスを学習する LSTM モデル．

RhythmTransformer: コードリズムからメロディのリズムを生成する Transformer [6]モデル．

② コード生成器

RhythmToChordGenerator: コードリズムとコードを元に次のコードを予測する LSTM モデル．

③メロディ生成器

ChordToMelodyGenerator: コードとメロディを元に次のメロディを予測する LSTM モデル．

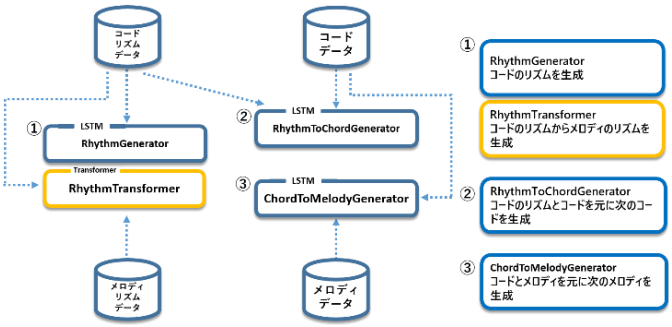

図 2　提案するモデル構造

データセットから抽出したコードリズムデータ，コードデータ，メロディリズムデータ，メロディデータを用いて，上記の三つの生成器により，順番に学習が行われる．

- RhythmGenerator

1)　データ表現方法

各種類のコードリズムを整数 ID に置き換え，リズムと ID のペアを辞書に保存する．Wikifonia データセットにおけるコードリズム ID の辞書サイズは 334 であった．各 ID は入力前に one-hot ベクトルでエンコードされる．入力ベクトルの次元数は，リズムの種類の数と同値である．

2)　モデル構造

RhythmGenerator の最初の層では，単語埋め込みを行い，リズム ID は実数のベクトルで表現される．単語埋め込みにより，各リズム間の関係を捉えることが可能である．リズム ID の one-hot ベクトル: x_{Rhythm} に埋め込み行列: W_{emb} を乗算し，30 次元の埋め込みベクトルを生成する．次に，埋め込みベクトルがサイズ 128 の LSTM 層に入力され，softmax 関数により，最終出力が得られる．

3)　モデルの学習

学習時には，損失関数として cross entropy loss を，最適化アルゴリズムとして Adam（学習率$lr = 2 * 10^{-4}$）を使用した．学習データ：テストデータ＝2,500 曲：500 曲とし，100　epoch 分学習した損失関数の結果を
図 3 に示す．

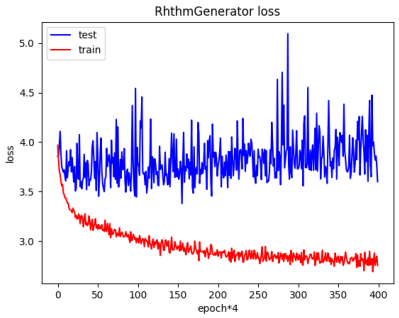

図 3　RhythmGenerator の損失関数グラフ

赤線は学習データ，青線はテストデータに対する損失関数のグラフを表している．図より，epoch 数を増やしても，テストデータの損失関数が減少しておらず，過学習を起こしていることが分かる．本モデルでは，リズム間の関係をとらえるために単語埋め込みを 1 層目に行っているが，そのために，テストデータに対しての予測に困難が生じていると考えられる．

4)　モデルの予測

新しいリズムを予測するために，初めに可変長のコードリズムのシードを RhythmGenerator に入力する．出力されたベクトルをサンプリングし，次のリズムが予測される．次に，予測されたコードリズムが RhythmGenerator に再び入力され，次のコードが再びサンプリングされる，という手順でコードリズムのシーケンスが生成される．

- RhythmTransformer

1)　データ表現方法とモデル構造

各種類のコードリズムとメロディリズムをトークンに置き換え，トークンと番号のペアを辞書に保存する．RhythmTransformer で学習させるコードリズムとメロディリズムの種類はそれぞれ 174 種類と 132 種類であった．

モデル構造に関して，本研究では [6]と同様のモデルを使用した．

2) モデルの学習

入力をコードリズムのシーケンス，出力をメロディリズムのシーケンスとして学習を行う．RhythmTransformer のパラメータは以下の様に設定した．

- エンコーダ，デコーダの層: 2 層
- モデルサイズ: 256 次元．
- 埋め込み層: 512 次元
- dropout の確率: 0.1
- バッチサイズ: 64
- 最適化手法Adamの初期学習率: 0.001

学習データ：テストデータ＝2,500 曲：500 曲とし，100 epoch 分学習した損失関数の結果を図 4 に示す．赤線が学習データ，青線がテストデータに対する損失関数を表している．図より，20 epoch まではうまく学習が進んでいるが，それ以降は過学習を起こしていることが分かる．これは音楽生成において，コードリズムとメロディリズムの組み合わせが多く，それらの特徴をとらえることが困難であるためだと考えられる．

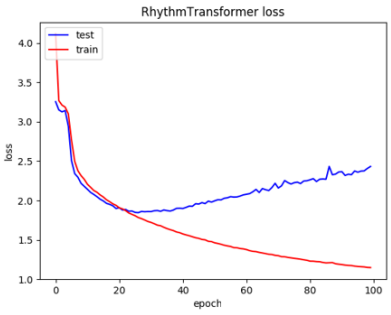

図 4 RhythmTransformer の損失関数グラフ

- RhythmToChordGenerator

1) モデル構造

図 5 に，時刻 t における RhythmToChordGenerator の入力ベクトル: x_{RtoC}^{t}と出力ベクトル: y_{RtoC}^{t}を示す．x_{RtoC}^{t}はコードの one-hot ベクトルと，RhythmGenerator で学習したリズムの埋め込みベクトルを合わせたベクトルである．こうすることにより，予測されるコードはリズムの特徴を考慮して生成される．出力ベクトルy_{RtoC}^{t}は，コードの数と同じ次元数を持ち，次のコードが予測される確率を示している．

$$\mathbf{x}_{RtoC}{}^{t} = \begin{pmatrix} 1 \\ \vdots \\ 0 \\ \vdots \\ 0.949 \\ \vdots \\ 3.721 \\ \vdots \end{pmatrix} \begin{matrix} \} Chord \\ \\ \\ \\ \} Rhythm \end{matrix} \qquad \mathbf{y}_{RtoC}{}^{t} = \begin{pmatrix} P(n_0 = 1 | x_{RtoC}^{0}, \ldots, x_{RtoC}^{t-1}) \\ \vdots \\ P(n_N = 1 | x_{RtoC}^{0}, \ldots, x_{RtoC}^{t-1}) \end{pmatrix}$$

図 5 時間tにおける RhythmToChordGenerator の入力ベクトルx_{RtoC}^{t}と出力ベクトルy_{RtoC}^{t}

モデルは 1 層の LSTM 構造である．初めにコードとコードリズムの情報を持ったベクトルx_{RtoC}^{t}を，サイズ 512 の LSTM 層に入力する．活性化関数には sigmoid 関数を使用し，最終出力としてy_{RtoC}^{t} が得られる．

2) モデルの学習

RhythmGenerator と同様の設定で，初期学習率を$lr = 10^{-7}$とし，100 epoch 学習した結果を図 6 に示す．赤線が学習データ，青線がテストデータに対する損失関数を表している．図より，うまく学習できていることが分かる．

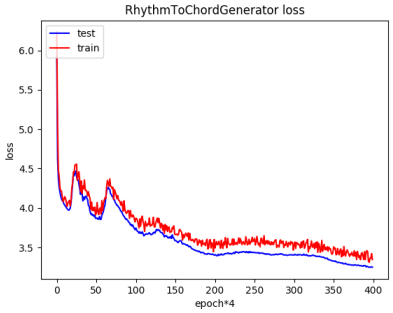

図 6 RhythmToChordGenerator の損失関数グラフ

- ChordToMelodyGenerator

1) 入力

入力ベクトルは，次元数 62 のメロディの one-hot ベクトルと，次元数 30 のコード埋め込みベクトルで構成され，出力はメロディが予測される確率である．モデル構造は RhythmToChordGenerator と同様である．

2) モデルの学習

RhythmToChordGenerator と同様の設定で学習した結果を図 7 に示す．赤線が学習データ，青線がテストデータに対する損失関数を表している．図より，正しく学習できていることが分かる．

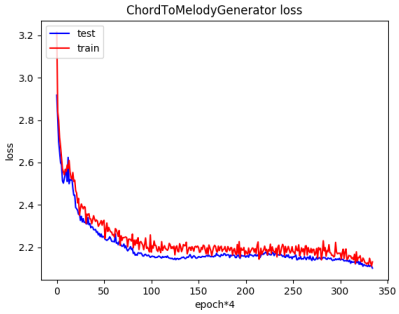

図 7 ChordToMelodyGenerator の損失関数グラフ

4. 音楽生成

学習したモデルを用いて，音楽を生成する． 図 8 に生成する際のモデル構造を示す．緑の丸で囲まれたコードリズム，コード，メロディはシードであり，それぞれ Wikifonia

データセットから無作為に抽出した 1 曲の最初の 8 小節分のものを使用する．学習時と同様に，以下の三段階で音楽生成が行われる．

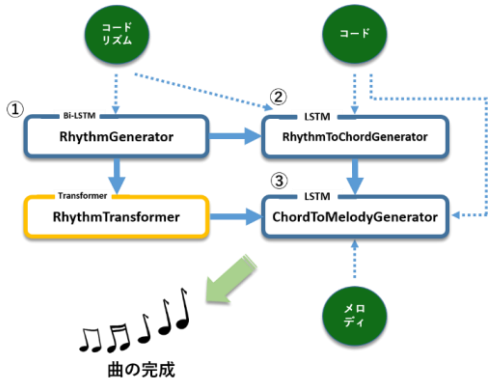

図 8　提案手法による音楽生成方法

① コードリズムのシードから，RhythmGenerator が 20 個の新しいコードリズムを生成する．それらを入力とし，RhythmTransformer がメロディのリズムを生成する．
② シードのコードリズムとコード，RhythmGenerator の出力を入力として，RhythmToChordGenerator がコードのシーケンスを生成する．
③ シードのコードとメロディ，RhythmToChordGenerator と RhythmTransformer の出力を入力として，ChordToMelodyGenerator がメロディのシーケンスを生成する．

生成されたリズム・コード・メロディを全て合わせた出力結果を図 9 に示す．

- コードリズム

初めの 4 小節は，2.5 拍→1.5 拍を組み合わせたリズムが続いている．しかし次の小節では，2.5 拍→0.5 拍→1 拍の様に変化していることが分かる．

- メロディリズム

コードリズムと違い様々な種類のリズムが生成されていることが分かる．

- コード

出力されたコードは 3 種類であった．初めの 2 小節は C コード，3 小節目では F コード→G コード，4 小節目で C コード→F コード，最後の小節で C コード，というコード進行になっている．各小節の始まりは全て C コードであり，小節の途中のリズムで他のコードが演奏されるため，聴きやすいコード進行となっている．

- メロディ

予測されたメロディのピッチは全 6 種類であった．各コードに対してメロディが調和しており，コードを考慮して生成が行われていることが分かる．

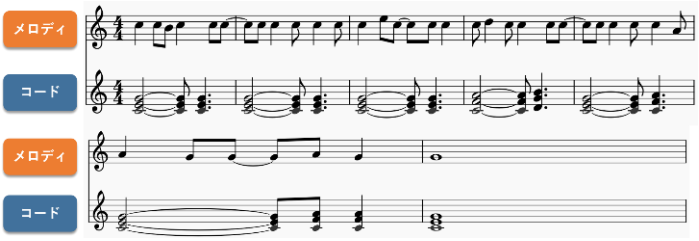

図 9　生成された音楽

5. おわりに

本研究では，リズム・コード・メロディを考慮して音楽を生成する三段階の深層学習モデルを提案した．本モデルでは，初めにリズムを生成し，そのリズムを考慮した上でコードを生成し，最後に，生成されたコードを元にメロディを生成する．実験では，リズム・コード・メロディが調和した音楽を生成することが出来た．今後の課題として，モデルのチューニング，評価方法の検討，進化計算アルゴリズムなどによる他音楽生成モデル [7]との比較を行う．

参考文献

[1] H. Dong, W. Hsiao, L. Yang and Y. Yang, "MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment," Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp.34-41, 2018.

[2] C. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu and D. Eck, "Music Transformer: Generating Music with Long-Term Structure," Proceedings of ICLR'2019, 2019.

[3] G. Brunner, Y. Wang, R. Wattenhofer and J. Wiesendanger, "JamBot: Music Theory Aware Chord Based Generation of Polyphonic Music with LSTMs," International Conference on Tools with Artificial Intelligence, pp.519-526, 2017.

[4] H. Lim, S. Rhyu and K. Lee, "Chord Generation from Symbolic Melody Using BLSTM Networks," Proceedings of ISMIR'2017, pp.621-627, 2017.

[5] M. S. Cuthbert and C. Ariza, "music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data," Proceedings of ISMIR'2010, pp.637-642, 2010.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, L. Kaiser, I. Polosukhin and A. N. Gomez, "Attention Is All You Need," Advances in neural information processing systems, 30:pp.5998-6008, 2017.

[7] N. Otani, D.Okabe and M. Numao, "Generating a Melody Based on Symbiotic Evolution for Musicians' Creative Activities," Proceedings of GECCO'2018, pp.197-204, 2018.