Team 16

Music Composition and Instrumental Performance with Deep Learning

DEMO is Here !

音楽とDeep Learningの邂逅

楽曲に対して**「こんなことといいな、できたらいいな！」**というニーズは昔から存在→ドラえもんのポケットは四次元。だが、それよりも遥かに高次元なDeep Learningがその手段として一助となり得るのではないか？という着想

例として以下が挙げられるが、何かDLの合わせ技ができないか？

- 曲から音声(ボーカル)を外したい、反対にボーカルだけ抽出したい！(ex.カラオケ練習)
- 特定の楽器を抽出したい！(ex.ギターソロをコピーしたい等、楽器の練習)
- ある楽曲を「○○風」にアレンジしたい！
  - アンパンマンのマーチをB'zに歌わせたらどうなる？…など
- 好きなアーティストAとBの曲を合体してみたい！
- 曲の中から一番美味しい箇所(サビなど)を探り当てたい！
  - ex.ストリーミング配信での試聴用音楽の生成
- 曲と曲との区切りを検出したい！
  - 無音部分の検出(頭出し)はカセットテープ時代から実在

手法案

2系統の音楽データ(演奏情報・波形)と分析アプローチ

1.既存曲演奏情報抽出 → 2.情報でモデル学習(MelodyRNN&MusicVAE) → 3.波形データで新楽器生成(GANSynth) → **作曲・編曲・演奏の全自動一気通貫を目指す！！**

	(1)演奏情報(MIDI(※)ファイル)	(2)波形データ(WAVE)
モデル	MelodyRNN(LSTMによる楽曲生成) / MusicVAE(潜在空間を学習したVAE)	GANSynth(GANを用いてAudioデータを合成する手法)
利点	「どの音がいつどのくらいの大きさ・長さで鳴るか」という情報の活用→楽曲情報が定量化されているため処理が容易	データが手に入り易い
欠点	そもそも当該MIDIデータを何らかの形で作成する必要(例えば、カラオケ配信会社が高品質なデータを大量に保有しているものと思料。)	学習・処理は重い(音楽CDで採用されているサンプリング周波数は44.1kHz。声波形を毎秒44,100回ずつ分割し、各時点の音声情報をデジタル情報にしたもの)

(※)「どの音がいつどのくらいの大きさ・長さで鳴るか」という情報を保持。MIDI(ミディ、Musical Instrument Digital Interface)は電子楽器の演奏データを機器間でデジタル転送するための世界共通規格。物理的な送受信回路・インタフェース、通信プロトコル、ファイルフォーマットなど複数の規定からなる。

例

参考例: Google Doodles 2019/3/21
ヨハン・セバスティアン・バッハを称えて

- 課題が追い込みを迎える中で、Googleの検索ロゴにバッハさんが登場。そして、「**あなただけの「バッハ調」のメロディを、AIを使ったDoodleで作曲しませんか？**」と我々の先を越される展開(?!)に
- https://www.google.com/doodles/celebrating-johann-sebastian-bach
  - 任意の2小節のメロディに対し、バッハの306曲を学習したハーモニー(単一の旋律要素のもとに、複数の声部が和声を構築する)を自動生成
  - 結果をMIDI形式で出力できる

「きらきらぼし」の和声構築

使用モデルの解説・実験

GANSynthについて

・Neural Synthesizerの一つ (ICLR 2019 waiting review)

・ギターとピアノの中間音を生成する、といったような音声合成ができる

・WavenetとGANの組み合わせ

・高速なサンプリングにより高効率な学習、合成が可能に

➡ Wavenetの50000倍ほどの速さ

・一つのglobalな潜在変数により一貫性のある特徴付けが可能に

➡ Wavenetではtime stepごとに特徴付ける必要があった

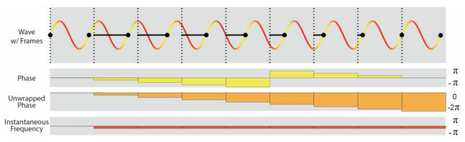
・音声波形の局所的な細かい特徴まで再現可能

➡ GANだけでは細かい特徴付けは難しい

GANSynthで用いられている工夫点

・Wavenetのように直接波形を生成するのではなく、log-magnitudeとphaseを生成することにより滑らかな音声合成が可能に

・phaseをそのまま用いるのではなくinstantaneous frequencyという特徴量を用いることで、より学習がしやすくなる



・STFTのフレームサイズやメル周波数のスケールを大きくすることで、低音域で重なりやすい倍音を分離することでき、パフォーマンスが向上

(論文：https://openreview.net/forum?id=H1xQVn09FX)

学習データ

- NSynthデータセット (https://magenta.tensorflow.org/datasets/nsynth)
  - 300,000曲の音程(pitch)など注釈が付いた楽器や声
  - その1つ1つのオーディオデータは、16kHzの周波数で4秒間に64000箇所サンプリングされたPCMのWaveフォーマット
- GANSynthでは
  - acousticの楽器かつ音程(24-84: 周波数32-1000Hz)のデータを使用
  - Waveデータとラベルに音程(pitch)を使用
  - Generatorの入力にはDiscriminatorの出力(z_noise)と音程をラベルとして渡し、Discriminatorの入力にはWaveデータを使用

ジャンル

Index	ID
0	acoustic
1	electronic
2	synthetic

GANSynth
今回の取り組み

楽器ごとのデータ数

Family	Acoustic	Electronic	Synthetic	Total
Bass	200	8,387	60,368	68,955
Brass	13,760	70	0	13,830
Flute	6,572	35	2,816	9,423
Guitar	13,343	16,805	5,275	35,423
Keyboard	8,508	42,645	3,838	54,991
Mallet	27,722	5,581	1,763	35,066
Organ	176	36,401	0	36,577
Reed	14,262	76	528	14,866
String	20,510	84	0	20,594
Synth Lead	0	0	5,501	5,501
Vocal	3,925	140	6,688	10,753
Total	108,978	110,224	86,777	305,979

本当は別のジャンル、例えばジャズや民族などの楽器のデータセットを自前で作って学習させたかったが、機械的に集める手段が見つからず断念。

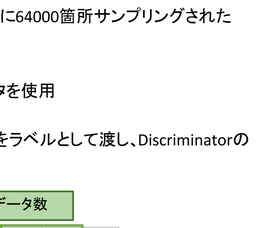
学習について

- 実装はgithubのmagentaに公開されているものを再利用
  - https://github.com/tensorflow/magenta/tree/master/magenta/models/gansynth
- 学習データは次の2パターン
  - 論文同様にNSynthのacoustic only, 音程 24-84
  - 少し変えてNSynthのelectronic + synthetic, 音程 24-84 (ソースコードdatasetを修正)
- ハイパーパラメータ
  - ベストパフォーマンスの設定を流用
    - Mel-Spectrograms (メル周波数スペクトグラム)
    - Progressive Training (Progressive GAN)
      - 学習経過の進行とともにアップサンプリングの解像度を高くして行く手法
      - Generator, Discriminator共にWasserstein Lossを使用
    - High Frequency Resolution
    - 1ステージ毎に800,000曲、12ステージまで解像度をあげながら学習して行く
    - 論文にはTesla V100 で3,4日とあるが、実際にはAWS p2.xlarge(Tesla K 80)で9,10日かかる見通し(資料を仕上げる時点でまだ学習が終わらず、8ステージ目)

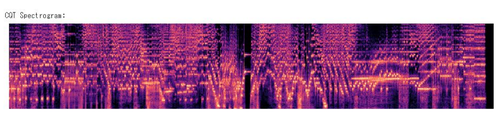
波形データの生成

- 入力はMidi音階
  - 各音符の潜在ベクトルを得る
  - この潜在ベクトルと音程をGeneratorに渡して合成する
  - 作曲した 米津玄師の「打ち上げ花火Lemon」を使用
    (以下の図はバッハの曲による生成結果)
- 様々な楽器を合成した波形データの生成結果

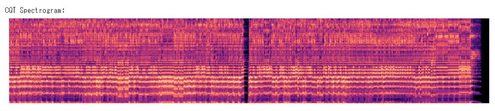
例：バッハの曲



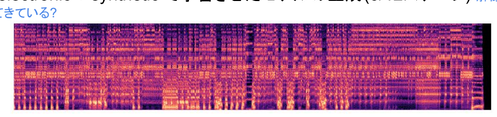
学習済みモデルでの生成 全ての楽器が正確に演奏され、滑らかに別の楽器へ変わっていく様子が魅力的な演奏に聞こえる



自前でacoustic で学習させたモデルの生成 (4/12ステージ) 同じ機械音にしか聞こえない…



自前でelectronic + synthetic で学習させたモデルの生成 (6/12ステージ) 解像度が上がるにつれ、少しましになってきている?



Melody RNN：モデル概要

- LSTMを用いた楽曲生成モデル
- 3つのオプション（Basic RNN, Lookback RNN, Attention RNN）からSpring Seminarでも扱ったAttentionを用いたRNNを選択
- AttentionはNeural Machine Translation by Jointly Learning to Align and Translate (D Bahdanau, K Cho, Y Bengio, 2014)の手法を採用している
- 元論文はエンコーダー・デコーダーモデルだが、本モデルでは、エンコーダー・デコーダーではなく、ある音を予測をする際に、前の nステップ分の音に常に注目するようにしている

$$u_i^t = v^T \tanh(W_1' h_i + W_2' c_t)$$
$$a_i^t = \mathrm{softmax}(u_i^t)$$
$$h_t' = \sum_{i=t-n}^{t-1} a_i^t h_i$$

v, W'1, W'2: ハイパーパラメータ

hi: 予測をする前のnステップ分の結果

ct: 現在のRNNのセル状態

ht'はRNNの予測結果と次の入力の双方に適用される

Melody RNN：学習と生成

モデル学習と楽曲生成のフロー

学習

学習用楽曲データ（.midi） → Notesequence（protocol buffers）（.tfrecord） → Training melody（.tfrecord） / Eval melody（.tfrecord） → 学習済みモデル（.ckpt）

生成

開始音（.midi または数値） → 学習済みモデル（.ckpt） → 生成された楽曲データ（.midi）

Melody RNN：データセット

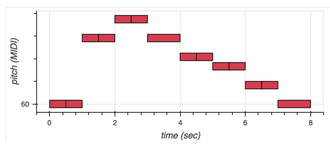
学習に用いたデータ

・米津玄師の楽曲9曲（Midi形式）（MuseSocreから取得）

・ビートルズの楽曲26曲（Midi形式）（Beatles MIDI filesから取得）

*MIDI/Notesequenceの楽曲はpitch（音程）、velocity（強弱）、start time and end time（音価）、tempos（テンポ）により表現される

```
# Add the notes to the sequence.
twinkle_twinkle.notes.add(pitch=60,
start_time=0.0, end_time=0.5, velocity=80)
....
twinkle_twinkle.total_time = 8
twinkle_twinkle.tempos.add(qpm=60);
```



https://colab.research.google.com/notebooks/magenta/hello_magenta/hello_magenta.ipynb

Melody RNN：学習と生成結果

学習時のパラメータ

- RNNのレイヤー数: [64, 64]
- Epoch数: 10,000回（米津玄師）／20,000回（ビートルズ）

生成時のパラメーター

- RNNのレイヤー数: [64, 64]
- ステップ数: 256 (16小節)
- 開始音: [60]（ド）

➡ 米津風、ビートルズ風のメロディの生成に成功！

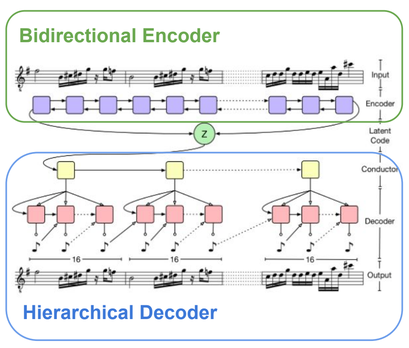
MusicVAE: A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music

概要
メロディシーケンス(16小節など)の潜在空間を学習したVAE
それっぽいメロディを一つの潜在変数から生成することができ、創作活動への応用が期待できる

このモデルでできること
Sample：潜在空間内のrandomな点(Latent Vector)から新たに曲を生成できる
Interpolate：複数のメロディをEncodeしてメロディ間の潜在変数を遷移させることでそれぞれのメロディ間をなめらかに補完するメロディを生成できる

入出力データ
Input/OutputともにMIDIファイル
Drumのみ、Melodyのみ、Trio(Melody, Bass, Dram)に対応している
2小節、16小節に対応している

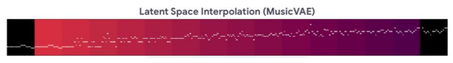
モデルの特徴
Bidirectional Encoder：Bidirectional LSTMをEncoderに用いることで双方向の時系列での情報をEncode
Hierarchical Decoder：小節単位で出力を行う階層型のDecoder、これにより16小節メロディの再生成誤差を劇的に改善している。



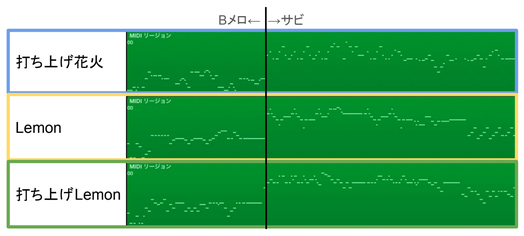
論文：https://arxiv.org/pdf/1803.05428.pdf
Blog：https://magenta.tensorflow.org/music-vae

挑戦1：既存のメロディから新しいメロディを合成する

提供されている学習済みの潜在空間上で同じアーティストの複数の曲をかけあわせることで、アーティストらしさやなにかしらの特徴をもった曲を生成できるはず



米津玄師の曲2つを合成して新しく曲を作る
使用した楽曲：打ち上げ花火、Lemon
楽曲を16小節の適切な範囲に切り出し、Bメロとサビの部分の位置を合わせた上で合成した



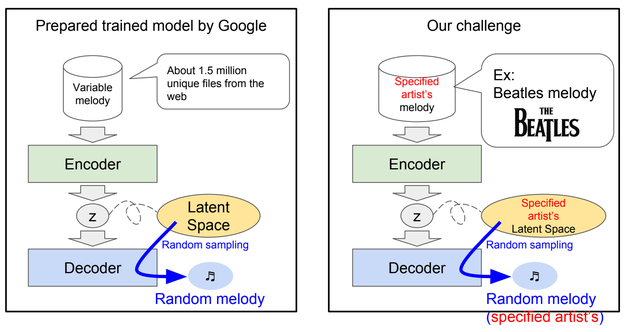
こちらのQRコードから実際のメロディが試聴できます

サビ部分は1秒～

新しいメロディを生成することができた！
Bメロからサビの盛り上がりも表現できていた。

挑戦2：アーティストのそれっぽいメロディを無限生成する

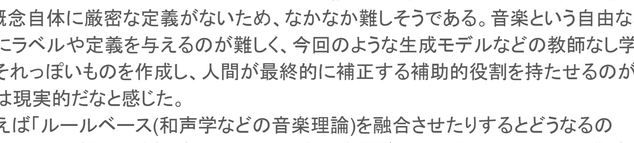
特定のアーティストについて独自データセットを用いて潜在空間を学習することで、ランダムサンプリングにてそのアーティストっぽいメロディの生成ができるはず！



結果と考察

使用したデータセット(MIDIファイル)：米津玄師の楽曲(9曲)、Beatlesの楽曲(14曲),東方の楽曲(52曲)

- 学習
  うまく学習してくれていそう(batch_size=32,learning_rate=0.0005)



- 生成 (タイトル横のQRコードにて実際の音を聞くことができます)
  - 米津玄師モデルで生成したメロディ (MusicVAE RandomSample Yonezu 1 or 2)
  - Beatlesモデルで生成したメロディ(MusicVAE RandomSample Beatles)
  - 東方モデルで生成したメロディ(MusicVAE Touhou)

それぞれについて、それっぽいメロディ？を生成することができた！

一部、元データのMelodyそのままのような箇所 (元の曲を切って貼った感じ)があるが、その Melodyに入る、もしくは終わる部分について、自然な印象を受ける Melodyとなっている。

まとめ

- 作曲という観点では、波形よりも音階情報を含むmidiをデータセットに軍配が上がる(**MelodyRNN,MusicVAE**)が、2曲間補完という観点では波形のほうがより特徴を平滑化できる(**GANSynth**)。ただし、当然学習時間というトレードオフがある。
- 一曲まるごと生成といったような手法はまだまだ発展途上で、これといった打ち手がない。補完や部分生成は潜在特徴の空間ベクトルから、中間表現をいか抽出/推論生成するかがキモとなっている(**GANSynth,MelodyRNN,MusicVAE**)
- 音源のスタイル変換手法として波形情報を持つwavの方が自由度も特徴量も格段に増え、表現力が高い(**GANSynth,NSynth**)
- この世に存在しない音源を作出することも可能なため、AIを全く新しい楽器として活用期待が持てる(**GANSynth,NSynth**)
- 一つのプロジェクトやサービスとして考えた場合や上記のようにそれぞれの特徴に強みのある複数のタスクを組み合わせて、**パートごとの作曲生成**(MelodyRNN)→**midiを波形に変換して独自音源にスタイル変換**(GANSynth)でオーケストラやバンドに→**生成したメロディの前後情報を補完**して1曲,アルバムの作成(MusicVAE)のような**1アルバムまるごとAIが生成**みたいなことは今後近うちに出てくるであろう。
- 今回メロディの生成には成功した。しかし、生成単位が最大で16小節なので、曲という観点だと繰り返しや盛り上がり(いわゆるBメロ/サビ)などの構成を学習する必要がある。実際にInterpolateでBメロ/サビの盛り上がりが作れたため、サビのメロディを多数エンコードして潜在空間上にMappingできればある程度サビっぽいメロディを抽出することができると思われる。
- MusicVAEとMelodyRNNを比較すると(データセットが少なく過学習している可能性もあるが)MusicVAEのほうが学習データの切り貼り感が強く感じられた。

プロジェクトを通じての所感

- 楽曲生成や曲間補完は実際のユースケースが見えやすい。例えばオープンワールドのゲームなどで街やフィールドの状態を、既存では音楽の切り替わりのフェードで対応しているが、GANSynthを利用すればフィールド間の曲調をシームレスに変換したり、アーティストのアルバムも複数の曲を結合し、連続した一つの作品として昇華できるようになるなど、アイディアにいとまがない
- 音楽データの奥深さを感じた。色々アイデアが出ても、midiデータの収集は容易ではなく、wav→midiの変換は難易度が高い等の制約があり、実現できなかった。機械学習の発展とともに、データが扱いやすくなれば、全人類作曲家・DJの時代がくるはず。
- GANSynthはNSynthという学習しやすいように加工されたデータセットの使用を前提としていて、データの変更による拡張は難しく感じた。今後、画像分野と同様に扱えるデータの制約が小さくなると当初の目的である「○○風の作曲」というようなものが実現に近づくのではないかと今後への期待が高まった。
- 曲の構成(Bメロ/サビ、繰り返しなど)を学習できるとメロディをいい感じに曲にしてくれるようなモデルが作れそう。そのために曲の構成を定義する必要があるが、サビという概念自体に厳密な定義がないため、なかなか難しそうである。音楽という自由な表現にラベルや定義を与えるのが難しく、今回のような生成モデルなどの教師なし学習でそれっぽいものを作成し、人間が最終的に補正する補助的役割を持たせるのがまずは現実的だなと感じた。
- 例えば「ルールベース(和声学などの音楽理論)を融合させたりするとどうなるのか？」などの新たな着想が湧いた。質の良い演奏データ(MIDIファイル)の準備が学習の鍵であるので、演奏データ作成の自動化も重要。これらも含めて、音楽分野での適用可能性の広がりを感じられた点が有意義であった。