ニューラルネットワークによる
音楽の自動変換

教養学部後期課程学際科学科総合情報学コース

陶山大輝

学籍番号：08-192021

指導教官：金子知適准教授

目次

第 1 章

初めに

本研究ではニューラルネットワークを用いて音楽を変換することを目標とする。変換は音楽における remix と呼ばれる手法と同様に行う。remix とは既存の曲の再構成及び加工を行ってその曲の新しいバージョンを作成する方法のことである。また、本研究では、remix を音楽の構造を変化させない音色の変換による加工に限定する。そして、音色の変換のみを扱うことで短期的な構造のみに着目できる点で他の音楽生成の研究よりも計算時間を削減できると期待される [1]。

第 2 章

音楽用語の定義

本章では音楽用語の定義及びその説明を行う。

2.1　音

音とは、弾性体 (空気) 中を伝播する弾性波により起こされる音波が聴覚により感じられるもののことである。また、音波に周期性があり明確な音程を持つ音として聞こえる場合は楽音と呼ばれる。

2.2　楽音の三要素

楽音は高さ, 大きさ, 音色の三つの要素 (音の三要素) から成り立ち、人間はこれらを知覚することができる。また、本論文では楽音のことを音と呼ぶ。

2.2.1　音の高さ

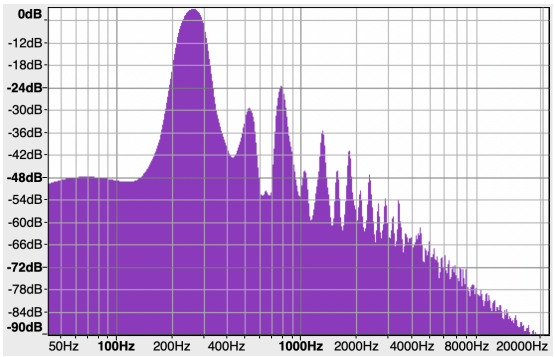

図 2.1　音波の周波数スペクトル

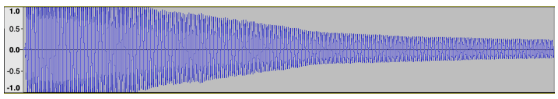

図 2.2　音波の波形

音の高さは音波の周波数により決まる。人間には周波数が高い音は高く、周波数が低い音は低く知覚さる。また、複数の周波数の音波が音に含まれる場合は最も低い周波数成分の音波 (基音) を音の高さとして知覚する。

図 2.1 は図 2.2 で示される音波にフーリエ変換を行うことで求められる周波数スペクトルである。周波数ス

ペクトルはそれぞれの周波数成分がどれだけその音波に含まれているかを示すので、この音波の基音は 264Hz であることがわかる。

2.2.2　音の大きさ

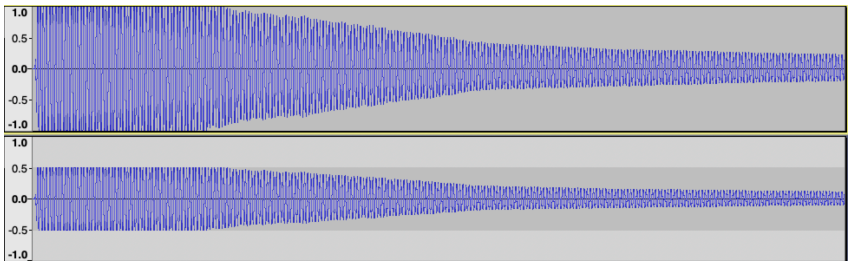

図 2.3　音の大きさの異なる音波

音の大きさは音波の振幅により決まる。人間には振幅が大きい音は大きく、振幅が小さい音は小さく知覚さる。

図 2.3 では、同じ楽器から出る同じ高さの音の音波を示しており、振幅の大きい後者の方が大きい音として人間には知覚される。

2.2.3　音の音色

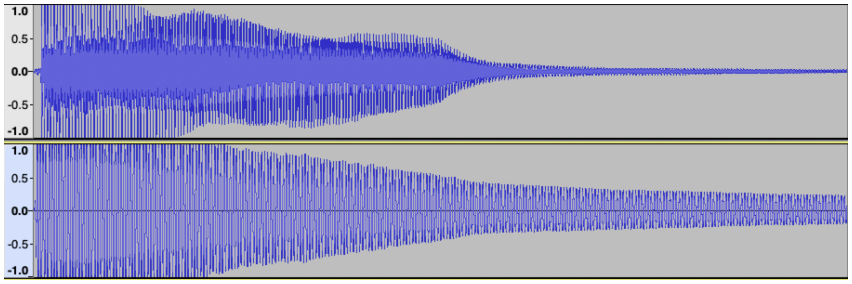

図 2.4　ギターとハープの音色

音の高さと大きさが同じであっても異なった音として人間には知覚される。この違いを音色と呼ぶ。

図 2.4 では、上側はギターの音波, 下側はハープの音波で同じ高さかつ同じ大きさである。この音波の波形の違いが音色の違いを作り出す。

第 3 章

背景

本章では、Multilayer perceptron 及びその応用例の Generative Adversarial Networks の説明を行った後、Generative Adversarial Networks を画像の変換に応用した Pix2pix を紹介する。

3.1 Multilayer perceptron

Multilayer perceptron (MLP) は複数のアフィン変換と非線形関数からなる関数により定義され、式 3.1 で定式化される。

$$F_{MLP}(\boldsymbol{x}) = f_n(W_n(f_{n-1}(W_{n-1}\cdots(f_1(W_1(\boldsymbol{x})+\boldsymbol{b_1}))\cdots+\boldsymbol{b_{n-1}}))+\boldsymbol{b_n}) \tag{3.1}$$

ここで、$\boldsymbol{x}, \boldsymbol{y}$ は実数ベクトル,n は層の総数,f_i は i 層目の非線形関数,W_i は i 層目の行列,b_i は i 層目の定数ベクトルである。

MLP を用いると、あるデータ集合 $D = \{(\boldsymbol{x_j}, \boldsymbol{t_j}); 1 \leqq j \leqq N\}$ について、それぞれの j で $\boldsymbol{t_j}$ の予測値として $\boldsymbol{y_j} = F_{MLP}(\boldsymbol{x_j})$ を出力することができる。

また、$\boldsymbol{t_j}$ に近い $\boldsymbol{y_j}$ を出力するにはそれぞれの i で W_i を適切に決める必要がある。この時、損失関数 $L(\boldsymbol{y_j}, \boldsymbol{t_j})$ を定義し、$W_i \leftarrow W_i - \eta\frac{dL}{dW_i}$ として更新すると、$\boldsymbol{y_j}$ を $\boldsymbol{t_j}$ に近づける方向に W_i を更新することができる (勾配降下法) 。

そして、損失関数としては、平均二乗誤差 $L_{MSE} = \frac{1}{n}\sum_{j=1}^{n}(\boldsymbol{y_j}-\boldsymbol{t_j})^2$ や平均絶対誤差 $L_{MAE} = \frac{1}{n}\sum_{j=1}^{n}|\boldsymbol{y_j}-\boldsymbol{t_j}|$ などが用いられる。

3.2 Generative Adversarial Networks

Generative Adversarial Networks (GAN) [2] は MLP の応用例であり学習データの特徴を学習して擬似的なデータを生成することを目指す。この手法は自然な手書きの文字を出力する際などに用いられる。

GAN は生成モデルと識別モデルと呼ばれる二つの MLP で構成される。生成モデルは学習データに近いデータを出力し、識別モデルはデータが生成モデルの出力と学習データのどちらであるかを識別し、学習データである確率を出力する。

これらの二つの MLP はランダムに初期化された後に競合的に学習を行う。これにより、生成モデルが学習データにより近いデータを生成できるようになると期待される。

また、生成モデルの目的関数は式 3.2, 識別モデルの目的関数は式 3.3 として定式化される。

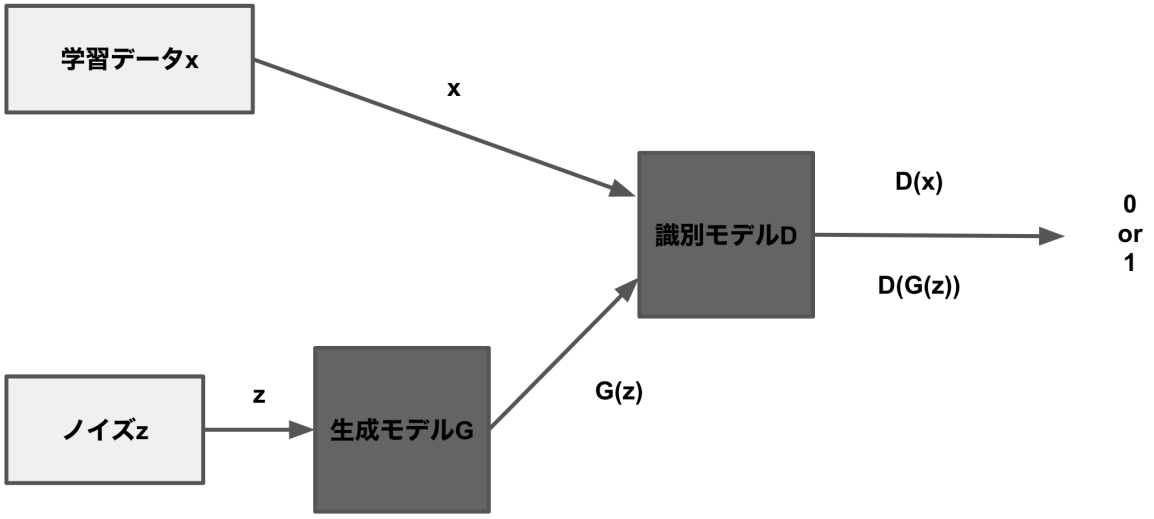

図 3.1　GAN のネットワーク

$$\arg\min_{\theta_G} \mathbb{E}_{\boldsymbol{z}}[\log(1 - D(G(\boldsymbol{z};\theta_G);\theta_D))] \tag{3.2}$$

$$\arg\max_{\theta_D} \mathbb{E}_{\boldsymbol{x}}[\log D(\boldsymbol{x};\theta_D)] + \mathbb{E}_{\boldsymbol{z}}[\log(1 - D(G(\boldsymbol{z};\theta_G);\theta_D))] \tag{3.3}$$

ここで、$\boldsymbol{x}$ は学習データ,$\boldsymbol{z}$ は生成モデルへの入力のノイズ,$G(\boldsymbol{z};\theta_G)$ はノイズ $\boldsymbol{z}$ を入力とする生成モデル,$D(\cdot;\theta_D)$ は識別モデル,θ_G は生成モデル G のパラメータ,θ_D は識別モデル D のパラメータである。

3.3　Pix2pix

Pix2pix [3] はある条件下で画像間の変換を行う GAN である。例えば、図 3.3 のようにピクセルの対応関係を変えずにスタイル変換を行うことができる。

また、Pix2pix は GAN に変換先の学習データを条件として与えることでスタイル変換を行う。生成モデルの目的関数は式 3.4, 識別モデルの目的関数は式 3.5 でとして定式化される。

$$\arg\min_{\theta_G} \mathbb{E}_{\boldsymbol{y},\boldsymbol{z}}[\log(1 - D(\boldsymbol{y}, G(\boldsymbol{y},\boldsymbol{z};\theta_G);\theta_D))] + \mathbb{E}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}[\|\boldsymbol{x} - G(\boldsymbol{y},\boldsymbol{z};\theta_G)\|_1] \tag{3.4}$$

$$\arg\max_{\theta_D} \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}[\log D(\boldsymbol{x},\boldsymbol{y};\theta_D)] + \mathbb{E}_{\boldsymbol{y},\boldsymbol{z}}[\log(1 - D(\boldsymbol{y}, G(\boldsymbol{y},\boldsymbol{z};\theta_G);\theta_D))] \tag{3.5}$$

ここで、$\boldsymbol{x}$ は変換元の学習データ,$\boldsymbol{y}$ は変換先の学習データ,$\boldsymbol{z}$ は生成モデルへの入力のノイズ,$G(\boldsymbol{y},\boldsymbol{z};\theta_G)$ は $\boldsymbol{y}$ を条件としノイズ $\boldsymbol{z}$ を入力とする生成モデル,$D(\boldsymbol{y},\cdot;\theta_D)$ は $\boldsymbol{y}$ を条件とする識別モデル,θ_G は生成モデル G のパラメータ,θ_D は識別モデル D のパラメータである。

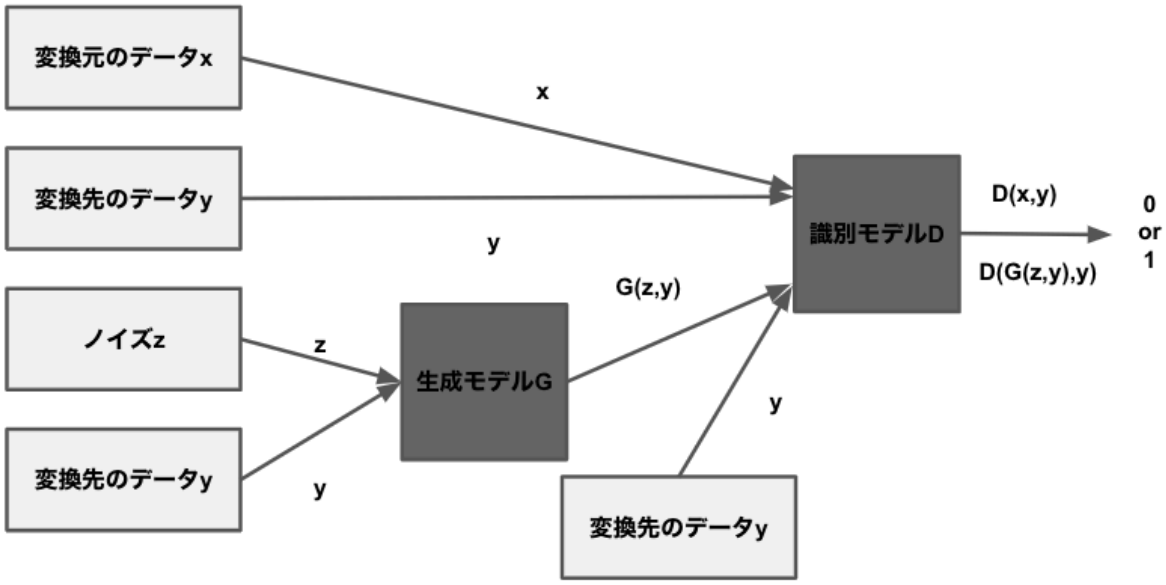

図 3.2　pix2pix のネットワーク

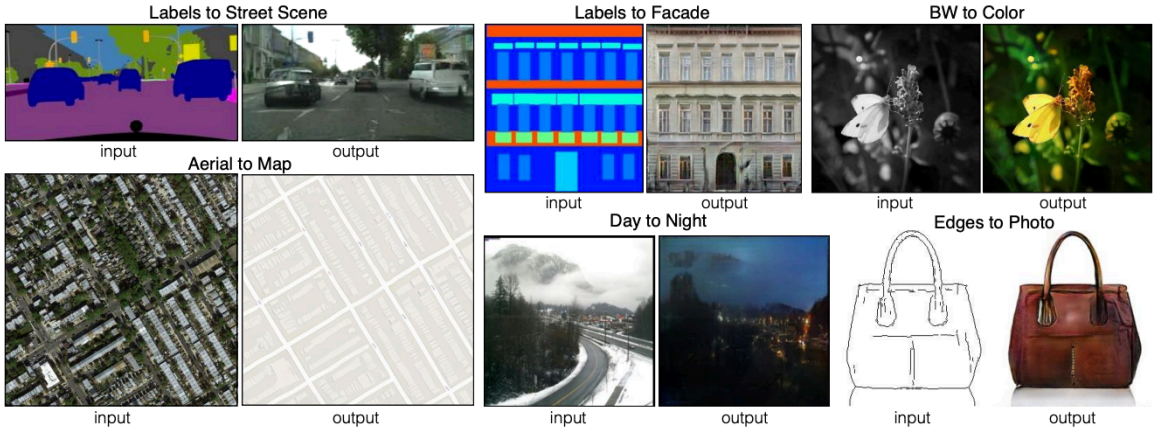

図 3.3　pix2pix のスタイル変換の例

3.3.1　生成モデルの構造

変換を行うための生成モデルには Encoder-Decoder のネットワークが用いられる。Pix2pix の生成モデルでは、画像データの基礎的な構造を保持するために、図 3.4 のように U-net[4] で用いられるスキップコネクションを持ったネットワークが用いられる。

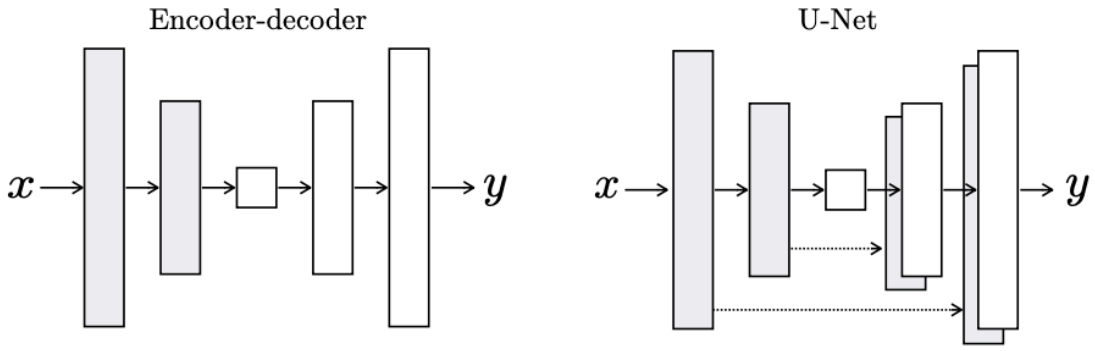

図 3.4　U-net のネットワーク

3.3.2　識別モデルの構造

Pix2pix の識別モデルでは、PatchGAN という手法が用いられる。PatchGAN は画像全体の誤差を求めるのではなくパッチと呼ばれる小領域ごとで誤差を求めて平均を取っている。これにより、局所的な部分の識別の精度が高まることが期待される。

第 4 章

実験

本章では、実験で使用したデータセットの説明を行った後に実験の結果及び考察をまとめる。

4.1 データセット

本研究のデータセットの作成方法及びその形式についての説明を行う。

4.1.1 データセットの作成方法

楽譜作成ソフトの MuseScore[*1]により国際の階名表記で A0 から C8 までの半音を wav 形式で 88 音生成した。これらは 88 鍵のピアノで出すことのできる音であり、最も一般的な音域として今回の実験では選んだ。

4.1.2 データ形式

音のファイル形式としては非圧縮形式の WAV を用いる。MP3 や MP4 などの非可逆圧縮形式も一般には広く用いられるが、音波の波形データを直接保持しているために扱いやすい WAV を本論文で用いることにした。また、WAV は波形データ以外にメタデータを持ち、本論文で扱うメタデータについて以下で説明をする。

サンプリング周波数
: サンプリング周波数とは、デジタル信号の 1 秒あたりの標本化の回数のことである。本論文では 44100Hz に固定して実験を行う。

サンプリング数
: サンプリング数とは、デジタル信号の標本化の合計の回数のことである。本論文では 44100 回に固定して実験を行う。

量子化ビット数
: 量子化ビット数とは、デジタル信号の細かさを表現するビット数のことである。本論文では 16bit に固定して実験を行う。

チャンネル数
: チャンネル数とは、モノラルな音声の出力の総数のことである。本論文では 1 に固定して実験を行う。

[*1] https://musescore.org/

4.2　実験環境

Pix2pix を用いてギターからハープへの音の変換を行うことを目標に実験を行った。ギターとハープの組は十分に音色が異なる楽器として選んだ。また、学習とテストの際のパラメータは A の A.1 に示す。

4.3　生成モデルの表現力

生成モデルの表現力を測るために学習データとテストデータに同じ 88 音を用いて実験を行った。また、過学習を防ぐために毎エポックのそれぞれのデータの振幅を $c \in [0.3, 1]$ 倍して学習を行った。

4.3.1　loss のグラフ

4.3.2　考察

本実験では波形の観察を通して考察を行う。

4.4　生成モデルの汎化能力

生成モデルの表現力が十分にあることを確認したので、その汎化能力を調べる実験を行った。また、先程の実験と同様に毎エポックのそれぞれのデータの振幅を $c \in [0.3, 1]$ 倍して学習を行った。

さらに、汎化能力を調べるために、88 音のデータに対し 4 分割交差検証を行った。この際にデータはランダムに分割しており、その区分は A の A.2 に示した。

4.4.1　loss のグラフ

4.4.2　考察

本実験では波形の観察を通して考察を行う。

第 5 章

まとめ

5.1 future work

5.1.1 音楽の変換

音色の変換を音楽で行う際に (音の特徴を学習する) が、以下の三つの点を解決するのが難しいと考えられる。また、以下の三つを解決することで、単音における音色の変換を音楽に適用することができる。

楽器の重ね合わせ

楽器ごとに音色が異なるので、楽器ごとの音波に分解して音色変換を行うことが良いと考えられる。なお、楽曲の作成時に楽器ごとに分離したデータ (パラデータ) で保存しておけば、直接楽器ごとの音波を利用できる。

時間方向の音の繋ぎ方

時間方向での音の繋ぎ方は都合の良いように分割していくことでなんとかなるのではないか…？、分割する (1 つの音の判定を行う) のは難しい…？、自己回帰？

section

音の重ね合わせ

単位時間の楽器の音に注目した時、楽器ごとの音波に分離したとしても和音のようにその単位時間で複数の種類の音が鳴っている場合も難しいと考えられる。

とりあえず試しでも良いので実験を行いたい

5.1.2 判定器

(1) 音程が維持されているか

(2) 変換されて音色が変換されているか

波形とスペクトログラム？

5.1.3 和音の手法

二音のみの組み合わせでできるのかネットワークを変えるべきかネットワークをアップデートする必要があるのか前処理を工夫するかフーリエ変換を用いて sin 波に分解するか

謝辞

本研究を進める際に丁寧なご指導をして頂いた指導教官の金子知適准教授と中屋敷太一先輩に厚く感謝を申し上げます。また、研究に関して助言を頂いた金子研の構成員の方々にも感謝の意を表します。

参考文献

[1] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. 2020.

[2] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[3] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

付録 A

A.1　実験時のパラメータ

実験時のパラメータを表 A.1 に示す。

表 A.1

パラメータ	値
インプットのバッチサイズ	1
学習でのエポック数	1000
学習での学習率	0.0002

A.2　データセットの区分

生成モデルの汎化能力を調べた際のデータセットの区分を表 A.2 に示す。

表 A.2

番号																					
0	a5s	g2	e6	f6	g3	c1	a2s	f7	f5	c2s	b5	e4	b6	d4s	a4	b7	e1	g4s	f4	a7s	g1	f2s
1	a2	d1	g1s	g4	d5s	d6	a3s	a1	a5	c6s	f1s	c1s	a1s	c5	d2s	g7	c2	e7	b2	g7s	f5s	d1s
2	e3	g5	c7s	d4	g5s	d6s	g6	c8	c4	c4s	a0s	a3	d3	d5	b0	a6	a6s	f7s	c5s	d3s	f3	d7s
3	b4	f3s	f6s	e2	b1	e5	f2	c3	a4s	f1	c6	a7	d2	g6s	g2s	g3s	c3s	f4s	a0	d7	b3	c7