

ニューラルネットワークによる 音色の自動変換

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	はじめに	3
第 2 章	背景：音楽	4
2.1	音	4
2.2	音の四要素	4
2.2.1	音の長さ	5
2.2.2	音の大きさ	5
2.2.3	音の高さ	5
2.2.4	音の音色	5
2.3	音の表現	6
第 3 章	背景：ニューラルネットワーク	7
3.1	教師あり学習	7
3.1.1	教師あり学習の目的	7
3.1.2	教師あり学習モデルの学習と汎化	7
3.2	MLP	8
3.2.1	ニューラルネットワーク	8
3.2.2	人工ニューロンの構造と定式化	8
3.2.3	MLP の構造と定式化	9
3.2.4	MLP の学習	10
3.3	CNN	11
3.3.1	畠み込み層	11
3.3.2	プーリング層	12
3.3.3	CNN の利点	12
3.4	GAN	13
3.4.1	GAN の学習	13
3.4.2	GAN の定式化	13
3.5	Pix2pix	14
3.5.1	Pix2pix の定式化	15
3.5.2	Pix2pix の生成モデル	15
3.5.3	Pix2pix の識別モデル	16
第 4 章	提案モデル	17

4.1	音の表現	17
4.2	ネットワーク構造	18
4.2.1	生成モデル	18
4.2.2	識別モデル	18
第 5 章	実験	20
5.1	データセット	20
5.2	評価方法	20
5.3	実験時の工夫	20
5.4	実験方法	20
5.4.1	生成モデルの表現力の評価	21
5.4.2	生成モデルの汎化能力の評価	21
5.5	実験結果	21
5.5.1	生成モデルの表現力の評価	22
5.5.2	生成モデルの汎化能力	24
第 6 章	まとめと展望	27
6.1	まとめ	27
6.2	展望	27
6.2.1	楽器の重ね合わせ	27
6.2.2	音の重ね合わせ	27
6.2.3	音の繋ぎ方	28
参考文献		30
付録 A		31
A.1	実験時のパラメータ	31
A.2	データセットの分割	31

第1章

はじめに

音楽は世界中で楽しまれ、その作成方法は技術発展により多様化している。近年注目されている音楽の作成方法としてリミックスと呼ばれるものがある。リミックスは既存の曲に音響操作を加えて新しい曲を作成する方法のことであり、1970年代のディスコの発達とともに世界的に普及した。当時はディスコでのDJによる即興のパフォーマンスにより行われていたが、近年のパソコンなどのデジタル機器の発達によって音楽の作成方法として一般的なものとなった。

しかし、録音や音の編集を行うソフトウェアアプリケーション(DAW)の操作がリミックスには必要であるため、音楽作成を円滑に行うためには一定の経験が必要となる。したがって、コンピュータプログラムによるリミックスの補助が音楽作成に役立つと考えられる。また、本研究では、リミックスの方法の一つである音色の変換に注目し、ニューラルネットワークによる変換手法を提案する。

さらに、音色の変換を行うためには、ある楽器の音を異なる楽器の音へ変換する技術が必要である。本研究では、画像のスタイル変換を行うPix2pix [1]を音色の変換に応用した。Pix2pixは、ニューラルネットワークにより自然な画像を生成する手法であるGAN [2]を応用した手法である。

そして、本研究では、ギターの単音をハープの単音へと提案モデルを用いて変換する実験を行った。その結果、ほとんどの音で音程を維持したまま音色の変換を行うことに成功した。また、データセットの一部の音のみで学習を行った場合でも、ほとんどの音で音程を維持したまま変換を行うことができた。

なお、本論文では、楽譜作成ソフトのMuseScore^{*1}を用いて音のデータセットを作成し、音声編集ソフトのAudacity^{*2}を用いて音波の波形の画像を取得した。

^{*1} <https://musescore.org/>

^{*2} <https://www.audacityteam.org/>

第 2 章

背景：音楽

本章では、音楽用語の定義を行った後に、音のデータの主要な前処理の方法を紹介する。

2.1 音

音とは、弾性体（空気）中を伝播する弹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、音は騒音と楽音の大きく二つに分けることができる。騒音とは不規則な振動の音波による音のことであり、楽音とは周期性のある音波による音のことである。本論文では、楽音のことを音と呼ぶ。

2.2 音の四要素

音は長さ、大きさ、高さ、音色の四つの要素から成り立ち、人間はこれらの四つの要素を知覚することができる。また、図 2.1 は四要素について簡単に示した図である。

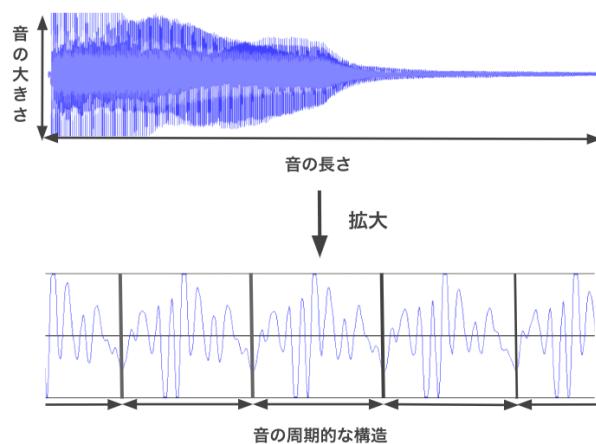


図 2.1 音の四要素

2.2.1 音の長さ

音の長さは図 2.1 のように音波の時間の長さにより決まる。一般に、音の長さは楽譜上での時間の長さ(音価)により決まるが、本論文では音波の時間の長さにより決まるものとする。

2.2.2 音の大きさ

音の大きさは図 2.1 のように音波の振幅により決まる。また、人間には振幅の大きい音は大きく、振幅の小さい音は小さく知覚される。

2.2.3 音の高さ

音の高さは音波の周波数により決まる。つまり、図 2.1 のような音の周期的な構造の長さにより決まる。また、人間には周波数の高い音は高く、周波数の低い音は低く知覚する。そして、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。

国際の音名表記では、C,D,E,F,G,A,B のセットを西洋音楽の七音音階におけるオクターブとして定める。さらに、それぞれのオクターブに番号を振り、440 Hz の音を A4 と定めることで、任意の半音の絶対的な表記を可能にしている。また、国際の音名表記では半音よりさらに細かい音(微分音)を表すことはできないが、本論文では扱わない。

2.2.4 音の音色

音の長さと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とがある。この違いを音色と呼ぶ。また、音色は図 2.1 のような音の周期的な構造の形により決まる。

ここで、長さと高さと大きさが同じギターとハープの音波を図 2.2 に示す。これらの音は波形が異なるので、その音色も異なる。また、音色の異なる音どうしは基音よりも高い音(上音)の組み合わせが異なるため、このような波形の違いが生まれる。

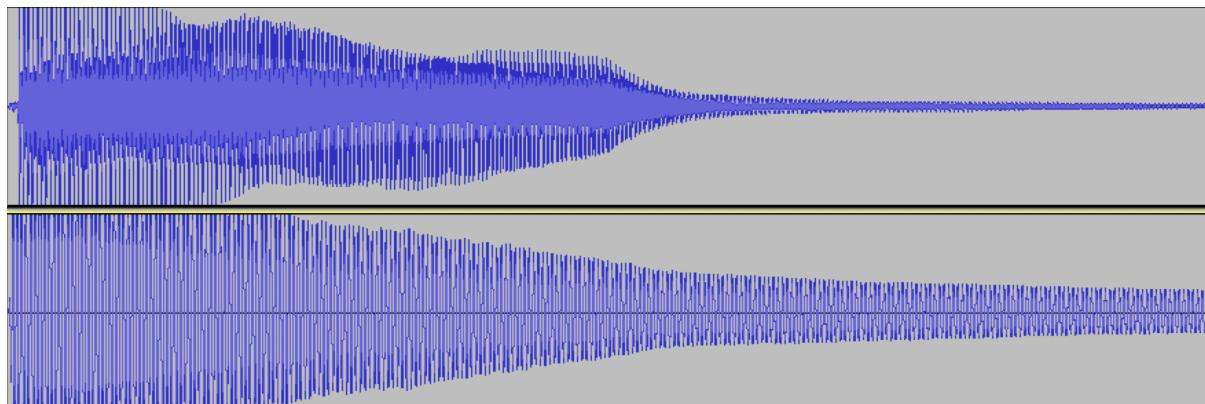


図 2.2 ギターとハープの音色

2.3 音の表現

第3章

背景：ニューラルネットワーク

本章では、教師あり学習と MLP の説明を行った後、MLP の応用例である CNN 及び GAN を紹介する。そして、これらの応用例を組み合わせた Pix2pix を紹介する。

3.1 教師あり学習

教師あり学習は機械学習の手法の一つである。機械学習とは、学習データと呼ばれるデータに含まれる特徴をコンピュータプログラム（モデル）が自動で学習し、学習したモデルを用いて何らかの問題を解く手法のことである。また、教師あり学習では、説明変数と対応するべき目的変数のペアとして学習データが与えられる。

3.1.1 教師あり学習の目的

X, Y をそれぞれ説明変数と目的変数の空間とすると、 $f : X \rightarrow Y$ のうち任意の $\mathbf{x} \in X$ について正しい値を出力する関数 f' を表現するモデルを作成することが教師あり学習の目的である。また、 $\mathbf{x} \in X$ について、 $f(\mathbf{x})$ の $f'(\mathbf{x})$ への近似の程度を評価する関数を損失関数 L と呼ぶ。よって、損失関数は \mathbb{R}^+ を非負の実数として $L : Y \times Y \rightarrow \mathbb{R}^+$ と定義でき、値が小さいほど近似の程度が良い。

3.1.2 教師あり学習モデルの学習と汎化

任意の $\mathbf{x} \in X$ と対応する $\mathbf{y} \in Y$ の組を用意することは現実的には難しいため、学習データはランダムに抽出されているという仮定のもとにある。この時、学習データに含まれる任意の説明変数 \mathbf{x} について損失関数 L の値の期待値を小さくするように学習を行う。すなわち、目的関数は式 3.1 となる。

$$\hat{f} = \arg \min_f \mathbb{E}[L(\mathbf{y}, f(\mathbf{x}))] \quad (3.1)$$

また、 \hat{f} はモデルの学習結果であるが、学習データに含まれる説明変数のみで最適であるため、 \hat{f} は f' に一致するとは限らない。したがって、学習データとは別に評価データを用意し、モデルの評価データに対する性能（汎化性能）を測定する必要がある。

3.2 MLP

MLP (Multilayer perceptron) はニューラルネットワークの一つであり、教師あり学習の手法として用いられる。

3.2.1 ニューラルネットワーク

ニューラルネットワークは、神経細胞と神経細胞間のシナプス結合を通る電気信号により実現される脳の機能に類似した数理モデルである。ニューラルネットワークでは、神経細胞を人工ニューロンとして表現し、シナプス結合の強度を人工ニューロン間の重みとして表現する。

3.2.2 人工ニューロンの構造と定式化

MLPにおける人工ニューロンは図3.1のように m 個の入力 x_i に対応した重み w_i をかけた重み付き和に対して活性関数 θ を作らせた出力 y を返す。つまり、式3.2として定式化される。

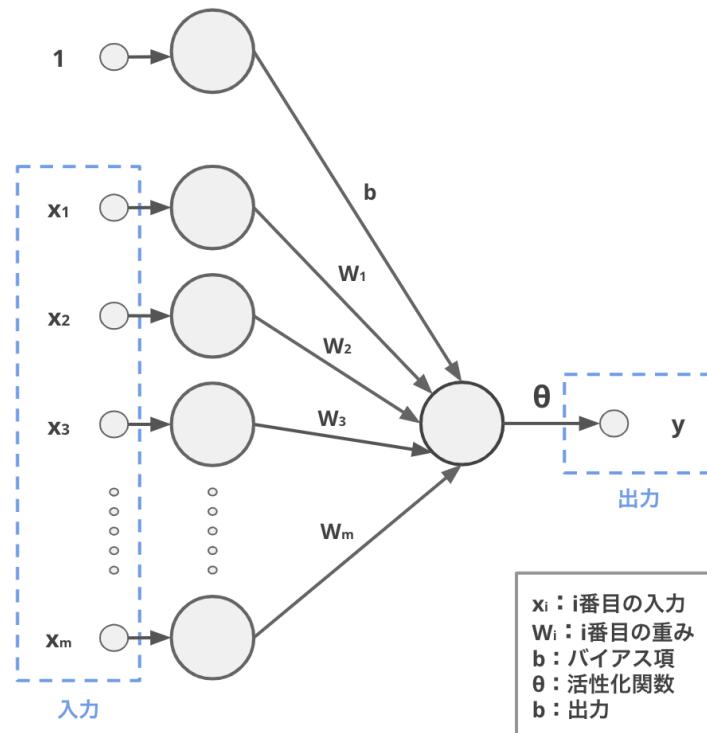


図3.1 人工ニューロン

$$\mathbf{y} = \theta\left(\sum_{i=1}^m W_i \times \mathbf{x}_i\right) \quad (3.2)$$

また、活性化関数としては出力値を調整するために適当な関数が選ばれる。多くの場合は式 3.3 の ReLU 関数や式 3.4 の Sigmoid 関数などの非線形関数が用いられる。

$$\theta_{ReLU}(x) = \max(0, x) \quad (3.3)$$

$$\theta_{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (3.4)$$

3.2.3 MLP の構造と定式化

MLP の構成単位は、3.2.2 節の人工ニューロンが複数並んだ層である（図 3.2）。この層の出力 \mathbf{y} は式 3.2 より式 3.5 として導出できる。また、 $\mathbf{x} \mapsto W\mathbf{x} + \mathbf{b}$ はアフィン変換と呼ばれ、 θ は活性化関数である。

$$\mathbf{y} = \theta(W\mathbf{x} + \mathbf{b}) \quad (3.5)$$

ここで、 \mathbf{x} は層の入力となる実ベクトル、 \mathbf{y} は層の出力となる実ベクトル、 \mathbf{b} は第 i 成分が i 番目の出力に対応したバイアス項となる実ベクトル、 W は (i, j) 成分が i 番目の人工ニューロンと j 番目の出力に対応した重みとなる実ベクトル、である。

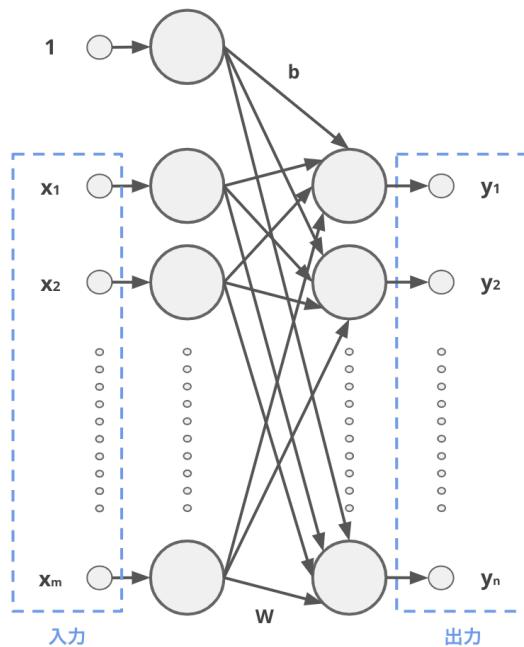


図 3.2 MLP の一層

MLP は、図 3.2 を一層とした順伝播型の階層構造のニューラルネットワークである（図 3.3）。一層ずつの入力層と出力層、その間の中間層から構成される。層を構成する人工ニューロンの出力は次の層の全ての人工ニューロンに渡されるため、MLP を構成する層を全結合層とも呼ぶ。また、 n を層数とおけば、MLP は式 3.6 として定式化される。

$$\hat{f}(\mathbf{x}) = \theta_n(W_n(\theta_{n-1}(W_{n-1} \cdots (\theta_1(W_1\mathbf{x} + \mathbf{b}_1)) \cdots + \mathbf{b}_{n-1})) + \mathbf{b}_n) \quad (3.6)$$

ここで、 \mathbf{x} は実ベクトル、 θ_i は i 番目の活性化関数、 W_i は i 番目の行列、 \mathbf{b}_i は i 番目の実ベクトル、である。

3.2.4 MLP の学習

MLP では、学習により任意の i で W_i, \mathbf{b}_i を適切に定める必要がある。また、教師あり学習では損失関数を減少させる方向に学習は進むため、それぞれ式 3.7 と 3.8 にしたがって W_i, \mathbf{b}_i は更新される（勾配降下法）。

$$W_i \leftarrow W_i - \eta \frac{dL}{dW_i} \quad (3.7)$$

$$\mathbf{b}_i \leftarrow \mathbf{b}_i - \eta \frac{dL}{dW_i} \quad (3.8)$$

ここで、 η は任意の正の実数であり、学習率と呼ばれる。なお、損失関数としては式 3.9 の平均二乗誤差 L_{MSE} や式 3.10 の平均絶対誤差 L_{MAE} などが用いられる。また、学習率を適切に設定する最適化アルゴリズム

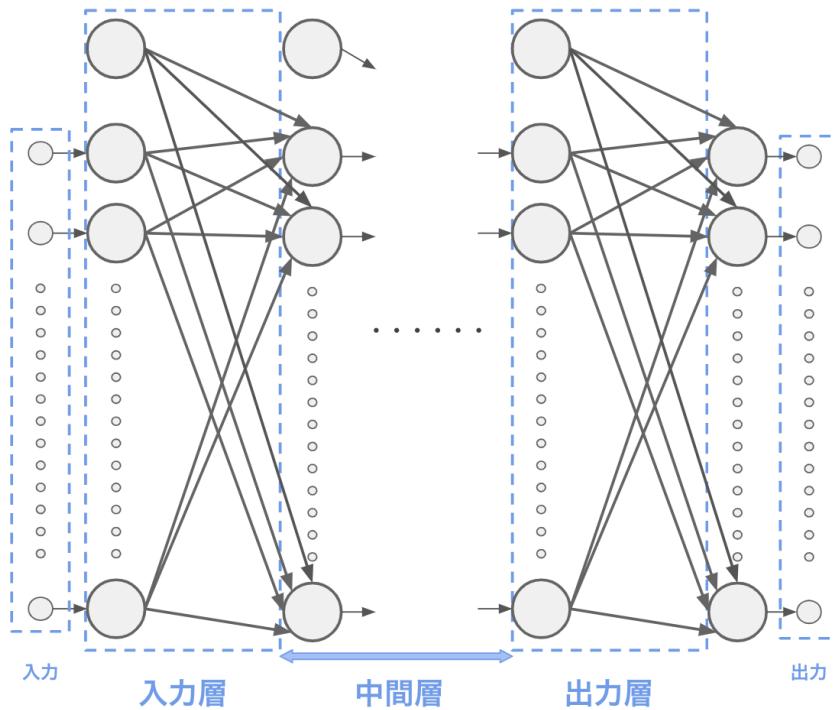


図 3.3 MLP のネットワーク

ムとして、Stochastic Gradient Descent [3] や Adam [4] などが用いられる。

$$L_{MSE} = \frac{1}{n} \sum_j (\mathbf{y}_j - \hat{f}(\mathbf{x}_j))^2 \quad (3.9)$$

$$L_{MAE} = \frac{1}{n} \sum_j |\mathbf{y}_j - \hat{f}(\mathbf{x}_j)| \quad (3.10)$$

3.3 CNN

CNN (Convolution Neural Network) は主に画像認識の分野で成果を残している代表的な順伝播型ニューラルネットワークである [5]。CNN のネットワーク構造は MLP において全結合層を脳の視覚野の構造を模倣するように置換したものとみなせる。また、脳の視覚野では、刺激の位置により異なったニューロンの活動電位が発生し、受容する情報の特徴によって受容野が階層構造を形成している [6]。つまり、CNN は、人工ニューロン間の結合が局所領域に限定され、層ごとに受容する情報が異なるニューラルネットワークである。具体的には、画像の局所領域の特微量を抽出する畳み込み層と局所領域の特微量をまとめるプーリング層（サブサンプリング層）を繰り返した構造である [7]。また、以下では画像（二次元データ）で CNN の説明を行うが、他の次元のデータについても同様の説明を行うことができる。

3.3.1 畳み込み層

畳み込み層においては、図 3.4 のようにカーネルと呼ばれるフィルターを用いて畳み込み演算を行う。畳み込み演算ではカーネルのそれぞれの値を重みと見た重み付き和の計算が行われる。また、カーネルは予め決めた幅（ストライド）によって移動させる。そして、そのまま畳み込み演算を行った場合は特微量マップが元の画像より小さいサイズとなるため、一般には元の画像の周りに適当な幅で値を埋める操作を行う（パディング）

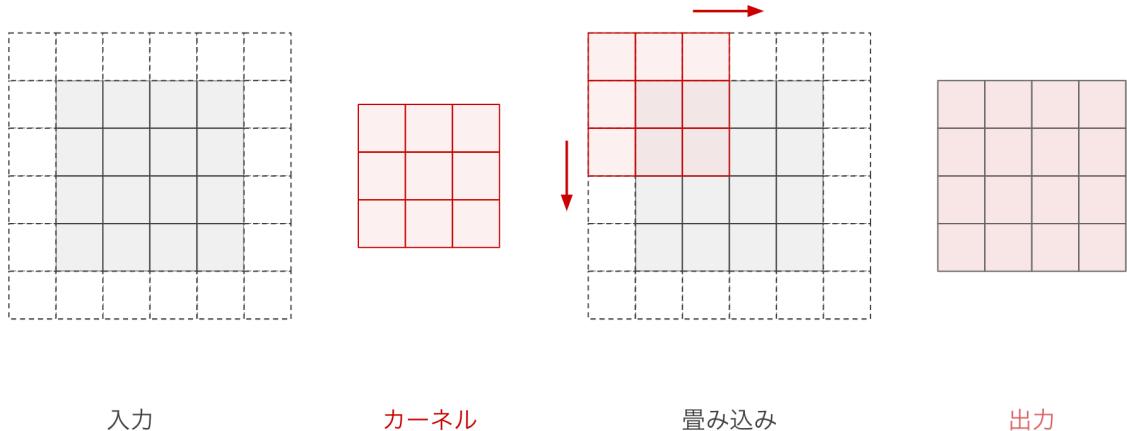


図 3.4 畳み込み層

グ)。また、カーネルサイズを ks 、ストライド幅を st 、パディング数を pad 、入力サイズを in 、出力サイズを out 、とした時、式 3.11 が成り立つ。

(3.11)

加えて、異なるカーネルが複数あっても良く、カーネルが複数あるときは出力する特微量マップの数はその数になる。入力のチャンネル数が 1 でない場合もそれぞれのチャンネルで求めた畳み込み演算の値を足したものが返される。

3.3.2 プーリング層

プーリング層においては、図 3.5 のように情報量を圧縮する（サブサンプリング、ダウンサンプリング）。入力を局所領域ごとに分解し、その局所領域ごとに何らかの操作を行う。この操作が最大値をとる場合は Max Pooling、平均値をとる場合は Average Pooling と呼び、主にこの二つがプーリング層としては用いられる。

3.3.3 CNN の利点

CNN には MLP と比べて主に二つの利点がある [7]。一つ目は、パラメータ数が減少する点である。全結合層の場合は入力の数と出力の数の積が層のパラメータ数となるが、畳み込み層においてはカーネルに含まれるパラメータ数のみが層のパラメータ数となるため、パラメータ数が減少する。また、プーリング層においてもサブサンプリングにより特微量マップのサイズが小さくなるためパラメータ数の減少に寄与する。二つ目は、様々な解像度での特徴を取得できる点である。プーリング層によりサブサンプリングすることで解像度が低下し、異なるスケールでの隣接する特徴の共起を得ることができると期待される。

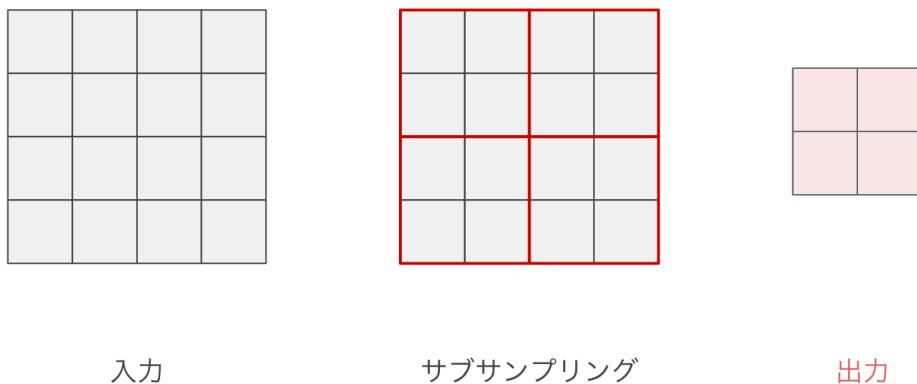


図 3.5 プーリング層

3.4 GAN

GAN (Generative Adversarial Networks) [2] はニューラルネットワークの応用例であり、学習データの特徴を持つ擬似的なデータを生成することを目指す手法である。この手法は実在しないアイドルの写真を生成する際などに用いられる [8]。また、GAN は図 3.6 のように二つのニューラルネットワークで構成され、それぞれのネットワークは Discriminator (識別モデル) と Generator (生成モデル) と呼ばれる。

3.4.1 GAN の学習

生成モデルと識別モデルの二つのネットワークはランダムに初期化された後に競合的に学習を進める。まず、識別モデルはデータが Fake data (生成モデルの出力) と Real data (学習データ) のどちらであるかを識別できるように学習を進める。そして、生成モデルは識別モデルが学習データであると誤って識別するよう、Noise (ノイズ) を元に学習データに近いデータを出力する。この二つの学習を交互に繰り返すことで、漸進的に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、ノイズは適当な次元の実ベクトルであり、生成モデルの出力の揺らぎを表現する潜在変数の役割を果たす。

3.4.2 GAN の定式化

GAN では、生成モデルの目的関数は式 3.12、識別モデルの目的関数は式 3.13 として定式化される。

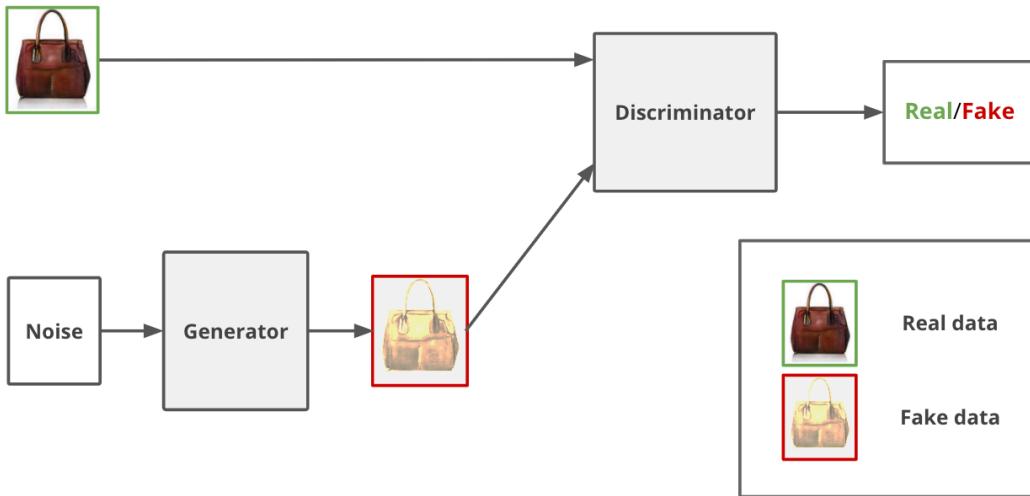


図 3.6 GAN のネットワーク
図は文献 [1] の Figure 1 を用いて作成した。

$$\arg \min_{\theta_G} \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}; \theta_G); \theta_D))] \quad (3.12)$$

$$\arg \max_{\theta_D} \mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x}; \theta_D)] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}; \theta_G); \theta_D))] \quad (3.13)$$

ここで、 \mathbf{x} は学習データ、 \mathbf{z} は生成モデルへの入力のノイズ、 $G(\mathbf{z}; \theta_G)$ はノイズ \mathbf{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。

3.5 Pix2pix

Pix2pix [1] は、図 3.7 のようにネットワークの入力に変換元の画像を Condition (条件) として与えることで画像の変換を行う GAN である。特定の条件をネットワークの入力に与える GAN としては Conditional GAN (CGAN) [9] が初めて考案されたが、Pix2pix は与えられた条件画像の構造を維持したまま変換するという点で CGAN とは異なる。具体的には、図 3.8 のように線画から写真への変換や白黒画像からカラー画像への変換を行うことができる。

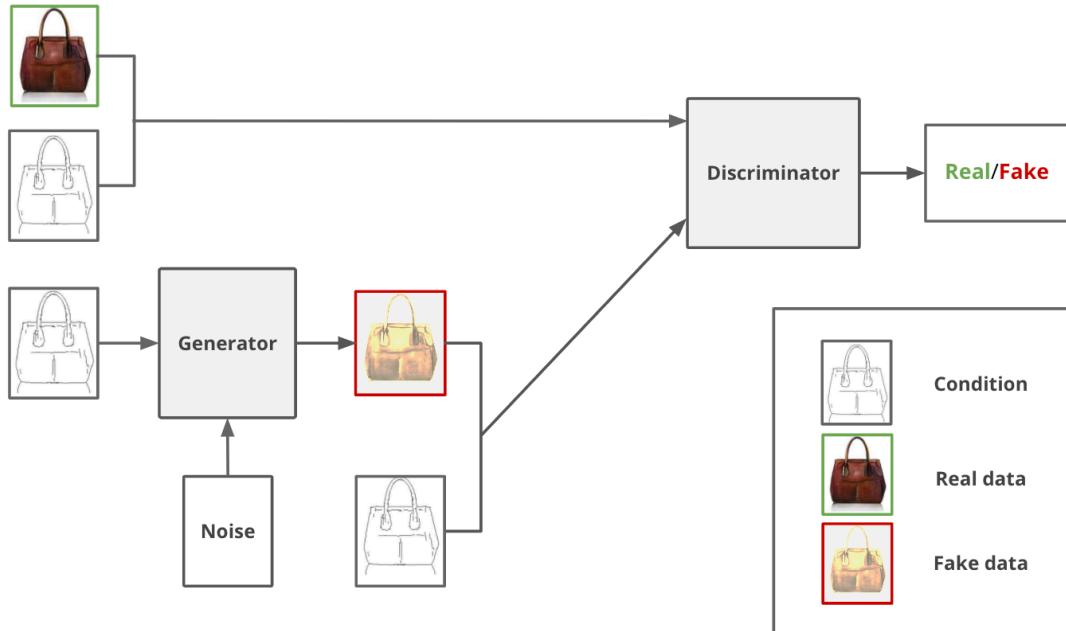


図 3.7 Pix2pix のネットワーク
図は文献 [1] の Figure 1 を用いて作成した。



図 3.8 Pix2pix のスタイル変換の例
図は文献 [1] の Figure 1 を用いて作成した。

3.5.1 Pix2pix の定式化

そして、Pix2pixにおいては、生成モデルの目的関数は式 3.14、識別モデルの目的関数は式 3.15 として定式化される。

$$\arg \min_{\theta_G} \mathbb{E}_{y,z} [\log(1 - D(y, G(y, z; \theta_G); \theta_D))] + \mathbb{E}_{x,y,z} [\|x - G(y, z; \theta_G)\|_1] \quad (3.14)$$

$$\arg \max_{\theta_D} \mathbb{E}_{x,y} [\log D(x, y; \theta_D)] + \mathbb{E}_{y,z} [\log(1 - D(y, G(y, z; \theta_G); \theta_D))] \quad (3.15)$$

ここで、 x は変換先の学習データ、 y は変換元の学習データ、 z は生成モデルへの入力のノイズ、 $G(y, z; \theta_G)$ は y を条件としノイズ z を入力とする生成モデル、 $D(y, \cdot; \theta_D)$ は 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。

3.5.2 Pix2pix の生成モデル

Pix2pix の生成モデルには、図 3.9 のように Encoder-Decoder 型のネットワークが用いられる。ただし、変換元の画像と返還後の画像で共通する基本構造であるピクセルの対応関係を維持するために、スキップコネクションが用いられる。このスキップコネクションは U-net [10] で用いられたものと同様の働きをする。具体的には、エンコード前の特微量マップをデコード時にも利用しており、この特微量マップによりピクセルの対応関係が維持されると期待される。

また、ノイズとしては実ベクトルではなく Dropout [11] が用いられる。Dropout とは、ニューラルネットワークの重みの更新の際にランダムにいくつかの重みを 0 として無視することである。

3.5.3 Pix2pix の識別モデル

Pix2pix の識別モデルには、PatchGAN という手法が用いられる。PatchGAN は図 3.10 のように画像全体ではなくパッチと呼ばれる局所領域ごとに真偽を求めて平均を出力とする。これにより、局所的な識別精度が高まることが期待される。

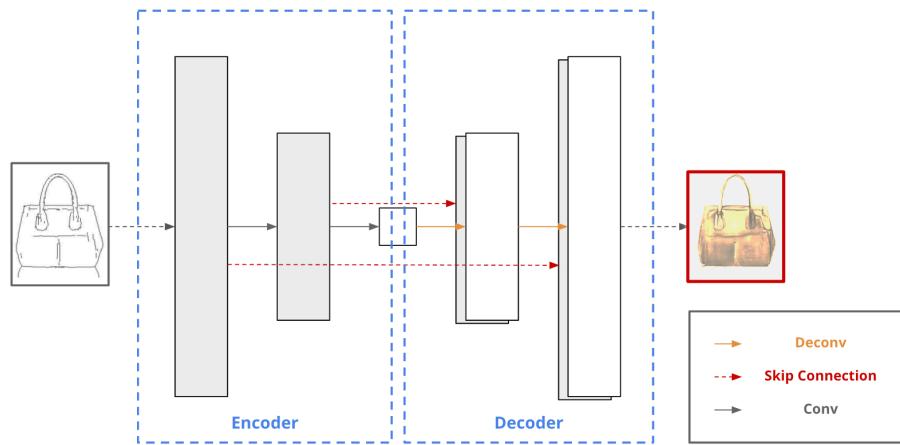


図 3.9 生成モデルのネットワーク
図は文献 [1] の Figure 1 と Figure 3 を用いて作成した。



図 3.10 通常の GAN と PatchGAN の比較
図は文献 [1] の Figure 1 を用いて作成した。

第4章

提案モデル

本研究では、図 4.1 のような Pix2pix を元に作成したモデルを提案する [1]。このモデルでは、生成モデルと識別のいずれにも条件として変換元の音を入力している。また、決定論的に音を生成するために、生成モデルの入力にノイズを使用していない。

4.1 音の表現

本研究では、44100 Hz のサンプリング周波数でサンプリングした 1 秒の音波をそのまま使用する。また、量子化ビット数を 16 ビット、チャンネル数を 1 として固定しているため、44100 の長さを持つ 16 ビット整数の一次元配列として音を扱う。

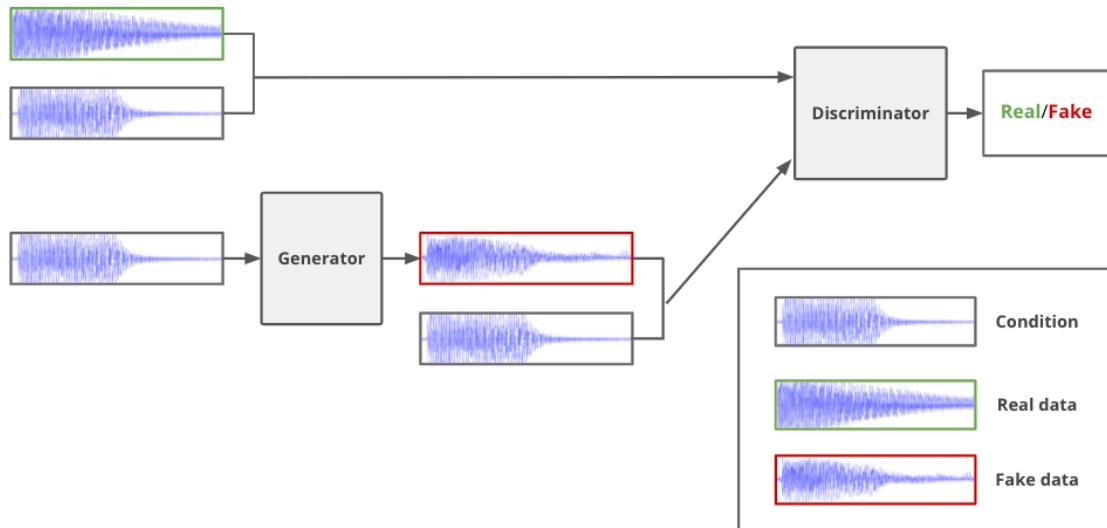


図 4.1 提案モデル

4.2 ネットワーク構造

図 4.2 及び図 4.3 に生成モデルと識別モデルのネットワーク構造をそれぞれ示す。まず、図にある灰色の箱は複数チャンネルの特徴量マップである。箱の上側にチャンネル数を示し、箱の左側に特徴量マップとなる一次元配列の長さを示している。そして、ネットワーク構造の下にはそれぞれの矢印の操作を示している。Convolution と Deconvolution については (カーネルサイズ、パディング数、ストライド幅) としてそれぞれの値を示し、LeakyReLU については負の実数の定義域での一次関数の傾きの値を示す。

4.2.1 生成モデル

本研究では、生成モデルに図 4.2 のような 1 つのスキップ接続を持つ Encoder-Decoder 型のネットワークを用いた。それぞれのパラメータは図 4.2 に示す。

4.2.2 識別モデル

本研究では、識別モデルに 4.3 のような CNN を用いた。それぞれのパラメータは図 4.3 に示す。

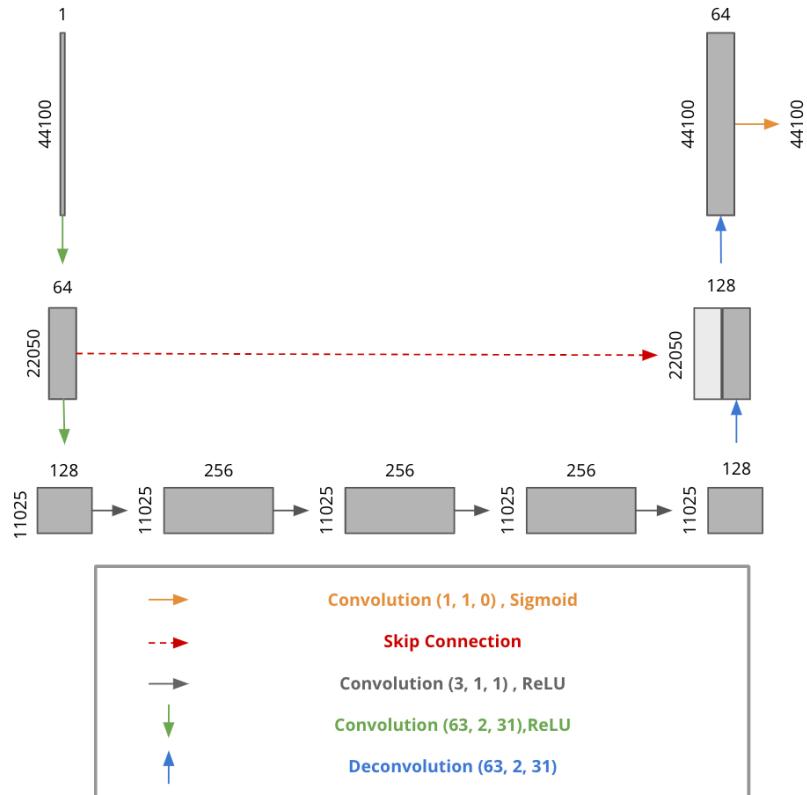


図 4.2 生成モデル
図は文献 [10] の Figure 1 を参考に作成。

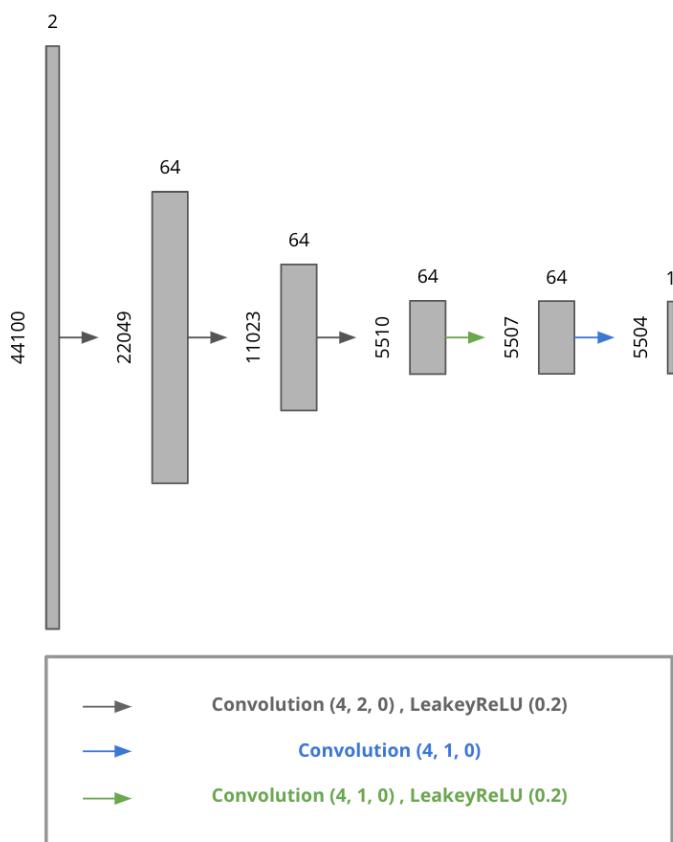


図 4.3 識別モデル
図は文献 [10] の Figure 1 を参考に作成。

第 5 章

実験

本章では、実験で用いるデータセットや実験方法などについて説明した後、実験結果の考察を行う。

5.1 データセット

1 秒のギターとハープの音の 88 組をデータセットとして用意した。また、データ形式としては非圧縮形式で波形を保持する WAV 形式を採用した。88 音としては A0～C8 の半音を全て選んだが、これらは図 5.1 のような一般的な 88 鍵のピアノで用いられる音である。

5.2 評価方法

生成した音は音波の波形の観察と音の聴き取りにより評価した。また、以降で示す波形の図は三つの音波を並べたものであり、変換元のギター、生成モデルの出力、変換先のハープ、の順で並べている。そして、生成モデルの出力が音の高さ及び音の大きさを保持したまま変換先のハープの音の音色に変換できているかという観点から評価を行った。

5.3 実験時の工夫

実験時には学習データの振幅を無作為化する工夫を行った。具体的には、学習データの振幅を 0.3～1.0 倍した。また、学習前に乱数のシードを固定した後に一様乱数により生成している。この工夫によりデータを拡張することができ、88 音と小さいサイズのデータセットへの過学習を防ぐことができていると期待した。

5.4 実験方法

提案モデルを用いて十分に音色が異なると考えられるギターからハープへの音の変換の実験を行った。学習と評価の際のパラメータは付録 A の A.1 節に示す。また、本研究で行うモデルの学習には Adam を用いた。

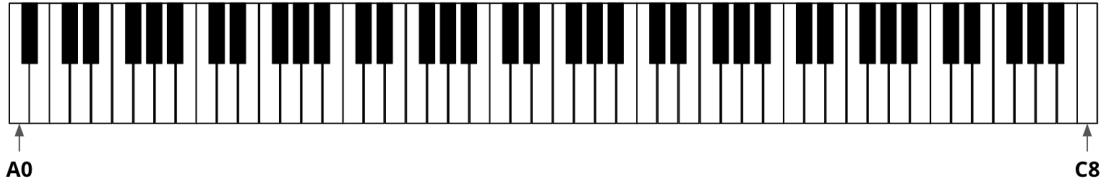


図 5.1 88 鍵のピアノの鍵盤

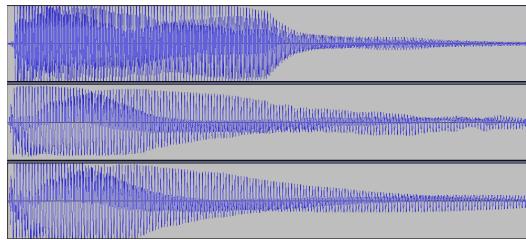


図 5.2 F3 の 0.800 秒から 1.000 秒までの音波

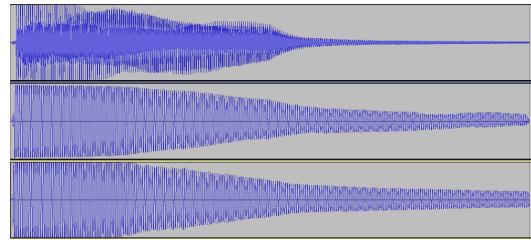


図 5.3 C4 の 0.200 秒から 0.300 秒までの音波

5.4.1 生成モデルの表現力の評価

まず、本手法のニューラルネットワークの生成モデルが十分な表現力を有し、識別モデルが正確に判定を行うことができるかを確認するため、学習データと評価データにそれぞれ同じ 88 音のデータセットを用いて実験を行う。

5.4.2 生成モデルの汎化能力の評価

一般に、単音の音色変換において任意の高さと大きさの音を学習データとして用意するのは不可能である。従って、未知のデータも適切に処理できるニューラルネットワークを構築する必要がある。このような、未知のデータを適切に処理できる能力のことを汎化能力と呼ぶ。

本手法のニューラルネットワークが未知のデータも適切に処理できるか調べるために、88 音のうち $3/4$ を学習データ、 $1/4$ を評価データとする 4 分割交差検定を行う。この際のデータセットの分割方法は付録 A の A.2 節に示す。

5.5 実験結果

本節では、実験結果及びその考察をまとめた。実験結果に記載する画像については三つの波形を上から並べており、上から順に元のギターの波形、変換後の波形、ハープの波形となっている。

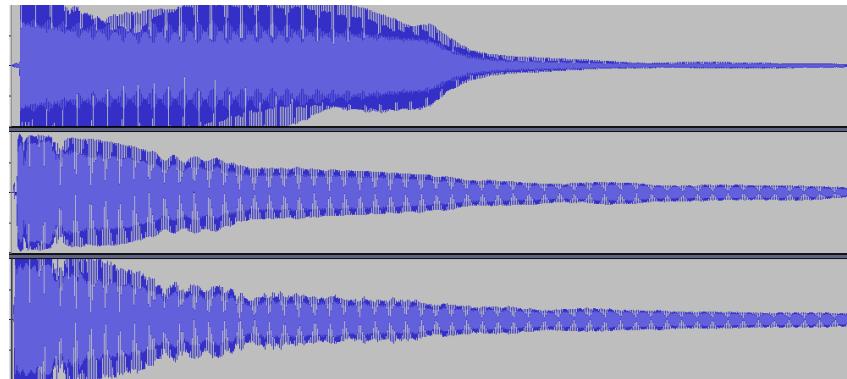


図 5.4 C5 の 0.000 秒から 1.000 秒までの音波

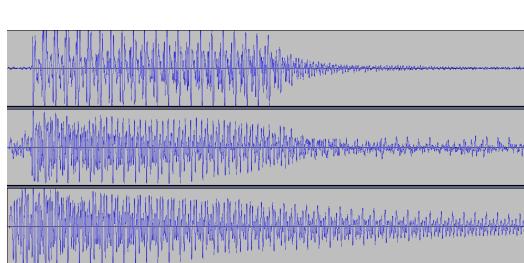


図 5.5 F1♯ の 0.000 秒から 1.000 秒までの音波

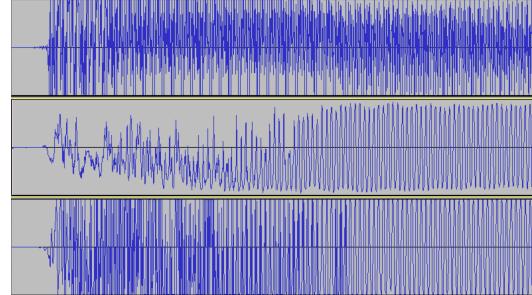


図 5.6 A7 の 0.000 秒から 0.030 秒までの音波

5.5.1 生成モデルの表現力の評価

まず、F2 から G6♯ の音は耳で聴いても波形を見ても図 5.2 のように正確にハープの音を表現することができていた。特に C4 から D5♯ の音は図 5.3 のように上音の音が少ないために綺麗なハープの音を生成することができていた。他の音についても下記に挙げた改善点はあるものの基音の部分は表現することができていた。以下では、ハープの音を生成において困難であった点を列挙する。

5.5.1.1 音波の振幅

図 5.4 の波形の前半のように、生成された波形の振幅が小さいという問題が発生した。これにより、減衰していく部分の表現が難しい場合があった。原因是、振幅をランダムに小さくしたためであると考えられ、学習の初段階では振幅を固定するなどの工夫する必要があると思われる。

5.5.1.2 音の鳴り出しの遅延

ギターとハープは弦楽器なので鳴り出すまでに遅延がある。特に、今回用いたデータセットでは図 5.5 のようにギターの音に遅延がある場合や図 5.6 のように周期的な音になるまでに遅延がある場合、その部分を学習することが難しかった。

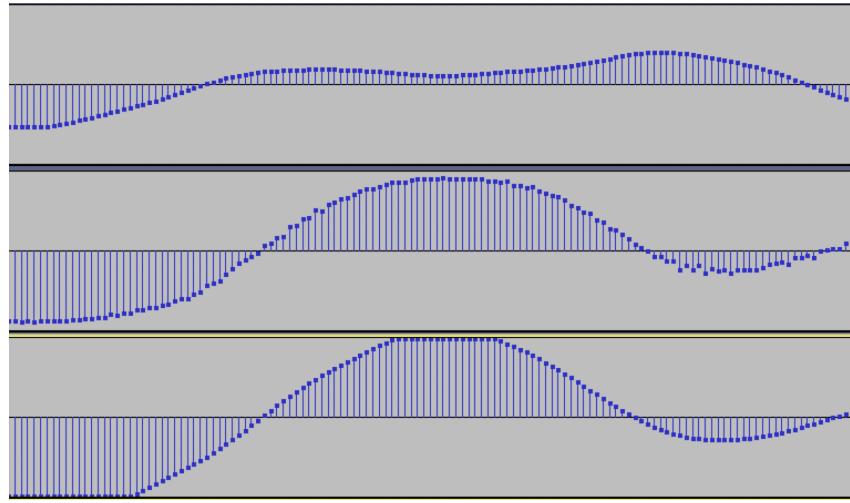


図 5.7 D2♯ の 0.100 秒から 0.103 秒までの音波

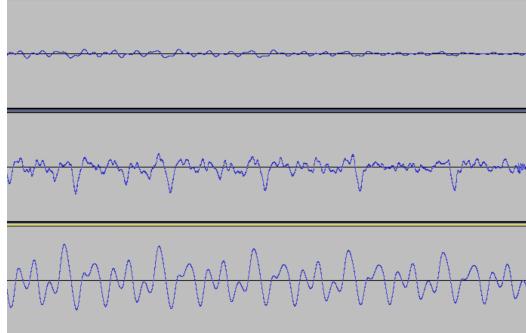


図 5.8 A0 の 0.800 秒から 1.000 秒までの音波

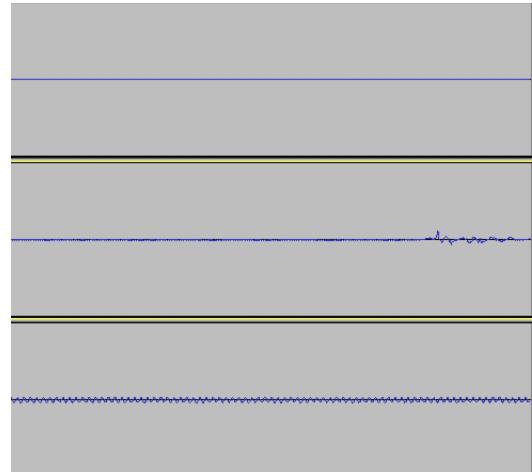


図 5.9 B7 の 0.980 秒から 1.000 秒までの音波

5.5.1.3 音波の滑らかさ

図 5.7 の上から 2 番目の波形と 3 番目の波形を比較すると、生成された波には滑らかさがないことがわかる。人間の耳にはこの滑らかでない音波の部分はノイズまじりの音として聞こえるので、この波を滑らかにするための工夫が必要であると考えられる。

5.5.1.4 音波の振動の減衰

音波の振動の減衰が全く表現できていない音波があった。具体的には、A0,A0♯,B0,C1,C1♯,D1,D1♯,E1,F1,F1♯,G1,G1♯,D2,D2♯,E2,A6,A6♯,B6,C7,C7♯,D7,D7♯,F7,F7♯,G7,G7♯,A7,A7♯,B7,C8 の音である。また、これ以外の音についても減衰が一部表現できていないものもあった。

これらの音のうち、E2 以下の低音域の場合は図 5.8 のようにハープとは全く異なる波形で減衰し、A6 以上の高音域の場合は図 5.9 のようにほとんど振動が見られなかった。原因としては、微小な振動をニューラル

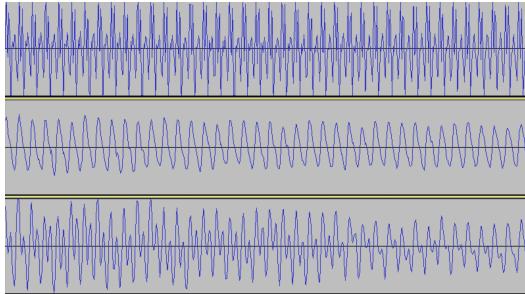


図 5.10 E7 の 0.055 秒から 0.070 秒までの音波

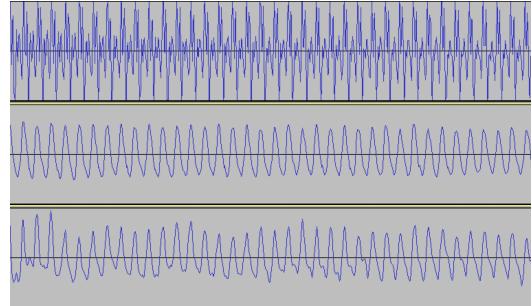


図 5.11 D7# の 0.055 秒から 0.070 秒までの音波

ネットワークが学習するのが難しいことと今回のデータセットではギターの方がハープよりも音の鳴る時間が短かったことが挙げられる。

5.5.1.5 安定したデータセットの作成

E7 の音については、図 5.10 のように上記に挙げた以外の部分で波が表現できていなかったが、ハープのデータセットの音波の振動の振幅が安定していないために、ニューラルネットワークによる学習が難しかったと考えられる。

また、E7 の半音下の音である D7# でも図 5.11 のようにハープのデータセットの音波の振動の振幅は安定しておらず、特に高音は安定した振幅のデータセットを生成することが難しいと考えられる。

5.5.2 生成モデルの汎化能力

まず、音の高さは変換前後で変わってないものが多く期待通りの結果となった。しかし、音の音色は元のギターの音色とは異なるもののハープとも異なる音色となり、5.5.1 節における課題も解決することはできなかった。以下では、生成されたハープの音について波形を元に考察を行う。

5.5.2.1 音波の単純さ

5.5.1 節と同程度にハープに近い音が生成される場合もあった。具体的には、D4,D4#,G4,F5,F5# の音である。これらの音は、音波が図 5.12 のように単純な波形でニューラルネットワークによる表現が容易であるため、生成が他の音波に比べて容易であったと考えられる。

5.5.2.2 音波の複雑さ

上記の単純な音波以外の場合はギターの音色とは異なるもののハープの音色とも異なる音が生成された。いくつか種類があったが、図 5.13 や図 5.14 のように、音波が安定した振動をせず上音の成分がハープよりも多い音波が多く観測された。

また、ほとんどの音波において基音は保たれていたが、図 5.15 のように基音が同じで位相が異なる音波や図 5.16 のように基音が異なる音波が生成されることもあった。

これらの原因は、振幅をランダムにすることにより学習が安定しないことと微細な振動を表現可能なニューラルネットワークを構築できていないためと考えられる。後者の場合は画像で用いられる GAN での画像の高

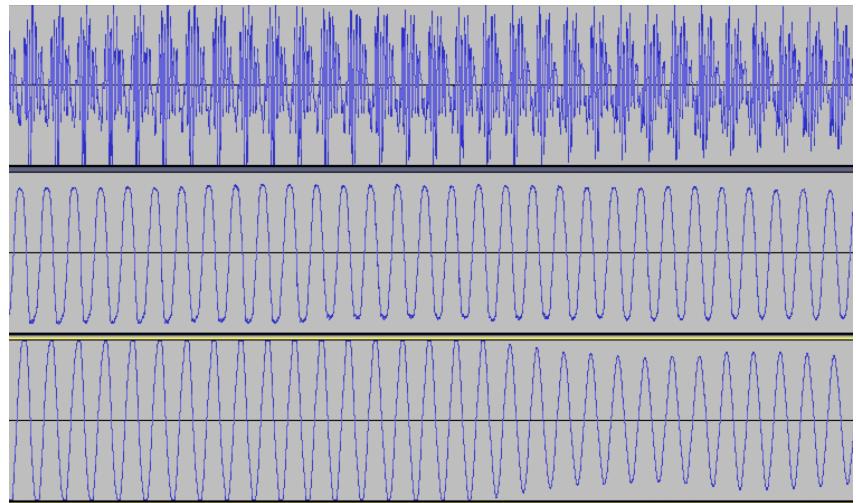


図 5.12 D4♯ の 0.100 秒から 0.200 秒までの音波

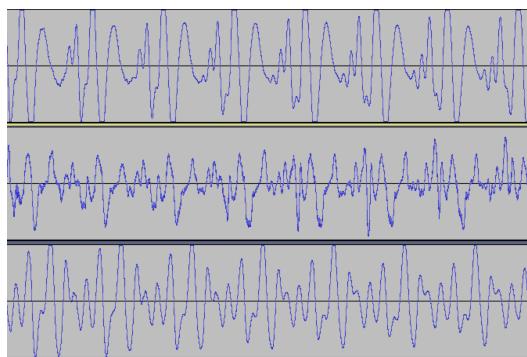


図 5.13 D1 の 0.300 秒から 0.500 秒までの音波

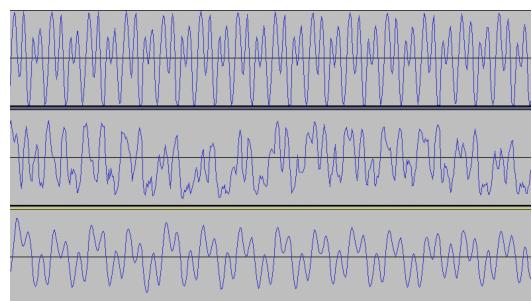


図 5.14 F6 の 0.070 秒から 0.080 秒までの音波

解像度化の工夫が適用できると期待される。

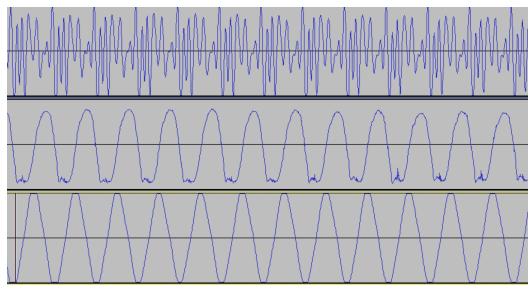


図 5.15 G4♯ の 0.150 秒から 0.180 秒までの音波

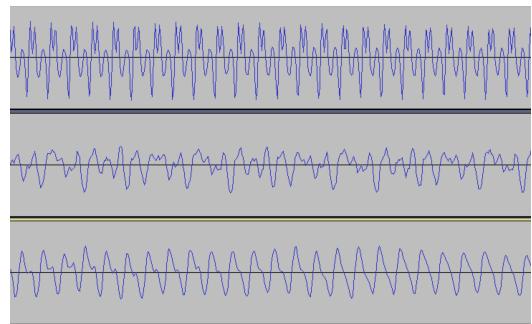


図 5.16 D7♯ の 0.1700 秒から 0.110 秒までの音波

第6章

まとめと展望

本研究での実験の結果、提案手法における生成モデルはギターからハープへと音色の変換を行うには十分な表現力を持つことを確認することができた。ただし、ニューラルネットワークのモデルによる細かい音波の表現、学習時のデータの振幅の乱雑さの加え方、安定したデータセットの作成、の三つの点においては課題が残った。そのため、4分割交差検証を行った際にはその影響が増幅し、音程はほとんどの変換で維持できているものの、ハープの音へと変換することができたのは一部のみであった。

また、本研究では波形の観察による考察を行ったが、より定量的な判定を行う必要がある。具体的には、音程が維持されているかと音色が正しく変換されているかの判定を考慮できると良い。前者についてはフーリエ変換を用いて実装することができるが、後者については考察の余地がある。

6.1 まとめ

6.2 展望

一般に、音楽で特定の音色への変換を行うことは難しいが、次の三つの要素に分解することで単音での音色の変換を音楽へと適用することが可能であると考えられる。また、三つの要素とは、楽器の重ね合わせ、音の重ね合わせ、時間方向の音の繋ぎ方であり、それについて具体的に以下で説明をする。

6.2.1 楽器の重ね合わせ

音楽はそれぞれの楽器により奏でられた音の重ね合わせになっている。楽器ごとに音色は異なるので楽器ごとの音波に分解して音色変換を行う（音源分離）が必要であると考えられる。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、音源分離を行わずに直接楽器ごとの音波を利用できる。

6.2.2 音の重ね合わせ

ある単位時間の音に注目した時、楽器ごとの音波に分解してもその単位時間で異なった高さや大きさの音の重ね合わせになっていることがある。この場合は、今回の提案手法で用意したデータセット以外に和音のデータセットも加えて学習させることで表現可能であると考えられる。

6.2.3 音の繋ぎ方

楽器ごとの音波に分解し単位時間の音が表現できた時、時間方向に音を繋ぐ必要がある。時間方向については先程定めた単位時間で区切って順に変換していくことで可能であると考へている。また、区切るのみでの変換が難しい場合は自己回帰モデルを取り入れるなどの工夫が必要であると考えている。

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] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018.
- [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] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, Vol. abs/1609.04747, , 2016.
- [4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
- [5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, Vol. 25, pp. 1097–1105. Curran Associates, Inc., 2012.
- [6] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, Vol. 36, pp. 193–202, 1980.
- [7] Cnn_survey.pdf. http://www.nlab.ci.i.u-tokyo.ac.jp/pdf/CNN_survey.pdf. (Accessed on 02/07/2021).
- [8] アイドル自動生成 ai を開発 — 株式会社データグリッド — datagrid inc. <https://datagrid.co.jp/all/release/33/>. (Accessed on 02/03/2021).
- [9] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014.
- [10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.
- [11] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, Vol. 15, No. 56, pp. 1929–1958, 2014.

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 データセットの分割

生成モデルの汎化能力の評価において 4 分割交差検証を行ったため、図 A.1 のようにデータセットを 4 つのサブセットに分割した。データセットの分割方法としては 88 音をシャッフルして配列に格納した後に 22 音ずつ順に選んでいる。

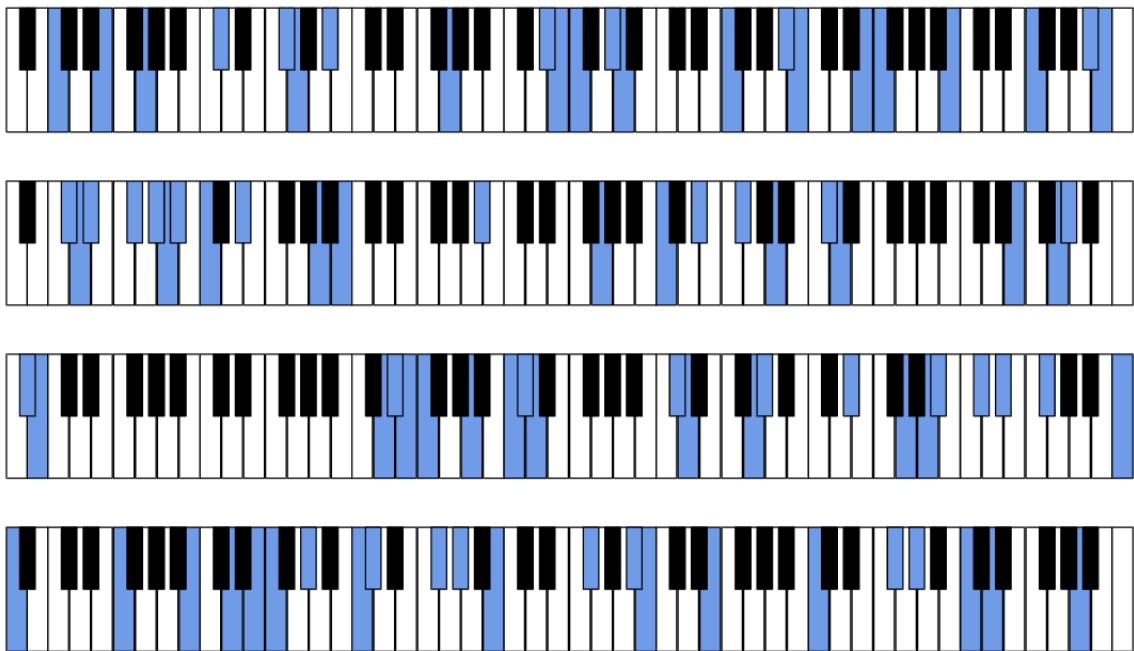


図 A.1 データセットの分割方法