

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	2
第 2 章	音楽	3
2.1	音	3
2.2	楽音の要素	3
第 3 章	音声処理	6
第 4 章	ニューラルネットワーク	7
4.1	Multilayer perceptron	7
4.2	Generative Adversarial Networks	8
4.3	Pix2pix	9
第 5 章	提案手法	12
5.1	提案モデル	12
5.2	実験方法	12
5.3	実験結果	14
第 6 章	まとめ	20
6.1	課題	20
6.2	展望	20
参考文献		23
付録 A		24
A.1	実験時のパラメータ	24
A.2	データセットの分割方法	24

第1章

はじめに

音楽は世界中で楽しめている。音楽の作成方法には様々なものがあり、既存の曲をアレンジして新しい曲を作成する Remix と呼ばれる方法がある。しかし、Remix は画面上で音を操作するソフトウェアアプリケーション (DAW) を用いるのが一般的である。初心者が DAW を使用するのは難しいため、コンピュータプログラムによる補助が役に立つと考えられる。本研究では、Remix の代表的な方法の一つである音色の変換に着目し、プログラムによる変換手法を提案する。

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。そこで、本研究では 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 Generative Adversarial Networks (GAN) [2] を用いて画像のスタイル変換を行う手法である。

本研究では、ギターの単音をハープの単音に提案手法を用いて変換する実験を行った。その結果、音の大きさが変わることなどの問題はあったものの、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、音は楽譜作成ソフトの MuseScore^{*1}により作成し、音波の画像は音声編集ソフトの Audacity^{*2}により作成した。

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

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

第2章

音楽

本章では音楽用語の定義を行う。

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 楽音の要素

楽音は長さ、高さ、大きさ、音色の四つの要素から成り立ち [3]、人間はこれらを知覚することができる。本論文では楽音のことを音と呼ぶ。

2.2.1 音の長さ

音の長さは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 音の高さ

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れる。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。国際の音名表記では、C,D,E,F,G,A,B を西洋音楽の七音音階におけるオクターブとして定め、それぞれのオクターブに番号を振り、440Hz 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。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。

ここで、図 2.1 は図 2.2 で示されるある音波にフーリエ変換を行うことで得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 とわかる。

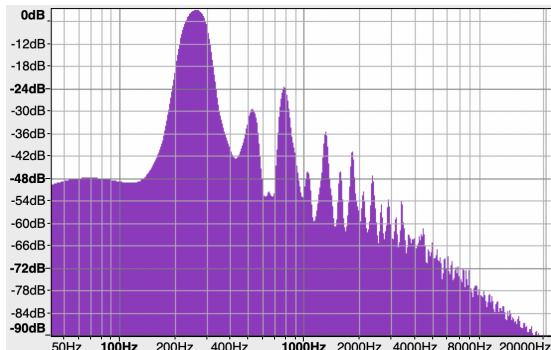


図 2.1 音波の周波数スペクトル

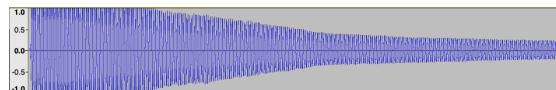


図 2.2 C4 の音波の波形

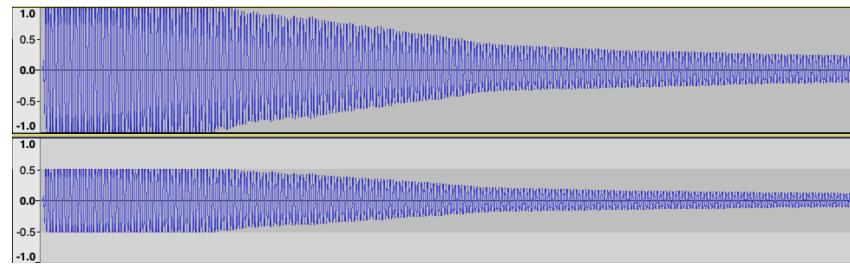


図 2.3 音の大きさの異なる音波

2.2.3 音の大きさ

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れる。

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。

2.2.4 音の音色

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とがある。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が波形は異なる。音色の異なる音どうしは基音より高い音（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。この波形の違いが音色の違いを作り出す。

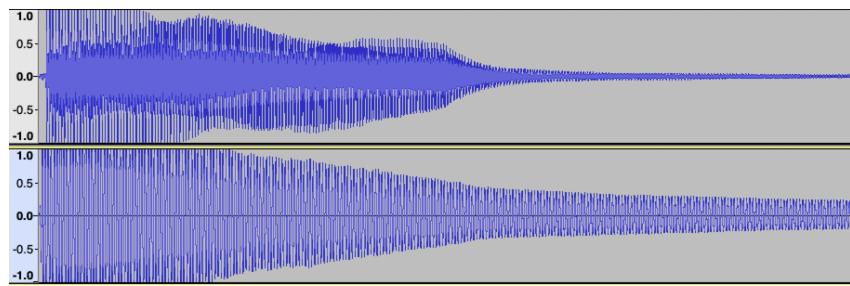


図 2.4 ギターとハープの音色

第3章

音声処理

本章では、主要な音声処理の方法を紹介する。

以下の単語は説明

サンプリング周波数] デジタル信号の1秒あたりの標本化の回数

サンプリング数] 音波のデジタル信号の標本化の合計の回数のこと

量子化ビット数] デジタル信号の細かさを表現するビット数のこと

チャンネル数] モノラルな音声出力の総数のこと

第 4 章

ニューラルネットワーク

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。

4.1 Multilayer perceptron

Multilayer perceptron (MLP) は複数のアフィン変換と非線形関数を合成した関数により定義され、式 4.1 で定式化される。

$$F_{MLP}(\mathbf{x}) = f_n(W_n(f_{n-1}(W_{n-1} \cdots (f_1(W_1\mathbf{x} + \mathbf{b}_1)) \cdots + \mathbf{b}_{n-1})) + \mathbf{b}_n) \quad (4.1)$$

ここで、 \mathbf{x} は実ベクトル, n は 1 以上の整数, f_i は i 番目の非線形関数, W_i は i 番目の行列, b_i は i 番目の実ベクトルである。また、MLPにおいては、 n を層数、 W_i, b_i を i 番目の層の重み、と呼ぶ。

MLP (F_{MLP}) を用いると、あるデータ集合 $D = \{(\mathbf{x}_j, \mathbf{t}_j); 1 \leq j \leq N\}$ について、任意の j で \mathbf{t}_j の予測値として $\mathbf{y}_j = F_{MLP}(\mathbf{x}_j)$ を出力することができる。 \mathbf{t}_j に近い \mathbf{y}_j を出力するためには、任意の i で W_i, b_i を適切に定める必要がある。

y_j と t_j の誤差を表す関数として損失関数 $L(\mathbf{y}_j, \mathbf{t}_j)$ を定義する。損失関数 L を式 4.2 に従って更新することで、 \mathbf{y}_j を \mathbf{t}_j に近づける方向に層の重み W_i, b_i を更新することができる(勾配降下法)。

$$W_i \leftarrow W_i - \eta \frac{dL}{dW_i} \quad (4.2)$$

$$b_i \leftarrow b_i - \eta \frac{dL}{dB_i} \quad (4.3)$$

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、この学習率を適切に設定するために Stochastic Gradient Descent [4] や Adam [5] などのアルゴリズムが用いられる。

さらに、式 4.4 で表される平均二乗誤差や式 4.5 で表される平均絶対誤差などが損失関数として用いられる。

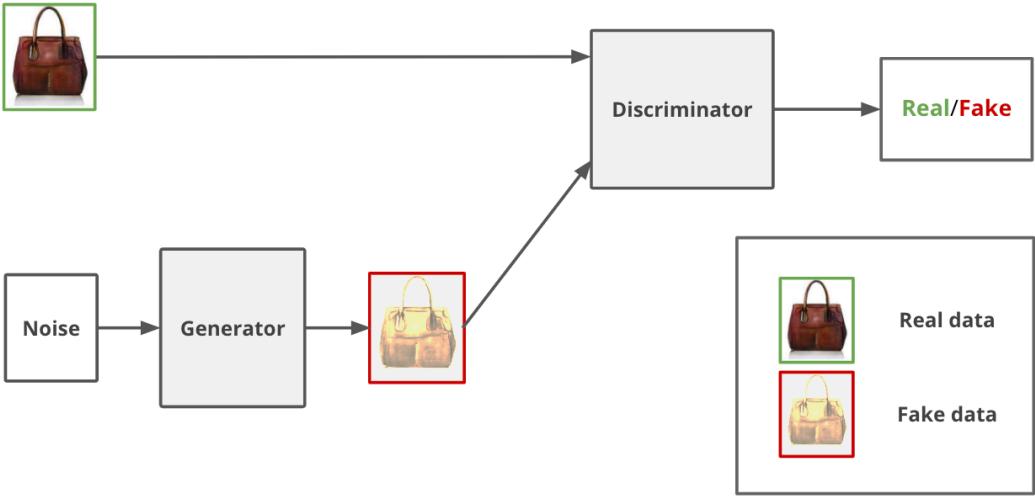


図 4.1 GAN のネットワーク、図は文献 [1] の Figure 1 を用いて作成。

$$L_{MSE} = \frac{1}{n} \sum_{j=1}^n (y_j - t_j)^2 \quad (4.4)$$

$$L_{MAE} = \frac{1}{n} \sum_{j=1}^n |y_j - t_j| \quad (4.5)$$

一般には、ニューラルネットワークは MLP より広い概念であるが、本論文ではこれ以降 MLP のことをニューラルネットワークと呼ぶ。

4.2 Generative Adversarial Networks

Generative Adversarial Networks (GAN) [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学習して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 [6]。

GAN は図 4.1 のように二つのニューラルネットワークで構成される。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。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に、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。

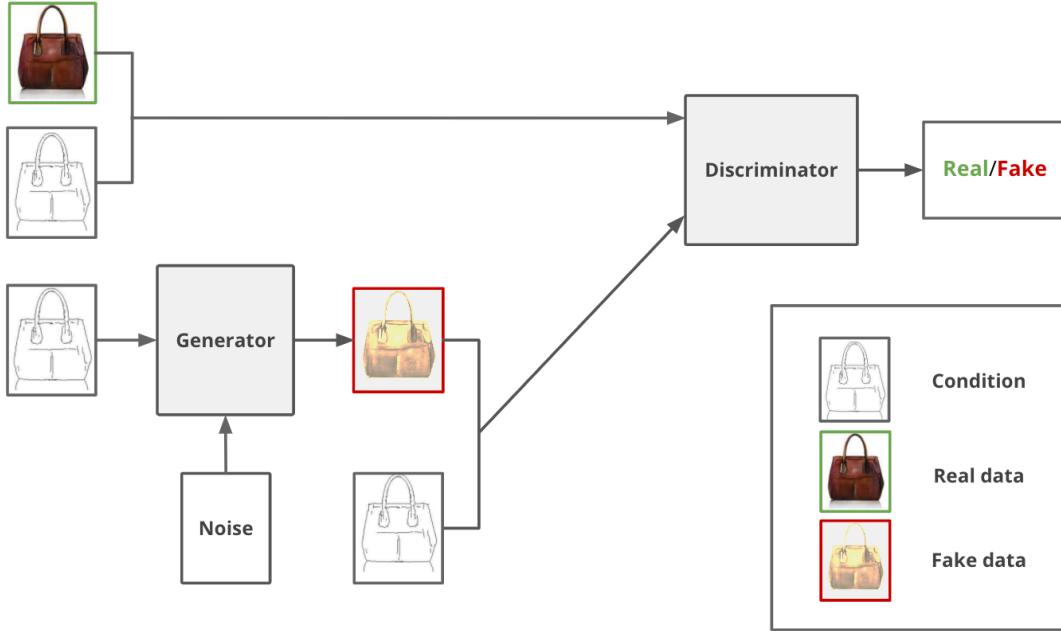


図 4.2 pix2pix のネットワーク、図は文献 [1] の Figure 1 を用いて作成。

そして、GANにおいては、生成モデルの目的関数は式 4.6、識別モデルの目的関数は式 4.7として定式化される。

$$\arg \min_{\theta_G} \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}; \theta_G); \theta_D))] \quad (4.6)$$

$$\arg \max_{\theta_D} \mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x}; \theta_D)] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}; \theta_G); \theta_D))] \quad (4.7)$$

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。

4.3 Pix2pix

Pix2pix [1] は図 4.2 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) [7] 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 4.3 のように線画の写真への変換や白黒写真のカラー画像への変換を行うことができる。

そして、Pix2pixにおいては、生成モデルの目的関数は式 4.8、識別モデルの目的関数は式 4.9として定式化される。



図 4.3 pix2pix のスタイル変換の例、図は文献 [1] の Figure 1 を用いて作成。

$$\arg \min_{\theta_G} \mathbb{E}_{y,z} [\log(1 - D(y, G(y, z; \theta_G); \theta_D))] + \mathbb{E}_{x,y,z} [\|x - G(y, z; \theta_G)\|_1] \quad (4.8)$$

$$\arg \max_{\theta_D} \mathbb{E}_{x,y} [\log D(x, y; \theta_D)] + \mathbb{E}_{y,z} [\log(1 - D(y, G(y, z; \theta_G); \theta_D))] \quad (4.9)$$

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。

4.3.1 生成モデルの構造

Pix2pix の生成モデルには、図 4.4 のように Encoder-Decoder 型のネットワークが用いられる。ただし、変換元の画像とのピクセルの対応関係を維持するために、Skip Connection (スキップコネクション) を持つ [8]。

また、ノイズとしては実ベクトルではなく Dropout が用いられる [9]。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。

4.3.2 識別モデルの構造

Pix2pix の識別モデルには、PatchGAN という手法が用いられる。PatchGAN は図 4.5 のように画像全体ではなくパッチと呼ばれる小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。

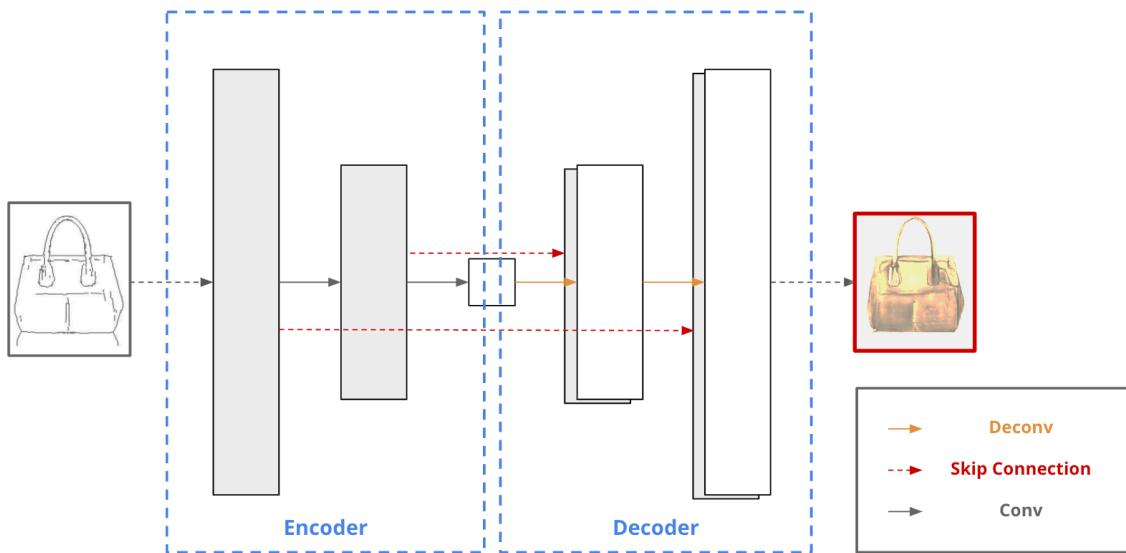


図 4.4 生成モデルのネットワーク、図は文献 [1] の Figure 1 と Figure 3 を用いて作成。



図 4.5 通常の GAN と PatchGAN の比較、図は文献 [1] の Figure 1 を用いて作成。

第 5 章

提案手法

本章では、提案モデルの説明を行った後、実験結果の提示及び考察を行う。

5.1 提案モデル

本研究では、Pix2pix を元に作成した図 5.3 のモデルを提案する。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。

5.1.1 データの前処理

本研究では、第 3 章にて紹介した音声処理を行わず、44100 Hz のサンプリング周波数でサンプリングした 1 秒の Raw Data を使用した。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、本研究で使用した音は 44100 の長さを持つ 16 ビット整数の一次元配列として扱われる。

5.1.2 生成モデル

本研究で使用した生成モデル

5.1.3 識別モデル

本研究で使用した識別モデル

5.2 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。

5.2.1 データセット

データセットとして 1 秒のギターとハープの音を 88 音用意した。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。そして、88 音としては国際の音階表記で A0 から C8 の半音を選んだ。これらは図 5.4 のような一般的な 88 鍵のピアノで用いられる音である。

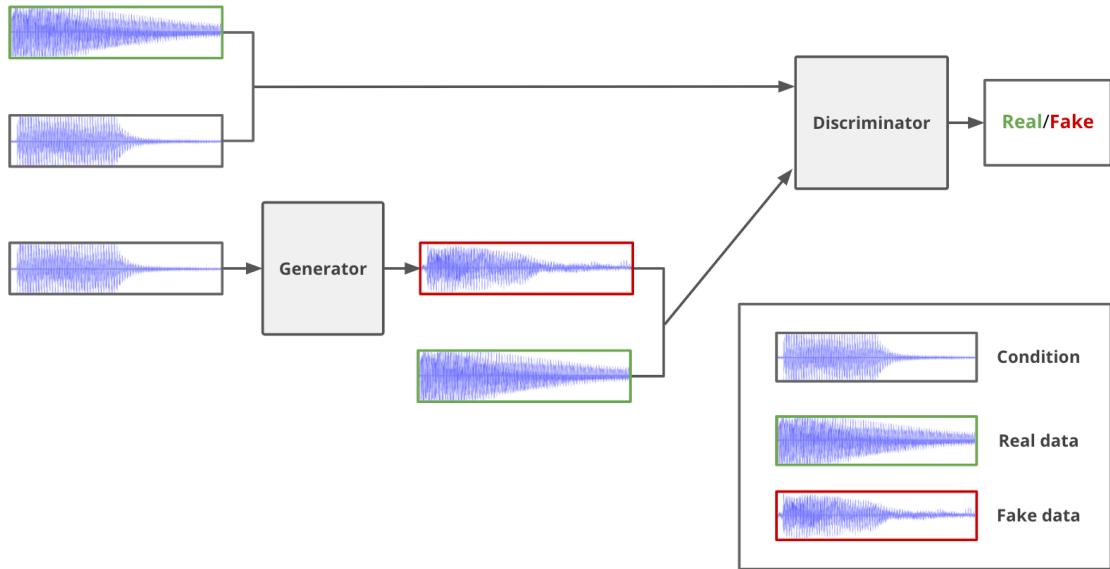


図 5.1 提案モデル

5.2.2 評価方法

5.2.3 実験時の工夫

実験時には学習データの振幅に以下の二つの工夫を行った。

5.2.3.1 学習データの振幅の正規化

各エポックで学習データとして与えるデータに正規化を行った。これにより、16bit 整数であった量子化ビット数は [0,1] で表現される。また、評価時に生成する音は 16bit 整数へと再変換している。

5.2.3.2 学習データの振幅の無作為化

学習データの振幅の正規化の後に振幅を c 倍した。 $c \in [0.3, 1]$ である。また、乱数は、学習の前にシードを固定した後に一様乱数で生成している。これにより、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する。

5.2.4 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同じ 88 音のデータセットを用いて実験を行う。

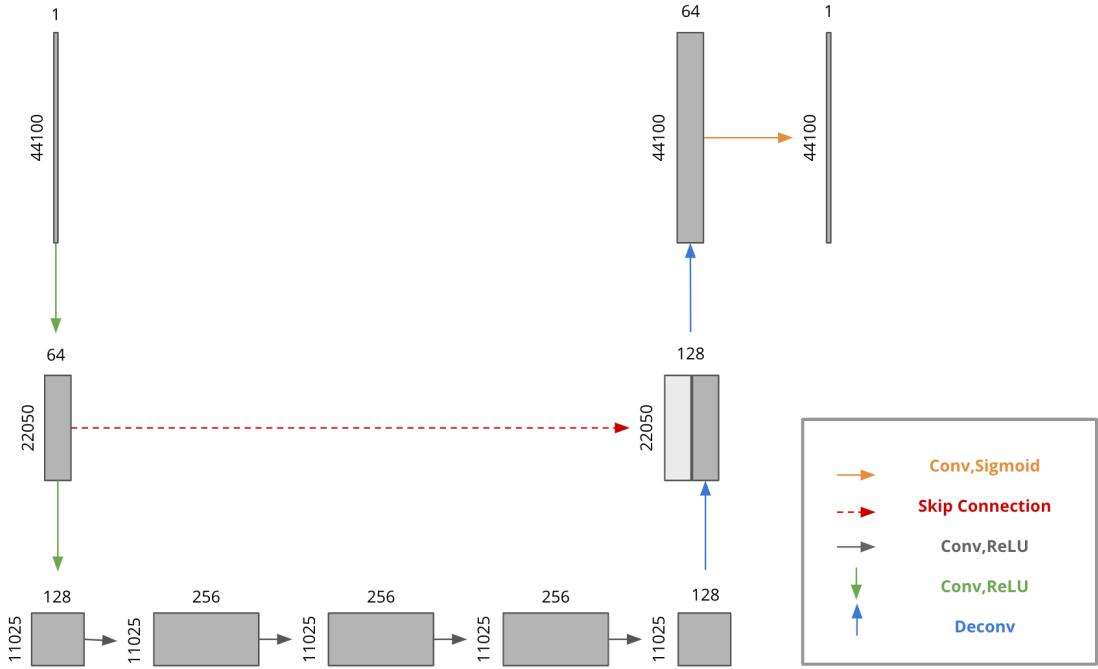


図 5.2 生成モデル

5.2.5 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。

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。

5.3 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。

5.3.1 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 5.5 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 5.6 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。

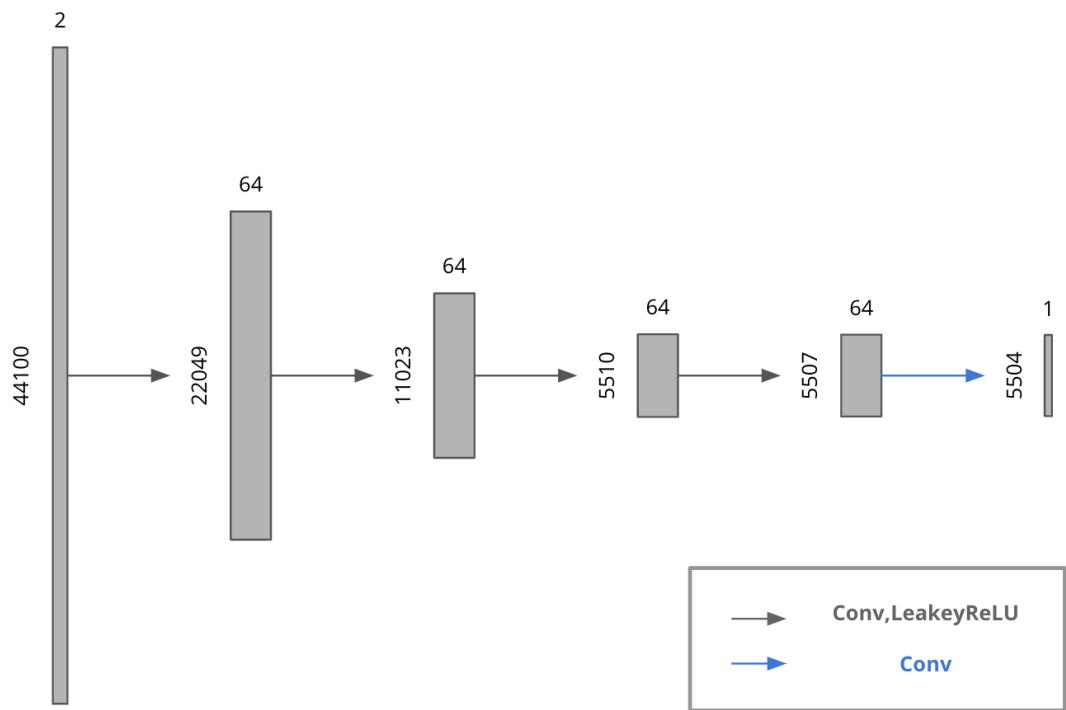


図 5.3 識別モデル

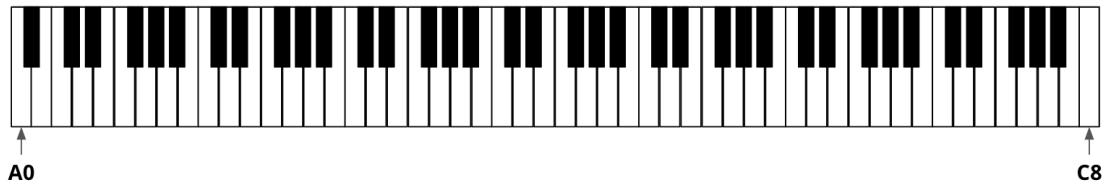


図 5.4 88 鍵のピアノの鍵盤

5.3.1.1 音波の振幅

図 5.7 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。原因は、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。

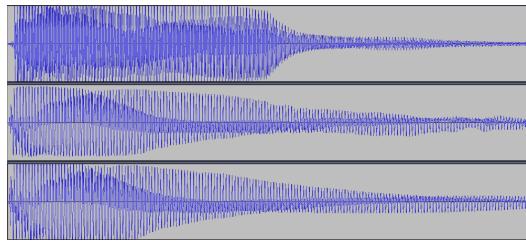


図 5.5 F3 の 0.800 秒から 1.000 秒までの音波

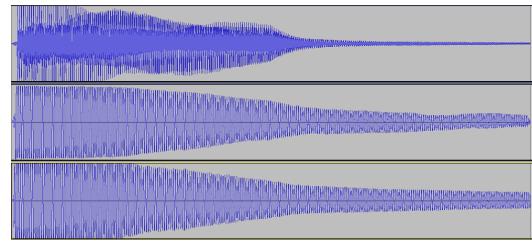


図 5.6 C4 の 0.200 秒から 0.300 秒までの音波

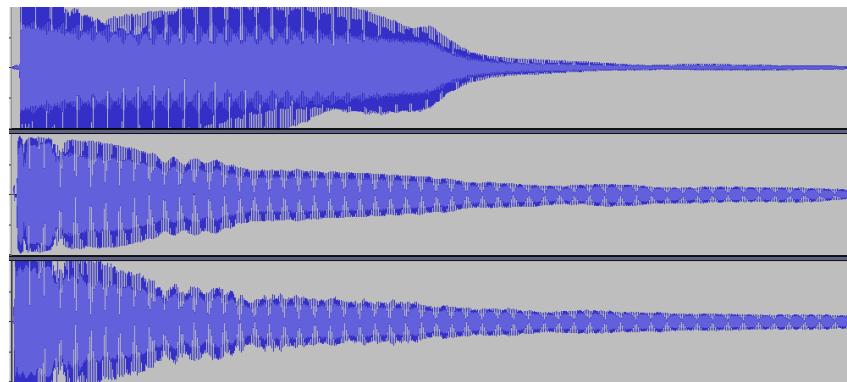


図 5.7 C5 の 0.000 秒から 1.000 秒までの音波

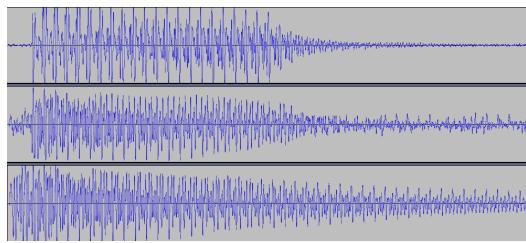


図 5.8 F1♯ の 0.000 秒から 1.000 秒までの音波

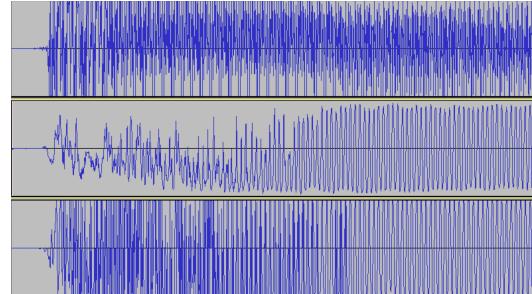


図 5.9 A7 の 0.000 秒から 0.030 秒までの音波

5.3.1.2 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 5.8 のようにギターの音に遅延がある場合や図 5.9 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。

5.3.1.3 音波の滑らかさ

図 5.10 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

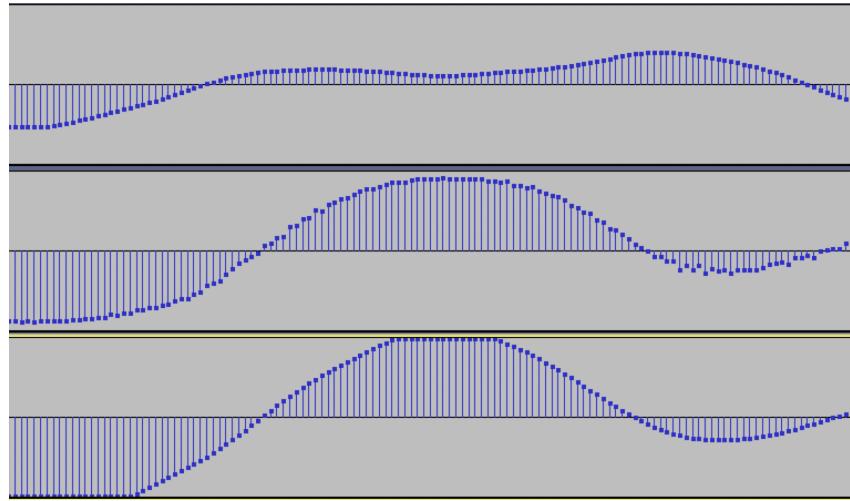


図 5.10 D2♯ の 0.100 秒から 0.103 秒までの音波

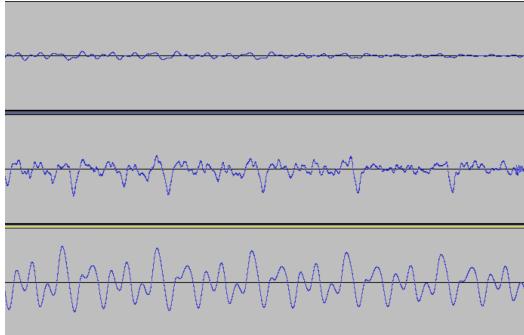


図 5.11 A0 の 0.800 秒から 1.000 秒までの音波

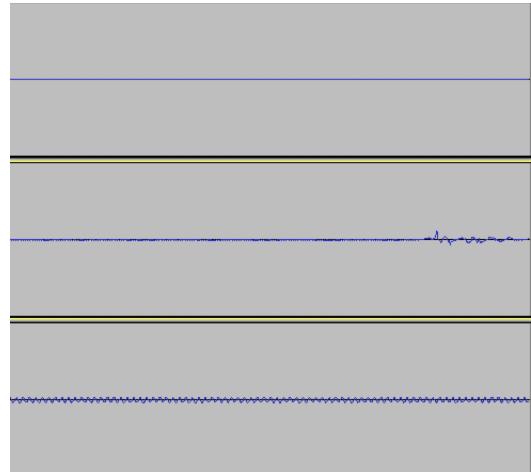


図 5.12 B7 の 0.980 秒から 1.000 秒までの音波

るための工夫が必要であると考えられる。

5.3.1.4 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はあった。

これらの音のうち、E2 以下の低音域の場合は図 5.11 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 5.12 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。

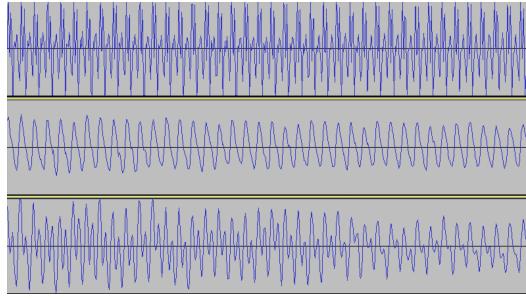


図 5.13 E7 の 0.055 秒から 0.070 秒までの音波

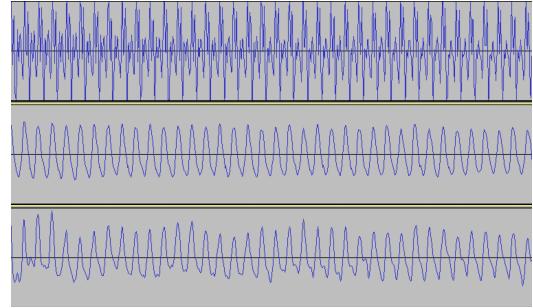


図 5.14 D7♯ の 0.055 秒から 0.070 秒までの音波

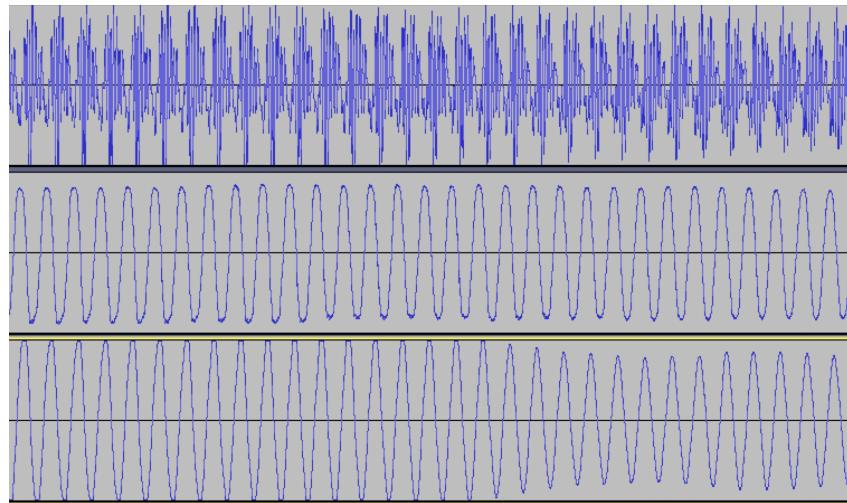


図 5.15 D4♯ の 0.100 秒から 0.200 秒までの音波

5.3.1.5 安定したデータセットの作成

E7 の音については、図 5.13 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 5.14 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。

5.3.2 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、5.3.1 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。

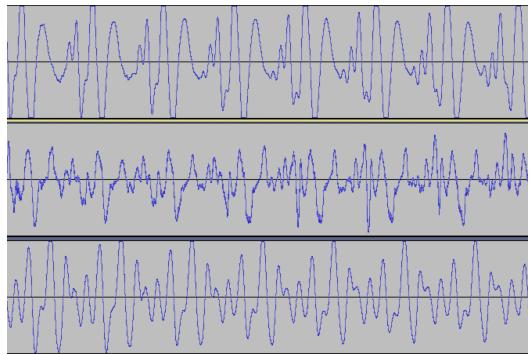


図 5.16 D1 の 0.300 秒から 0.500 秒までの音波

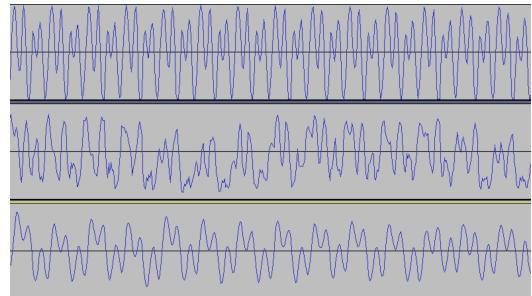


図 5.17 F6 の 0.070 秒から 0.080 秒までの音波

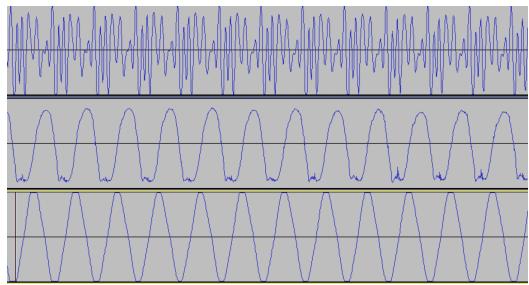


図 5.18 G4♯ の 0.150 秒から 0.180 秒までの音波

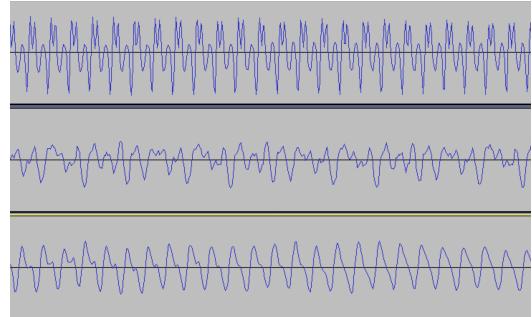


図 5.19 D7♯ の 0.1700 秒から 0.110 秒までの音波

5.3.2.1 音波の単純さ

5.3.1 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 5.15 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。

5.3.2.2 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 5.16 や図 5.17 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 5.18 のように基音が同じで位相が異なる音波や図 5.19 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 での画像の高解像度化の工夫が適用できると期待される。

第6章

まとめ

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 課題

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、それぞ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 考えている。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] 芥川也寸志. 音楽の基礎. 岩波新書. 岩波書店, 1971.
- [4] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, Vol. abs/1609.04747, , 2016.
- [5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
- [6] 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).
- [7] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014.
- [8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.
- [9] 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方法

生成モデルの汎化能力を調べた際のデータセットのサブセットへの分割を表 A.2 に示す。表記は国際の階名表記に従い、音の高さの昇順に並んでいる。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。また、4 分割検証を行っているため、番号 i ($0 \leq i \leq 4$) の 22 音のサブセットを評価に用いた際は番号 j ($0 \leq j < 4$ かつ $i \neq j$) のサブセットの 66 音を全て学習に用いている。

表 A.2

番号	
0	C1 E1 G1 C2♯ F2♯ G2 A2♯ G3 D4♯ E4 F4 G4♯ A4 F5 A5♯ B5 E6 F6 B6 F7 A7♯ B7
1	C1♯ D1 D1♯ F1♯ G1♯ A1 A1♯ C2 D2♯ A2 B2 A3♯ G4 C5 D5♯ F5♯ A5 C6♯ D6 E7 G7 G7♯
2	A0♯ B0 D3 D3♯ E3 F3 A3 C4 C4♯ D4 C5♯ D5 G5 G5♯ D6♯ G6 A6 A6♯ C7♯ D7♯ F7♯ C8
3	A0 F1 B1 D2 E2 F2 G2♯ C3 C3♯ F3♯ G3♯ B3 F4♯ A4♯ B4 E5 C6 F6♯ G6♯ C7 D7 A7