

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

	Page
第 1 章 はじめに	5
第 2 章 背景：音楽	6
2.1 音の定義	6
2.1.1 音の長さ	6
2.1.2 音の大きさ	6
2.1.3 音の高さ	7
2.1.4 音の音色	7
2.2 音楽の特徴	8
2.2.1 楽器の重ね合わせ	8
2.2.2 音の重ね合わせ	8
2.2.3 音の繋ぎ方	8
2.3 音楽の表現	8
2.3.1 一次元データと二次元データ	8
2.3.2 音響信号	9
2.3.3 STFT	9
2.3.4 メルスペクトログラム	9
2.3.5 CQT	10
2.3.6 クロマグラム	10
第 3 章 背景：ニューラルネットワーク	11
3.1 教師あり学習	11
3.1.1 教師あり学習の目的	11
3.1.2 教師あり学習モデルの学習と汎化	11
3.2 MLP	12
3.2.1 MLP と脳機能の関係	12
3.2.2 MLP の構造と定式化	13
3.2.3 MLP の学習	14
3.3 CNN	15
3.3.1 CNN の構造	15
3.3.2 畳み込み層	16

3.3.3 プーリング層	17
3.4 GAN	18
3.4.1 GAN の学習	18
3.4.2 GAN の定式化	19
3.5 Pix2pix	19
3.5.1 Pix2pix の定式化	20
3.5.2 Pix2pix の生成モデル	20
3.5.3 Pix2pix の識別モデル	21
第 4 章 提案手法	22
4.1 データセットと音の表現	22
4.2 提案モデル	22
4.2.1 生成モデル	23
4.2.2 識別モデル	24
4.3 実験方法	24
4.3.1 提案モデルの表現力の評価実験	24
4.3.2 提案モデルの汎化能力の評価実験	24
4.3.3 評価方法	25
4.3.4 データ拡張	25
4.4 実験結果	25
4.4.1 概要：提案モデルの表現力の評価実験	25
4.4.2 概要：提案モデルの汎化能力の評価実験	26
4.4.3 課題	27
第 5 章 まとめ	29
謝辞	30
参考文献	31
付録 A Adam	32
付録 B 学習時のパラメータ	33
付録 C データセットの分割	34
付録 D DDSP	35
付録 E 実験結果	36
E.1 提案モデルの表現力の評価実験	36
E.2 提案モデルの汎化能力の評価実験	36

図目次

	Page
第 2 章	
図 2.1 音波	6
図 2.2 音波の拡大図	7
第 3 章	
図 3.1 MLP の人工ニューロン	12
図 3.2 MLP の一層	13
図 3.3 MLP のネットワーク	13
図 3.4 AlexNet のネットワーク	15
図 3.5 CNN の畠み込み層	16
図 3.6 CNN のプーリング層	17
図 3.7 GAN のネットワーク	18
図 3.8 Pix2pix のネットワーク	19
図 3.9 Pix2pix のスタイル変換の例	20
図 3.10 Pix2pix の生成モデル	21
図 3.11 通常の GAN の識別モデルと PatchGAN の比較	21
第 4 章	
図 4.1 88 鍵のピアノの鍵盤	22
図 4.2 本研究の提案モデル	23
図 4.3 本研究の生成モデル	23
図 4.4 本研究の識別モデル	24
図 4.5 F3 の音波	25
図 4.6 C4 の音波	25
図 4.7 D7♯ の音波	26
図 4.8 E7 の音波	26
図 4.9 D4♯ の音波	26
図 4.10 D7♯ の音波	26
図 4.11 D1 の音波	27

図 4.12	F6 の音波	27
図 4.13	C5 の音波	27
図 4.14	D2♯ の音波	27
図 4.15	A0 の音波	28
図 4.16	B7 の音波	28
図 4.17	F1♯ の音波	28
図 4.18	A7 の音波	28

付録 C

図 C.1	データセットのサブセット	34
-------	--------------	----

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、本論文での音の定義を行った後に、音楽の特徴量抽出の方法を紹介する。

2.1 音の定義

音とは、弾性体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。そして、本論文では楽音を音と定義するが、楽音は長さ、大きさ、高さ、音色の四要素を持つ。ここで、図 2.1 はある音波の図であり、図 2.2 はその音波の適当な部分の拡大図である。

2.1.1 音の長さ

音の長さは音波の時間長として定義する（図 2.1）。また、音の長さは楽譜上での時間の長さ（音価）として一般には定義される。

2.1.2 音の大きさ

音の大きさは音波の振幅により決まる（図 2.1）。また、人間は振幅の大きい音ほど大きく知覚する。

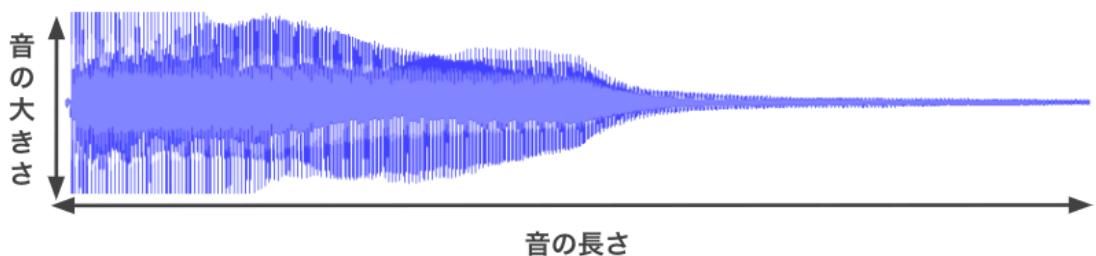


図 2.1 音波

2.1.3 音の高さ

音の高さは音波の周波数により決まる。直感的には、音波の周期構造の長さにより決まる(図2.2)。また、人間は周波数の高い音ほど高く知覚する。

2.1.3.1 単音と重音

ある楽器である高さの一音を鳴らした時に出力される音を単音と呼ぶ。また、ほとんどの単音は複数の周波数の音波の合成波であり、最も低い周波数成分の音(基音)の高さを音の高さとして人間は知覚する。そして、この単音の音を重ね合わせた音のことを重音と呼ぶ。

2.1.3.2 音階表記

本論文では、西洋音楽で用いられる12音階の表記を音の高さを表すために使用する。この表記では、 $C, C^\sharp, D, D^\sharp, E, F, F^\sharp, G, G^\sharp, A, A^\sharp, B$ の12段階の音の高さの集合をオクターブとして定める。そして、それぞれのオクターブに番号を振り、440 Hzの音をA4と定めることで音の高さの絶対的な表記を可能にしている。また、この表記で表す音を半音と呼ぶが、半音より細かい音(微分音)を表すことはできない。ただし、本論文では半音のみを音の高さとして扱う。

2.1.4 音の音色

音の長さと大きさと高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び、別の楽器は異なった音色を持つ。直感的には、音色は音波の周期構造の形により決まる(図2.2)。また、この周期構造の違いは基音より高い音(上音)の音波の組み合わせの違いに起因する。

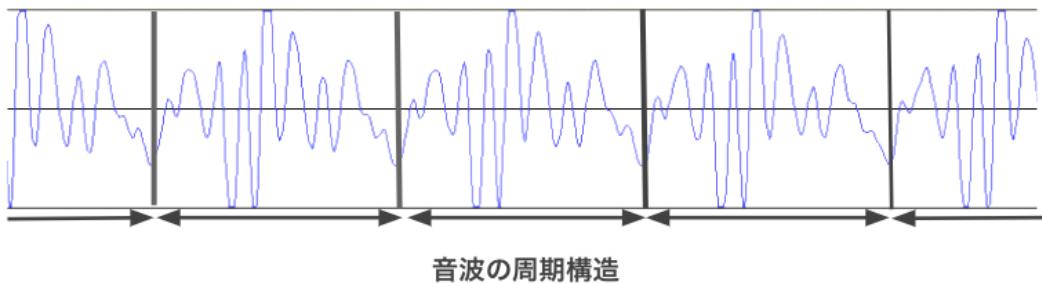


図2.2 音波の拡大図

2.2 音楽の特徴

一般には音楽で特定の音色への変換を行うことは難しいが、本研究では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た。

2.2.1 楽器の重ね合わせ

音楽はそれぞれの楽器から出力される音の重ね合わせになっている。楽器ごとに音色は異なるため、音色変換を行うには楽器ごとの音波に分解すること（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）として保存することが一般的であるため、パラデータの公開が一般的になれば音源分離の必要はなくなる。

2.2.2 音の重ね合わせ

ある音が重音である場合はそれぞれの単音について音色変換を行う必要がある。本研究では、データセットとして単音のみを使用するが、重音もデータセットに加えることで音色変換が可能であると考えられる。また、この際にデータセットが膨大な量になる可能性があり、追加するデータセットの工夫が必要である。

2.2.3 音の繋ぎ方

楽器ごとの音波に分解可能で重音も表現可能である時、時間方向の音の繋ぎ方を工夫する必要がある。また、単音の変換を一定の単位時間で行うことで、音楽もその単位時間で分割して変換することで可能であると考えられる。ただし、単位時間の定め方により性能が変わると考えられるので、単位時間は慎重に定める必要がある。

2.3 音楽の表現

ニューラルネットワークで音楽を扱うためには表現の工夫が必要である。以降の内容は [3] を参考に作成した。

2.3.1 一次元データと二次元データ

まず、基本的な表現は一次元データの音響信号であるが、ほとんどの場合は何らかの加工を施した二次元データである。また、二次元の軸としては周波数と時間を選ぶことが多い。この時、音楽のどのような特徴を把握したいか及びネットワークの時間計算量と空間計算量に合わせて適切な表現を選ぶ必要がある。

そして、多くの場合、二次元データでは中心周波数を利用したフィルタを用いて音響信号を分解し、信号を明確化する。これにより、効果的なデータ表現として扱うことができる。さらに、二次元の場合は画像と同じアルゴリズムを用いることができるが、異なる特徴が存在する。まず、画像の場合は局所的な相互関係が存在する。例えば、近くのピクセル同士は色合いや強度が似ている。しかし、スペクトログラムでは、周波数成分

の方向で調和的な関係は存在する一方で局所的な相互関係は弱い。また、調和的な関係を把握するために表現を変更するような研究もある。そして、物体認識ではスケール不变性も求められるが、音楽についてはあまり関係はないと考えられる。

2.3.2 音響信号

音響信号とは信号としての音の呼び名である。また、連続量である音響信号(アナログ信号)はコンピュータではデジタル信号に変換されて処理される。この時、標本化(サンプリング)と量子化が必要である。サンプリングとはアナログ信号を一定の間隔を空けて離散的に測定を行うことであり、量子化とは信号の大きさを離散的に近似して表現することである。そして、1秒あたりのサンプリング回数をサンプリング周波数、信号の大きさを表現するビット数を量子化ビット数、と呼ぶ。

音響信号は一次元の音の素朴な表現であるが、ニューラルネットワークの学習により音の特徴を把握するには多くのデータセットが必要になると考えられ、以降で紹介する二次元での表現へと変換されて扱われることが多い。ただ、○○などの研究では音響信号のままニューラルネットワークを用いることがあり、本研究ではこちらの音響信号の形で音を扱った。

2.3.3 STFT

STFTは線形間隔の中心周波数を用いて周波数成分を分解し、時間-周波数の二次元の表現を行う方法である。また、高速フーリエ変換を用いることにより、他の二次元の表現と比べて高速に計算を行うことができる。

しかし、線形間隔の中心周波数の利用は音楽に適しているとは言えない。なぜなら、人間の知覚系の周波数分解能は線形とかなり異なるからである。また、STFTは音楽の解析のために作られたわけでもないからである。これらの理由から、STFTはディープラーニングにおいては最も人気のある手法ではない。

ただし、STFTの音響信号との間の可逆性を利用して、音源分離などに用いられる。

2.3.4 メルスペクトログラム

メルスペクトログラムは人間の知覚系に合わせて最適化された二次元の表現であり、STFTにおいて周波数方向に関してメル尺度[4]を用いて圧縮を行ったものである。そのため、知覚として必要な情報を保持したまま圧縮を行うことができる。ただし、メルスペクトログラムは位相の情報を保持せず音の大きさのみを保持しているため、音響信号との間で非可逆的である。

また、メル尺度を表す式にはいくつかあり、式2.1はその一つである。

(2.1)

経験的かつ心理的な理由でメル尺度を用いたメルスペクトログラムは音楽のタスクに適している。(例もある)

2.3.5 CQT

CQT は対数振幅の中心周波数を使用した二次元の表現である。CQT は対数振幅を用いており、音の高さの分布に近い。したがって、基音を正確に識別する際に用いられ、和音の識別や書き換えに使うことができる。

また、計算量としては STFT やメルスペクトログラムより重い。そして、単純な対数振幅のスペクトログラムが有効な例もある。

2.3.6 クロマグラム

与えられた音の高さの集合におけるエネルギー分布のこと。多くの場合は西洋音楽の 12 音階をその集合とする。つまり、クロマグラムは周波数方向に畳み込んだ CQT と見なすことができる。また、クロマグラムは他の表現方法よりも処理が進んだものなので、それ自身を特徴量として用いることができる。

第3章 背景：ニューラルネットワーク

本章では、教師あり学習の説明を行った後、本論文で必要となるニューラルネットワークの説明を行う。

3.1 教師あり学習

教師あり学習とは、学習データとして説明変数と対応するべき目的変数のペアが与えられる機械学習の手法である。また、機械学習とは、学習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。

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集合とすると、 $f : X \rightarrow Y$ のうち任意の $\mathbf{x} \in X$ について正しい値 $\mathbf{y} \in Y$ 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、 $f(\mathbf{x})$ 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。損失関数 L は $L : Y \times Y \rightarrow \mathbb{R}^+$ として定義され、値が小さいほど近似の程度が良い関数である。 \mathbb{R}^+ は非負の実数を表す。

ここで、モデルのパラメータを θ とすると、 θ が実行可能解で損失関数の期待値を返す関数が目的関数となる最適化問題として教師あり学習を捉えることができる。また、この時の最適解 $\hat{\theta}$ により定まる関数が f' であり、目的関数は 3.1 として定式化される。

$$\mathbb{E}[L(\mathbf{y}, f(\mathbf{x}, \theta))] \quad (3.1)$$

3.1.2 教師あり学習モデルの学習と汎化

任意の \mathbf{x} と対応する \mathbf{y} の組を用意することは現実的には難しい。したがって、学習データのみでの最適化問題を解いて最適解として $\hat{\theta}$ を求めることが教師あり学習モデルの学習の目標である。

しかし、 $\hat{\theta}$ により定まる \hat{f} は f' に一致するとは限らない。従って、 \hat{f} の f' への近似の程度を評価する必要がある。そこで、学習データとは別のデータ（評価データ）を用意し、評価データについての損失関数の期待値を求めるなどにより未知のデータへのモデルの性能（汎化性能）を評価する。

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。また、ニューラルネットワークとは、神経細胞と神経細胞間のシナプス結合を通じて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の総称である。

3.2.1 MLP と脳機能の関係

MLP では、神経細胞を図 3.1 のような人工ニューロンとして表現する。この人工ニューロンでは、樹状突起からの入力を x_i として表し、神経細胞間の結合強度を重み W_i として表し、活動電位の非線形な発生を活性化関数 θ として表し、神経細胞の活動電位の発生の閾値を調整するオフセット項 b として表す。そして、 W_i を重みとした x_i の重み付き和に b を加えた値に活性化関数 θ を作用させた出力 y を返す。つまり、式 3.2 として定式化される。

$$y = \theta\left(\sum_{i=1}^m W_i \times x_i + b\right) \quad (3.2)$$

活性化関数としては ReLU 関数 (式 3.3) や Sigmoid 関数 (式 3.4) などの非線形関数が選ばれる。また、誤差逆伝播法を用いるために、一般には活性化関数に微分可能な関数を選ぶ。

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

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

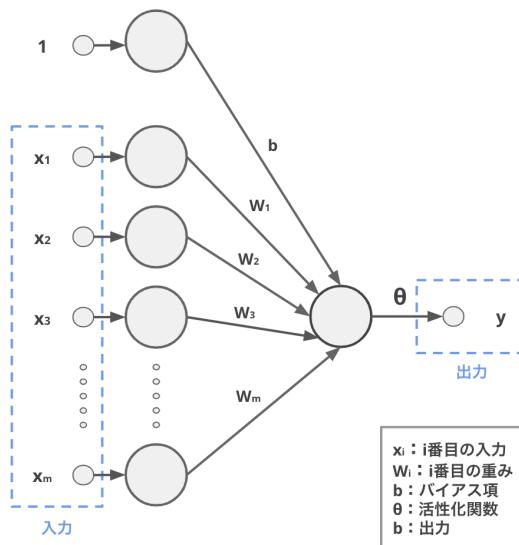


図 3.1 人工ニューロン

3.2.2 MLP の構造と定式化

MLP は、3.2.1 節に示す人工ニューロンが複数並んだ層（図 3.2）により構成され、順伝播型の層構造のニューラルネットワークを形成する（図 3.3）。また、層構造は入力層と中間層と出力層を持ち、入力層と出力層はそれぞれ一層のみである。そして、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呼ぶ。

この時、式 3.2 を元にして MLP の一層の出力 \mathbf{y} は式 3.5 として導出される。 $\mathbf{x}, \mathbf{y}, \mathbf{b}$ は第 i 成分が層の i 番目の人工ニューロンに対応するベクトルであり、それぞれ入力、出力、バイアス、である。そして、 W は (i, j) 成分が層の j 番目の人工ニューロンの i 番目の出力の重みに対応する重み行列であり、活性化関数 θ はそれぞれの出力に対して同様に作用する。

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

式 3.5 より、MLP は式 3.6 として定式化される。また、 n は層数であり、下付きの数字は何番目の層であるかを表す。

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

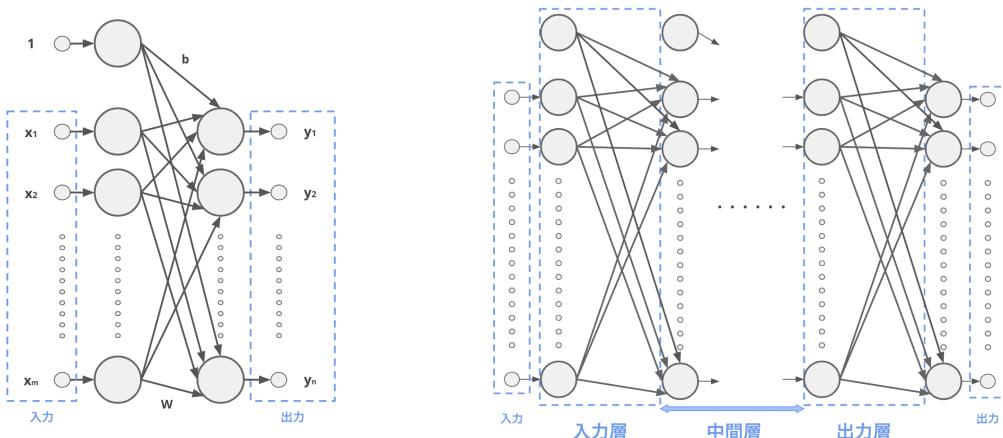


図 3.2 MLP の一層

図 3.3 MLP のネットワーク

3.2.3 MLP の学習

MLP では、説明変数と目的変数をベクトルに変換した後に学習を行う。また、実行可能解はパラメータとなる W_i, b_i であり、最適解を求めるために順次更新を行う。この時、損失関数 L の勾配を用いた最適化アルゴリズム（勾配降下法）を用いて更新するのが一般的である。

3.2.3.1 勾配降下法

勾配降下法の中で最も単純なアルゴリズムである最急降下法を紹介する。最急降下法では、最も急な降下方向である勾配の反対方向へパラメータの更新を行う。具体的には、パラメータ W_i, b_i はそれぞれ式 3.7 と式 3.8 にしたがって更新される。ここで、 η は学習率と呼ばれる正の実数であり、更新の程度を指定する。

$$W_i \leftarrow W_i - \eta \frac{\partial L}{\partial W_i} \quad (3.7)$$

$$b_i \leftarrow b_i - \eta \frac{\partial L}{\partial b_i} \quad (3.8)$$

また、勾配降下法としては SGD (Stochastic Gradient Descent) や Adam (Adaptive Moment Estimation) などが一般には用いられる [5]。本研究で用いるアルゴリズムの Adam については付録 A にその挙動を示す。

3.2.3.2 誤差逆伝播法

勾配降下法ではそれぞれのパラメータについて損失関数の勾配を求める必要があり、高速に勾配を求める方法として誤差逆伝播法が一般に用いられる。誤差逆伝播法は大きく二つの段階に分けることができる。一つ目の段階では、与えられた学習データを元に入力層から出力層の方向に（順伝播）それぞれの人工ニューロンの出力を求める。二つ目の段階では、出力層における損失関数の勾配を求めた後、勾配を誤差として出力層から入力層の方向に（逆伝播）伝えて連鎖律を使用した勾配の計算をそれぞれのパラメータについて行う。また、誤差逆伝播法と連鎖律についてはそれぞれ [6] の 5.3 節と [7] の 4.4 節に詳しい。

3.2.3.3 バッチ処理

3.2.3.1 節における最急降下法を MLP で用いる場合、一回の更新ごとに全ての学習データの期待値として誤差関数の値を求める。このように全てのデータをひとまとめに扱う方法をバッチ処理と呼び、並列計算による高速化も行うことができる。しかし、一回の更新ごとに全データの勾配計算をするために計算効率が悪いかつメモリに収まらない場合は計算を行うことができない。そこで、SGD では一回の更新ごとに一つのデータをランダムに選ぶことでバッチ処理からの改善を狙った。しかし、SGDにおいてはその更新における最適解への収束が難しいだけでなく、並列計算による高速化も行うことができない。そこで、ミニバッチ処理と呼ばれる方法を採用するのが一般的である。ミニバッチ処理では、一回の更新ごとに指定したバッチサイズのサブセットのみを用いる。これにより、バッチ処理と SGD の両方の良い点を用いることができると期待できる。

3.3 CNN

CNN (Convolution Neural Network) は、MLPにおいて全結合層を畳み込み層とプーリング層の繰り返し構造に置換した順伝播型のニューラルネットワークである。2012年のILSVRCで2位以下に10%以上の画像の認識精度の差をつけて優勝したAlexNet [8]はその著名な例である。また、AlexNetは図3.4のような単純なネットワーク構造を持ち、5層の畳み込み層（うち3層ではMax Poolingを行う）と3層の全結合層のみから構成される。

3.3.1 CNNの構造

CNNはネオコグニトロン[9]を起源に持ち、脳の視覚野の機能を模倣するようなネットワーク構造を持つ。また、脳の視覚野には主に二つの機能がある。一つ目は刺激の位置により異なった位置のニューロンの活動電位が発生することであり、二つ目は視覚情報の特徴を段階的に受容することである。この二つの機能を畳み込み層とプーリング層によりCNNは表現する。

まず、畳み込み層では、人工ニューロン間の結合を局所領域に限定することで局所領域の特徴を抽出する。したがって、局所領域間の位置関係は変わらないため、前者の機能を模倣することができる。そして、プーリング層では、局所領域の人工ニューロンの出力をまとめることで局所領域の特徴を集約する。したがって、層を進むにつれて高段階の特徴が残るため、後者の機能を模倣することができる。

さらに、CNNはMLPと比べてパラメータ数を減少させることができる。なぜなら、MLPの全結合層では入力の人工ニューロン数と出力の人工ニューロン数の積が一層のパラメータ数となるが、CNNの畳み込み層ではカーネルに含まれるパラメータ数のみが一層のパラメータ数となるからである。また、CNNのプーリング層でも局所領域の人工ニューロンの出力をまとめることで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大きく寄与する。ここで、特微量マップとはそれぞれの層の入力から得られる出力をまとめたもののことである。

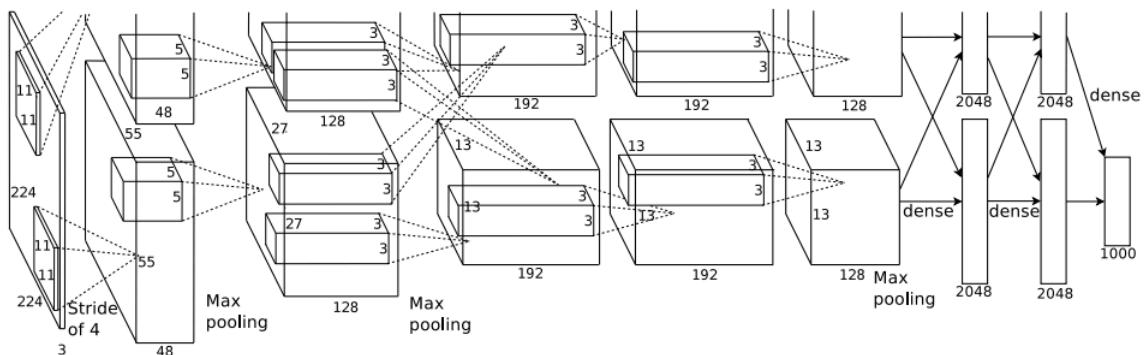


図3.4 AlexNetのネットワーク

3.3.2 疊み込み層

疊み込み層においては、カーネルと呼ばれるフィルターを一定の間隔（ストライド）でスライドさせながらフィルターにより覆われた入力の部分の疊み込み演算を順に行う。疊み込み演算により、カーネルのそれぞれの値を重みとした対応する入力の部分の重み付き和を計算する。また、入力と出力の大きさを変えないために、一般には入力の周りに適当な幅で値を埋める操作を行う（パディング）。そして、画像のような二次元データにおいては、疊み込み演算は図3.5のように表現することができる。

さらに、入力を $I_h \times I_w$ の行列、カーネルを $F_h \times F_w$ の行列、ストライドを S_h, S_w 、パディング幅を P_h, P_w 、出力を $O_h \times O_w$ の行列、とすれば、式3.9が成り立つ。図3.5においては、 $I_h = I_w = 4, F_h = F_w = 3, S_h = S_w = 1, P_h = P_w = 1$ であるため、 $O_h = O_w = 4$ となる。

$$O_h = \frac{I_h + 2P_h - F_h}{S_h} + 1 \quad (3.9)$$

$$O_w = \frac{I_w + 2P_w - F_w}{S_w} + 1 \quad (3.10)$$

また、上記では入力と出力がいずれもチャンネル数が1つの場合を考えたが、実際には複数存在することが多い。したがって、カーネルの数は（入力チャンネル数）×（出力チャンネル数）である。また、それぞれの出力チャンネル数に対し特徴量マップが入力チャンネル数だけ存在するが、これらの和にバイアス項を加えたものが実際の出力となる。

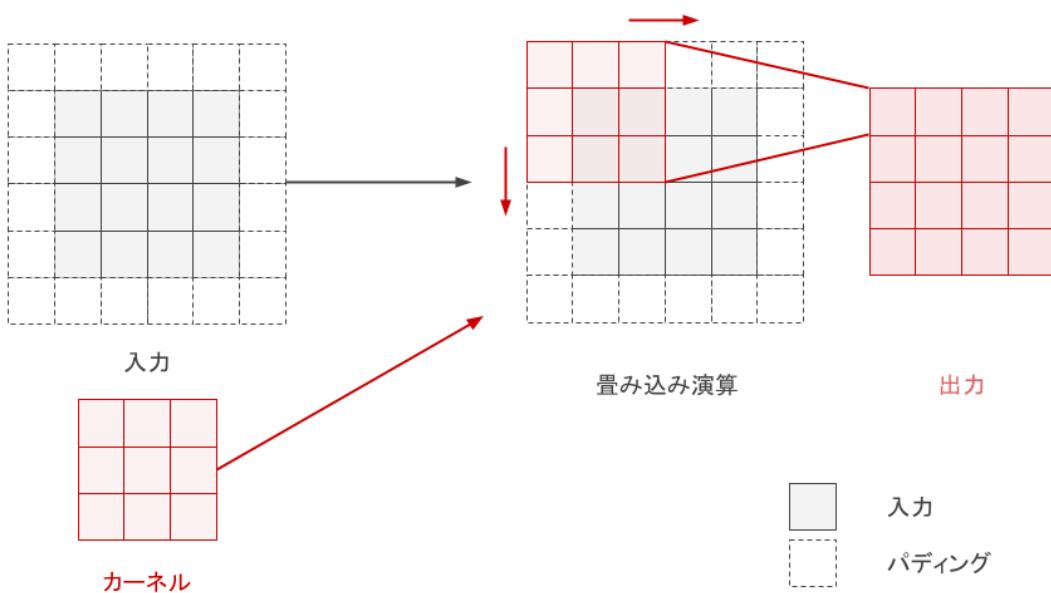


図3.5 疊み込み層
図は[8]のFigure 2を転載した。

3.3.3 プーリング層

プーリング層においては、入力を局所領域ごとに分解し、領域ごとへの何らかの操作により情報量の圧縮を行う（サブサンプリング、ダウンサンプリング）。また、この時の操作で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、主にこの二つが用いられる。

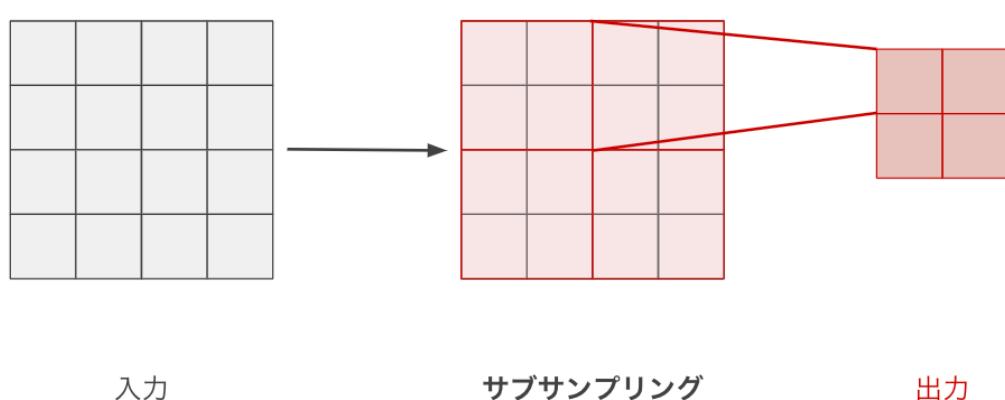


図 3.6 プーリング層

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 [10]。また、GAN は図 3.7 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。

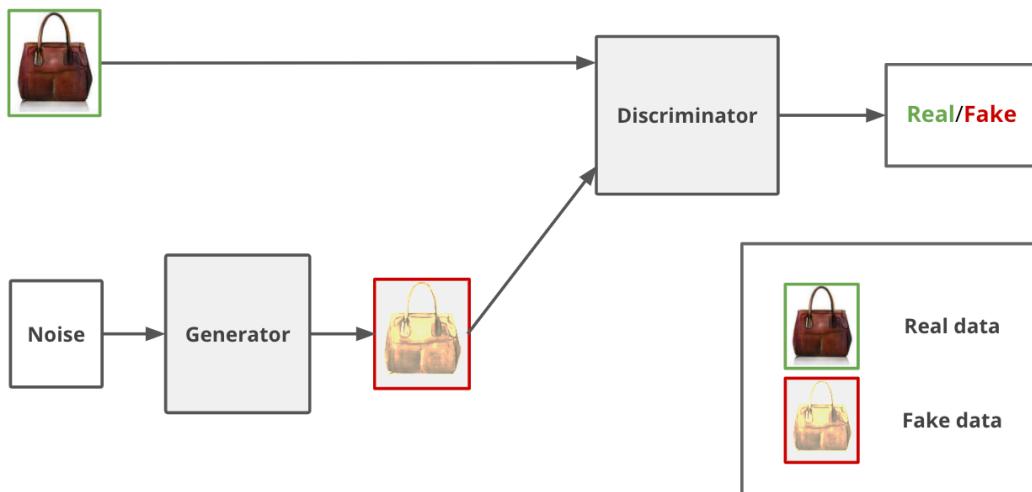


図 3.7 GAN のネットワーク
図は [1] の Figure 1 を用いて作成した。

3.4.2 GAN の定式化

GAN では、生成モデルの目的関数は式 3.11、識別モデルの目的関数は式 3.12 として定式化される。

$$\arg \min_{\theta_G} \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}; \theta_G); \theta_D))] \quad (3.11)$$

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

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、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 (図 3.8)。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) [11] 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、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 Pix2pix により行うことができる (図 3.9)。

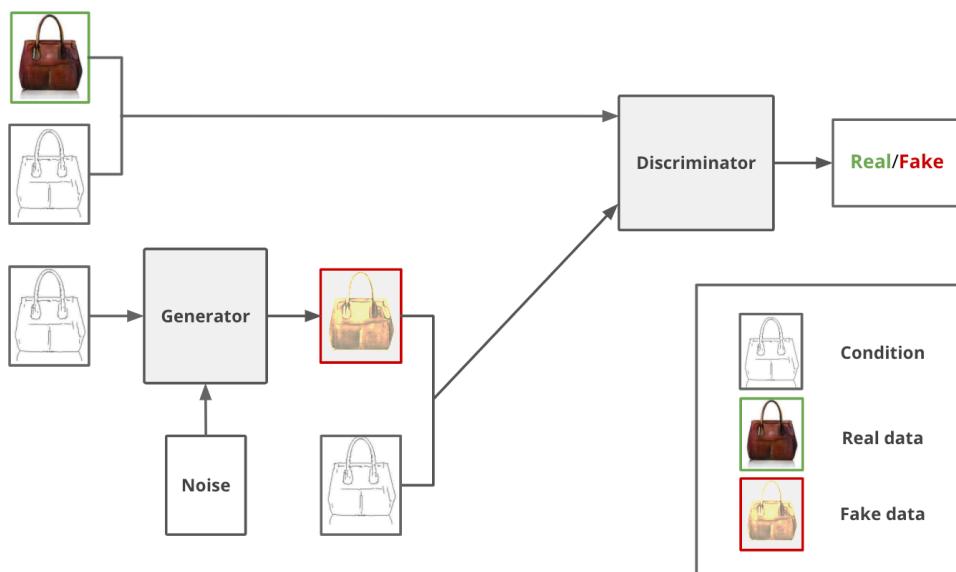


図 3.8 Pix2pix のネットワーク
図は [1] の Figure 1 を用いて作成した。



図 3.9 Pix2pix のスタイル変換の例
図は [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 3.13、識別モデルの目的関数は式 3.14 として定式化される。

$$\arg \min_{\theta_G} \mathbb{E}_{y,z} [\log(1 - D(y, G(y, z; \theta_G); \theta_D))] + \mathbb{E}_{x,y,z} [\|x - G(y, z; \theta_G)\|_1] \quad (3.13)$$

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

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、 $D(y, ; \theta_D)$ 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 3.10 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 [12] 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、ノイズとしては通常の GAN のようなベクトルではなく Dropout [13] 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 として無視する手法のことである。

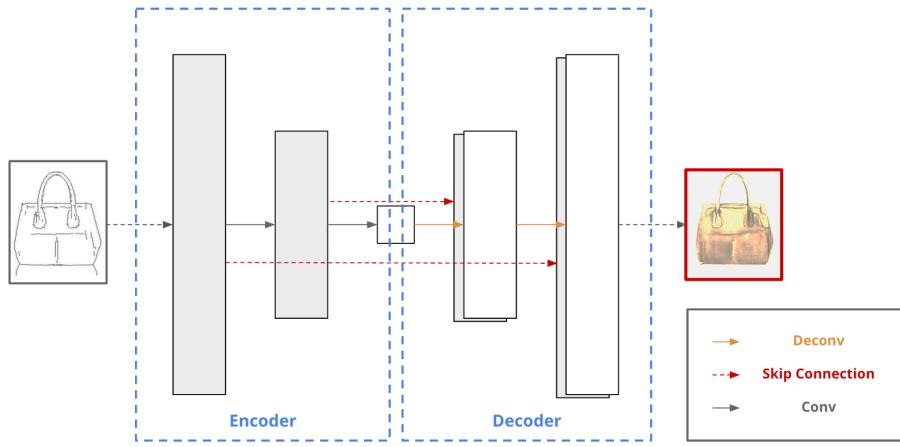


図 3.10 生成モデルのネットワーク
図は [1] の Figure 1 と Figure 3 を用いて作成した。

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 3.11 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 3.11 通常の GAN の識別モデルと PatchGAN の比較
図は [1] の Figure 1 を用いて作成した。

第4章 提案手法

本章では、実験で用いるデータセットと提案モデルの説明を行った後、実験結果の考察を行う。

4.1 データセットと音の表現

データセットとして1秒のギターとハープの音の88組を用いた。88音としてはA0～C8の半音を全て選び、これらは一般的な88鍵のピアノのそれぞれの鍵盤の音に対応する(図4.1)。また、ギターとハープを変換対象とした理由は、弦楽器という共通点を持ちながらも人間の耳で十分に異なると判定できる楽器であると考えられたからである。

そして、44100 Hzのサンプリング周波数でサンプリングを行い、音響信号を圧縮せずに保持するWAV形式によりデータを生成した。また、量子化ビット数は16ビット、チャンネル数は1である。したがって、本研究で用いる音は44100の長さを持つ16ビット整数の一次元配列として表現される。

4.2 提案モデル

本研究では、Pix2pix [1]を元に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することで音色の変換を行うGANを提案モデルとして作成した(図4.2)。また、このモデルでは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。

そして、本節の図の灰色の箱は複数のチャンネルを持つ特微量マップである。箱の上側にチャンネル数を示し、箱の左側に特微量マップとなる一次元配列の長さを示す。また、ネットワーク構造の下にそれぞれの矢印の操作の内容を示す。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。

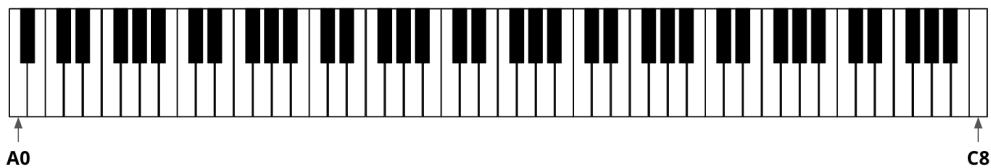


図4.1 88鍵のピアノの鍵盤

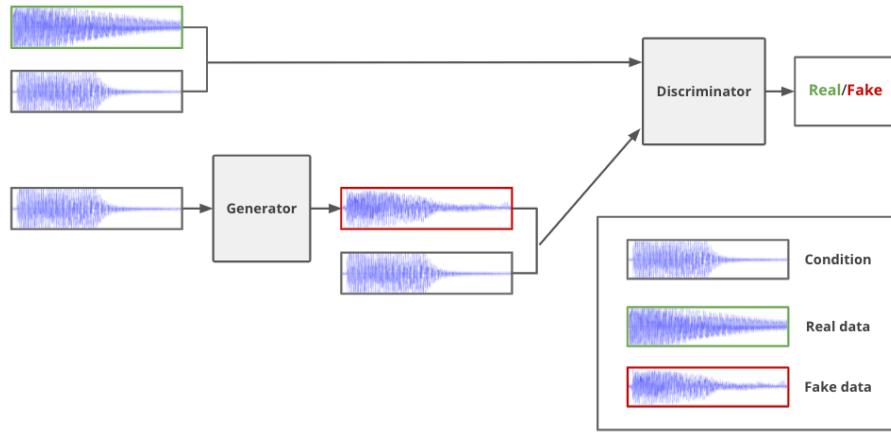


図 4.2 提案モデル

4.2.1 生成モデル

生成モデルには 1 つのスキップコネクションを持つ Encoder-Decoder 型のネットワークを用いた。また、入力は条件となる変換元の音波である (図 4.3)。

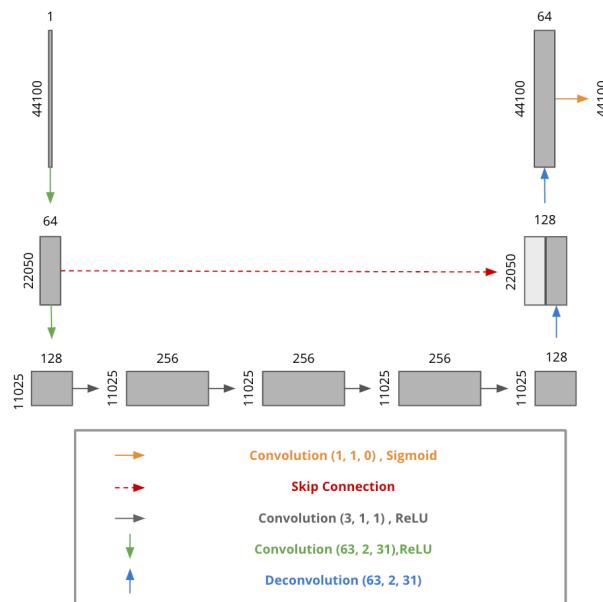


図 4.3 生成モデル
図は [12] の Figure 1 を参考に作成した。

4.2.2 識別モデル

識別モデルには一般的な CNN を用いた。また、出力は最終層の特徴量マップの平均値である（図 4.4）。

4.3 実験方法

4.2 節の提案モデルを用いてギターからハープへの単音の変換の実験を行った。また、最適化アルゴリズムとして用いた Adam [14] のハイパーパラメータを付録 B に示す。

4.3.1 提案モデルの表現力の評価実験

提案モデルの表現力を評価する実験を行った。具体的には、用意した 88 音を学習データと評価データのいずれでも使用し、同じ高さの音の間での変換の実験を行った。

4.3.2 提案モデルの汎化能力の評価実験

単色の音色変換であっても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ため、提案モデルの汎化能力を評価する必要がある。ここでは、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 分割交差検定により実験を行った。また、データセットの分割方法は付録 C に示す。

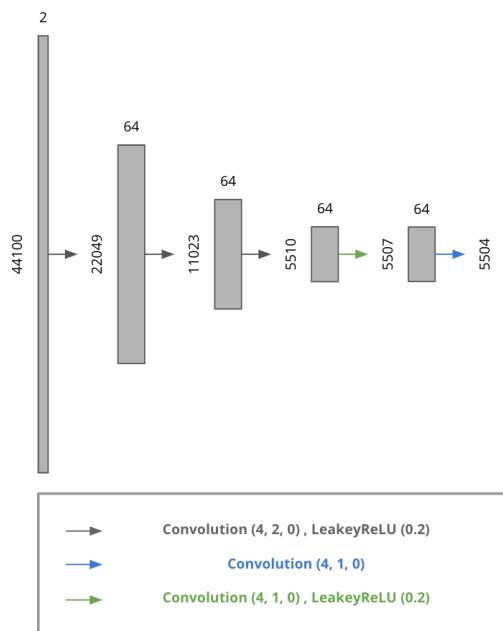


図 4.4 識別モデル
図は [12] の Figure 1 を参考に作成した。

4.3.3 評価方法

生成した音を音波の波形の観察と音の聴き取りにより考察することで評価した。また、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。

4.3.4 データ拡張

各エポックの任意の学習データの振幅を無作為化することでデータ拡張を行った。具体的には、学習前に乱数のシードを固定した後に一様乱数により 0.3 ~ 1.0 の乱数を生成した。また、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を期待した。

4.4 実験結果

実験結果の考察を本節では行う。また、実験結果に記載する波形の図は三つの波形を上から並べている。これらは上から順に、変換元のギターの波形、生成モデルの出力波形、変換先のハープの波形、である。そして、本節では一部の音波のみを記載しているが、付録 E に他の音波の波形を記載している。

4.4.1 概要：提案モデルの表現力の評価実験

提案モデルの表現力の評価実験を行ったところ、実験結果は二つに大別された。

4.4.1.1 ハープの音を表現できた場合

88 音のうち 86 音はハープの音を表現することができた。(図 4.5)。また、特に C4 から D5♯ は目標のハープの音に極めて近い音を生成することができた(図 4.6)。これらの音は上音が少ないとみられ、他の音と比べて表現が容易であったと推察される。

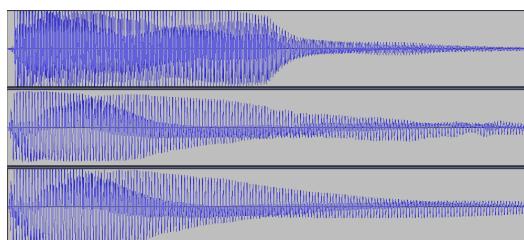


図 4.5 F3 の 0.800 秒から 1.000 秒までの音波

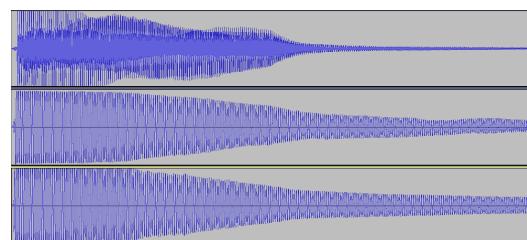


図 4.6 C4 の 0.200 秒から 0.300 秒までの音波

4.4.1.2 ハープの音を表現できなかった場合

D7♯とE7の2音は音の高さは維持できたものの、ハープの音を十分に表現できなかった（図4.7と図4.8）。いずれの音についてもハープの音波の振動が安定しておらず、この不安定さを表現することはできなかった。また、他の高音においても振動が不安定なものが見られたため、高音において安定したデータセットをハープで生成するのが難しいと推察される。

4.4.2 概要：提案モデルの汎化能力の評価実験

提案モデルの汎化能力の評価実験を行ったところ、実験結果は三つに大別された。

4.4.2.1 ハープの音を表現できた場合

D4,D4♯,G4,F5,F5♯の5音はハープの音を表現することができた（図4.9）。これらの高さの音では、提案モデルの表現力の評価実験の際にも特に綺麗なハープの音を表現できており、上音が少ないほど表現が容易であるという推察を示していると思われる。

4.4.2.2 ハープの音を表現できず音の高さも維持できなかった場合

C7,D7♯,E7,F7♯,G7,G7♯,A7,B7,C8の9音は音の高さも維持することができず、騒音が生成された（図4.10）。これらの高さの音では提案モデルの表現力の評価実験の際にもハープの音を表現できておらず、やはり高音域における安定したデータセットの作成は難しいと考えられる。また、高音域では周波数が高くノイズの影響が大きく出たとも考えられる。

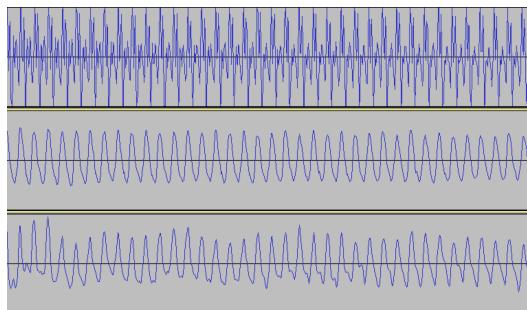


図4.7 D7♯の0.055秒から0.070秒までの音波

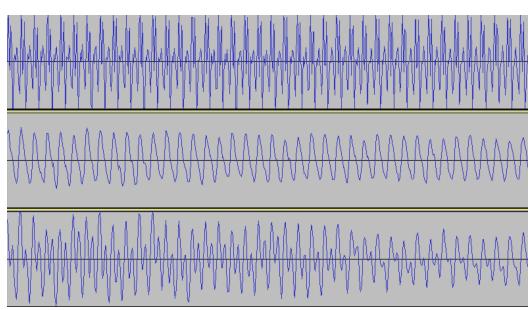


図4.8 E7の0.055秒から0.070秒までの音波

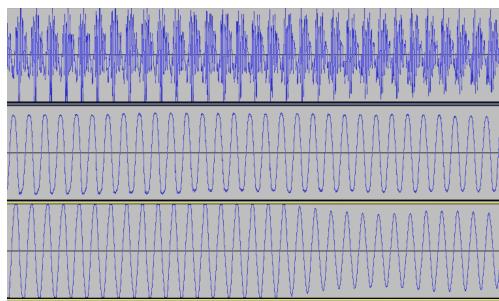


図4.9 D4♯の0.100秒から0.200秒までの音波

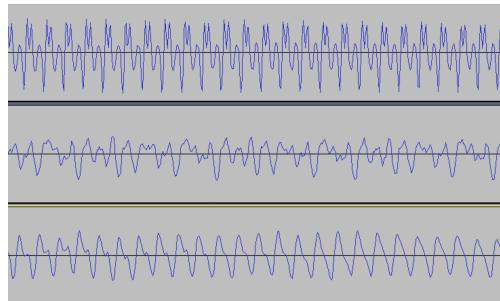


図4.10 D7♯の0.1700秒から0.110秒までの音波

4.4.2.3 ハープの音を表現できず音の高さを維持できた場合

14 音を除く 74 音については音の高さは維持できているもののハープの音を表現することができなかった(図 4.11、図 4.12)。これらの音では、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、これらの高さの音は提案モデルの表現力の評価実験の際には表現することができていたため、提案モデルの汎化能力の低さが明らかになった。

4.4.3 課題

実験結果から四つの課題が明らかになった。

4.4.3.1 音の大きさの維持

部分的に大きさを維持できていない音がいくつかあった。図 4.13 では、波形の前半において生成波形の振幅が小さいことがわかる。原因は振幅をランダムに小さくしたことにあると考えられ、学習の初段階では振幅を固定しておくことや振幅の大きさを別で調整するなどの工夫が必要であると考えられる。

4.4.3.2 音波の滑らかさの表現

ノイズまじりの音がいくつかあった。これらの音を詳細に観察すると、生成された音波では滑らかさを表現できていないことがわかった(図 4.14)。音波を滑らかにするような加工を生成後に加えるなどの工夫が必要であると考えられる。

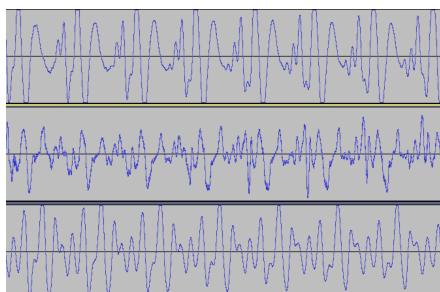


図 4.11 D1 の 0.300 秒から 0.500 秒までの音波

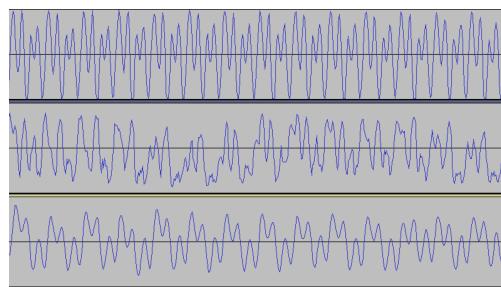


図 4.12 F6 の 0.070 秒から 0.080 秒までの音波

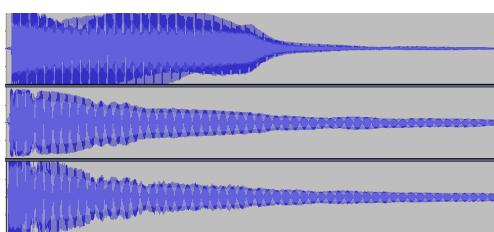


図 4.13 C5 の 0.000 秒から 1.000 秒までの音波

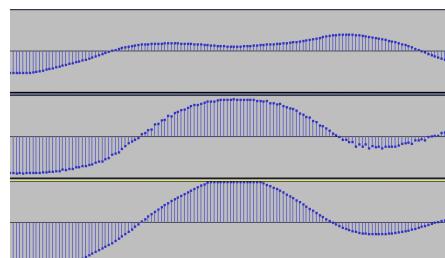


図 4.14 D2♯ の 0.100 秒から 0.103 秒までの音波

4.4.3.3 音の減衰の表現

振動の減衰を表現できていない音がいくつかあった。これらの音のうち、E2以下の低音域ではハープとは全く異なる波形で減衰し(図4.15)、A6以上の高音域ではほとんど振動が見られなかった(図4.16)。また、提案モデルでは表現力が足りず微小な振動の学習が難しいためであると考えられる。

4.4.3.4 データセットの不安定さ

ここまで三つは提案モデルの改善により解消されると考えられるが、問題のあるデータセットも一部に存在した。一つの問題点は高音において振動が不安定になることであり、ここまで述べた。もう一つの問題点は音の鳴り出しでの振動が不安定であるという点である。具体的には、図4.17のようにギターの音の鳴り出しの遅延の方がハープより大きい場合や図4.18のように周期的な音になるまでに遅延があるために不規則な振動となる場合などがあった。



図4.15 A0 の 0.800 秒から 1.000 秒までの音波

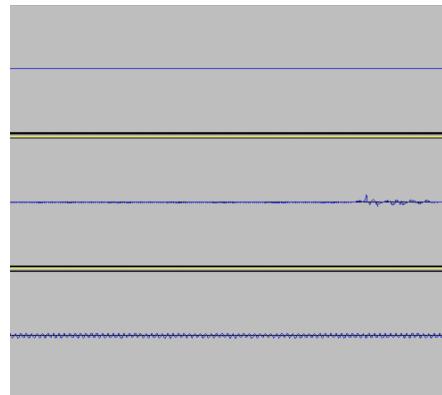


図4.16 B7 の 0.980 秒から 1.000 秒までの音波

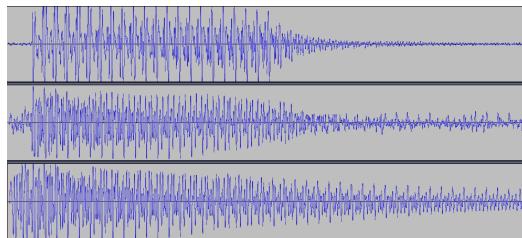


図4.17 F1♯ の 0.000 秒から 1.000 秒までの音波

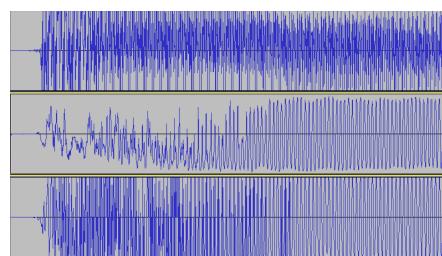


図4.18 A7 の 0.000 秒から 0.030 秒までの音波

第5章 まとめ

本研究での実験の結果、提案モデルは十分な表現力を持つことを示すことができたが、汎化能力は十分でないことも同時に明らかになった。提案モデルの生成モデルにおいて Pix2pix のような Dropout 層を用いていない点やデータ拡張が十分に機能しない点が汎化能力を十分ではない主な原因であると考えられる。また、音波の細かな部分の観察により表現力も十分ではないとも推察される。そして、4.4 節にあるように、音の大きさの維持、音波の滑らかさの表現、音の減衰の表現、データセットの不安定さ、の四点の課題も残った。

さらに、本研究では波形の観察を中心とした考察を行ったが、本研究の考察を明確化するためにはより定量的な判定を行う必要がある。具体的には、音の高さの維持、音の大きさの維持、音色の変換、という三点での定量的な判定を行う必要がある。

また、汎化能力の向上、課題の解決、定量的な判定法に取り組んだ後に音楽への適用を考える必要があるが、Google の Magenta プロジェクトにより発表された DDSP [15] は特徴量として二次元データを用いることでこれらの問題を解決している。DDSP の説明は付録 D にて行う。

謝辞

本研究を進める際にご指導をして頂いた指導教官の金子知適准教授にまずは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や試問教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] Keunwoo Choi, György Fazekas, Kyunghyun Cho, and Mark Sandler. A tutorial on deep learning for music information retrieval, 2018.
- [4] A scale for the measurement of the psychological magnitude pitch: The journal of the acoustical society of america: Vol 8, no 3. <https://asa.scitation.org/doi/10.1121/1.1915893>. (Accessed on 02/10/2021).
- [5] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, Vol. abs/1609.04747, , 2016.
- [6] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.
- [7] James J. Callahan, 2010.
- [8] 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.
- [9] 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.
- [10] 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).
- [11] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014.
- [12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.
- [13] 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.
- [14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
- [15] Jesse Engel, Lamtharn (Hanoi) Hantrakul, Chenjie Gu, and Adam Roberts. Ddsp: Differentiable digital signal processing. In *International Conference on Learning Representations*, 2020.

付録 A Adam

Adam は勾配降下法の中で最も用いられるアルゴリズムであり、Algorithm 1 に示す手順にしたがってパラメータの更新を行う。また、Adam については [14] に詳しい。

HyperParameter : $\beta_1, \beta_2 \in (0, 1], \eta, \epsilon$

Objective Function : f

Parameter : θ

Initialization:

```

|    $t \leftarrow 0$                                 /* Initialize timestep */
|    $\theta \leftarrow \theta_0$                           /* Initialize Parameter */
|    $m_1 \leftarrow 0$                                 /* Initialize 1st moment */
|    $m_2 \leftarrow 0$                                 /* Initialize 2nd moment */
|
| end

```

Procedure:

```

while  $\theta$  not converged do
|    $t \leftarrow t + 1$                                 /* Update timestep */
|    $g \leftarrow \nabla_{\theta} f(\theta)$                   /* Compute gradient of  $f(\theta)$  */
|    $m_1 \leftarrow \beta_1 \cdot m_1 + (1 - \beta_1) \cdot g$  /* Update biased 1st moment */
|    $m_2 \leftarrow \beta_2 \cdot m_2 + (1 - \beta_2) \cdot (g \odot g)$  /* Update biased 2nd moment */
|    $\hat{m}_1 \leftarrow m_1 / (1 - \beta_1^t)$              /* Compute bias-corrected 1st moment */
|    $\hat{m}_2 \leftarrow m_2 / (1 - \beta_2^t)$              /* Compute bias-corrected 2nd moment */
|    $\theta \leftarrow \theta - \eta \cdot \hat{m}_1 / (\sqrt{\hat{m}_2} + \epsilon)$  /* Update Parameter */
|
| end
| return  $\theta$ 
end

```

Algorithm 1: Adam の疑似コード

付録 B 学習時のパラメータ

提案モデルの学習時のパラメータの値を表 B.1 に示す。また、表 B.2 は Algorithm 1 に示すハイパーパラメータの値である。

パラメータ	値
バッチサイズ	1
エポック数	1000

表 B.1

パラメータ	値
β_1	0.5
β_2	0.999
η	0.0002
ϵ	10^{-8}

表 B.2

付録 C データセットの分割

22 音ずつの 4 つのサブセットにデータセットを分割した (図 C.1)。また、データセットの 4 分割は、88 音をシャッフルして配列に格納した後に 22 音ずつ順に選ぶことで実装した。

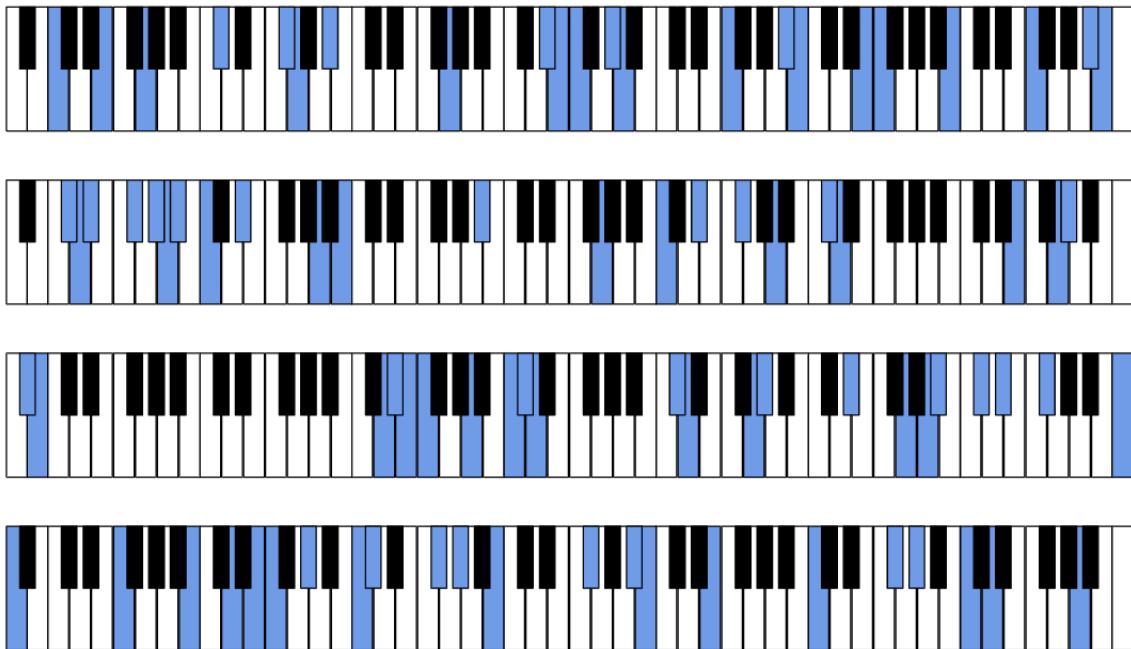


図 C.1 データセットのサブセット

付録 D DDSP

付録 E 実験結果

4.4 節に載せることのできなかった波形の図を本章に載せる

E.1 提案モデルの表現力の評価実験

提案モデルの評価実験を行ったところ、88 音のうち 87 音については変換先のギターの音を表現できていると判断することができた。

E.2 提案モデルの汎化能力の評価実験