Auto-Encoder,U-netと音源分離

Deep-people #8

Tsugumasa Yutani

参考

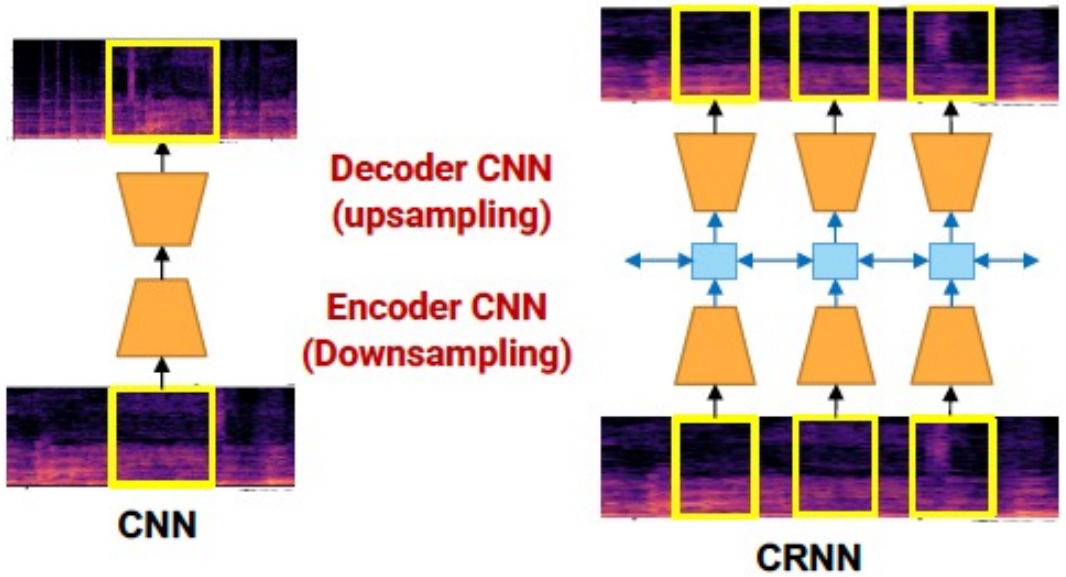
- Musical Applications of Machine Learning
https://mac.kaist.ac.kr/~juhan/gct634/index.html

- [week11-1] AE, U-Net, and source separation
https://mac.kaist.ac.kr/~juhan/gct634/Slides/[week11-1]%20AE,%20U-Net,%20and%20source%20separation.pdf

Juhan先生の講義資料を一部改変して直訳しました
その他参考文献・リンク等は適宜スライド内に記載しています。

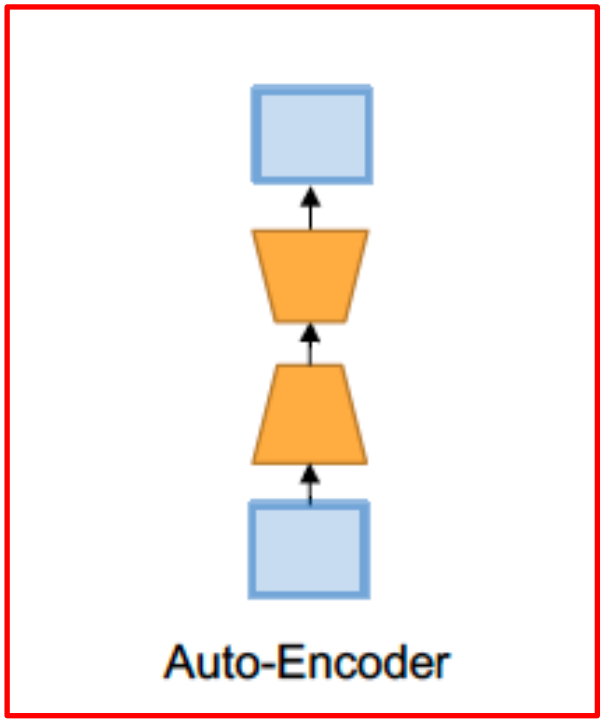
はじめに

- これまでの勉強会では、分類タスクに対してCNNやRNNを使用
  - オーディオからラベルやスコア(MIDI,ビート,コード)の特徴抽出を目的としていた
- 出力をオーディオ信号にしたい場合はどうすればよいか
  - 例えば、音源分離やオーディオスタイル変換をするには？
  - 特徴空間をオーディオ信号に戻すデコーダー(アップサンプリング)が必要

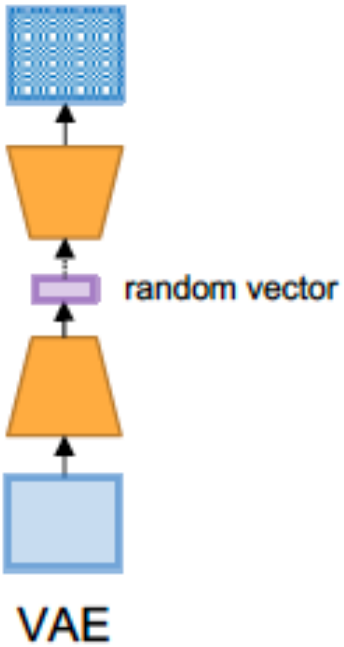


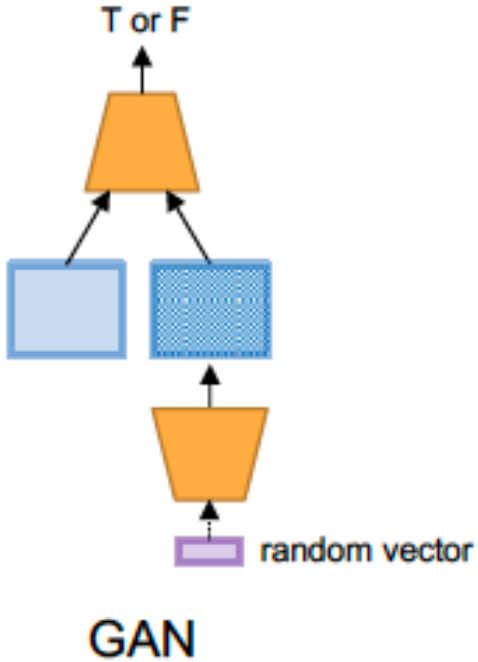
Encoder-Decoder model

- Encoder-Decoder の代表的なモデル
  - Auto-encoder(AE), Variational auto-encoder(VAE), Generative adversarial network(GAN)など



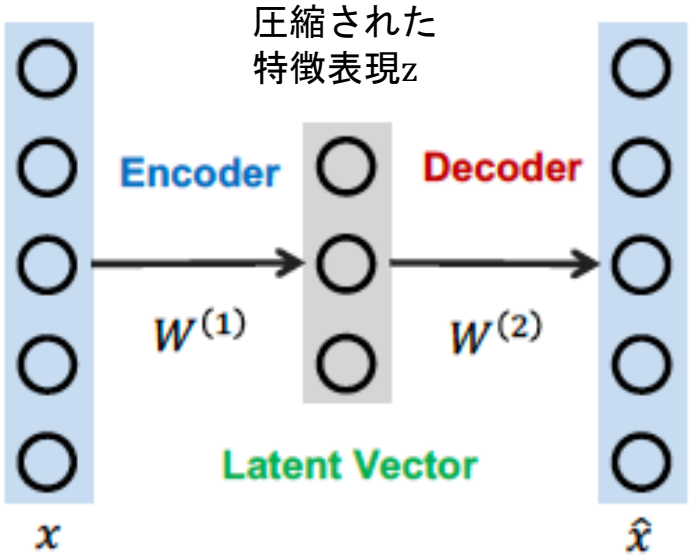
↑今回はここを説明





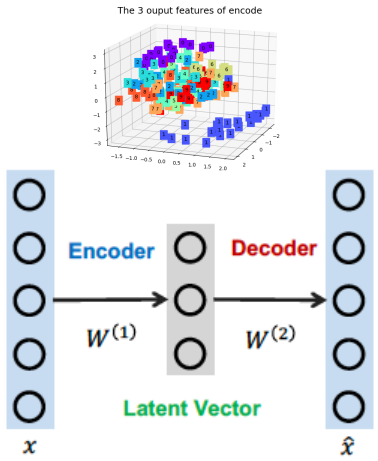
Auto-Encoder(AE)

- 入力を再構成する様に構成されたニューラルネットワーク(教師なし学習)
  - エンコーダー：入力を潜在的なベクトルに変換する（次元圧縮）
  - デコーダー：潜在ベクトルから入力を再構成する
  - 再構成誤差を最小化するように回帰的に学習させる (損失関数 $l(W;x) = \left|\left|x - \hat{x}\right|\right|^2$)



Auto-Encoder(AE)

- 入力を再構成する様に構成されたニューラルネットワーク(教師なし学習)
  - エンコーダー：入力を潜在的なベクトルに変換する（次元圧縮）
  - デコーダー：潜在ベクトルから入力を再構成する
  - 再構成誤差を最小化するように回帰的に学習させる (損失関数 $l(W;x) = \left|\left|x - \hat{x}\right|\right|^2$)

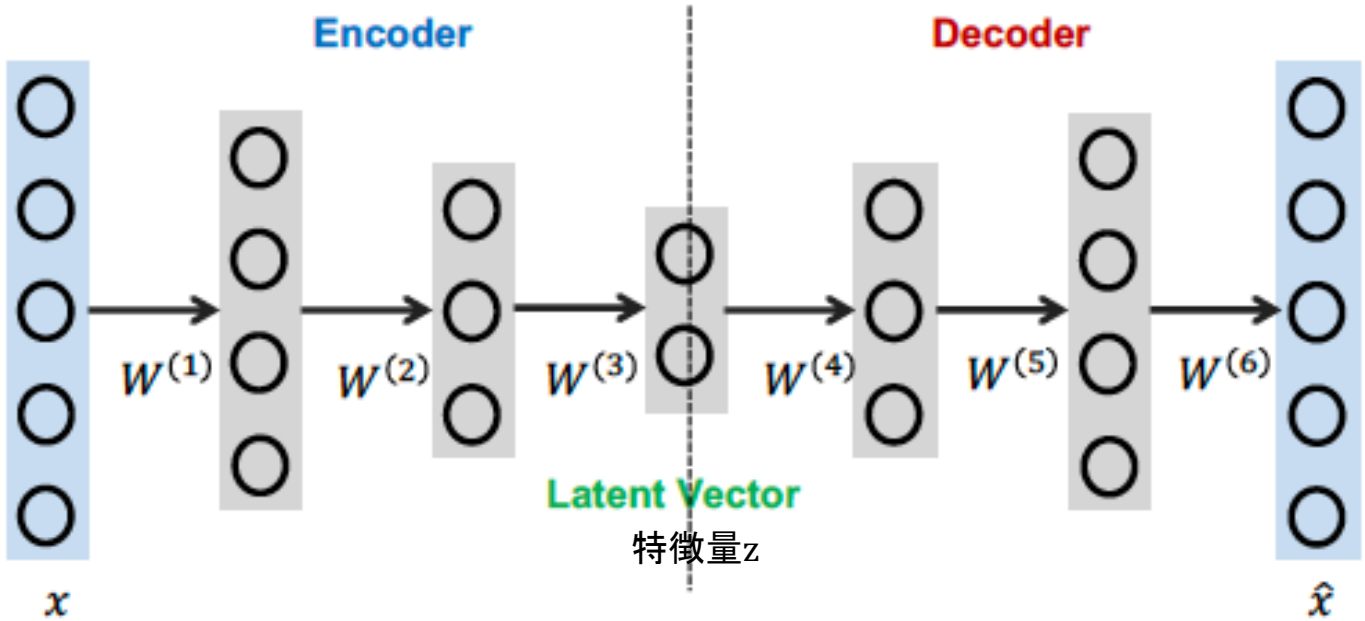


学習のステップ

Deep Auto-Encoder(DAE)

- AEよりも表現力の高い学習することができる
  - 重み行列はEncoder-Decoderで対象な層同士で共有できる

例： $\mathrm{W}^{(6)} = \left(W^{(1)}\right)^T$, $\mathrm{W}^{(5)} = \left(W^{(2)}\right)^T$, $\mathrm{W}^{(4)} = \left(W^{(3)}\right)^T$



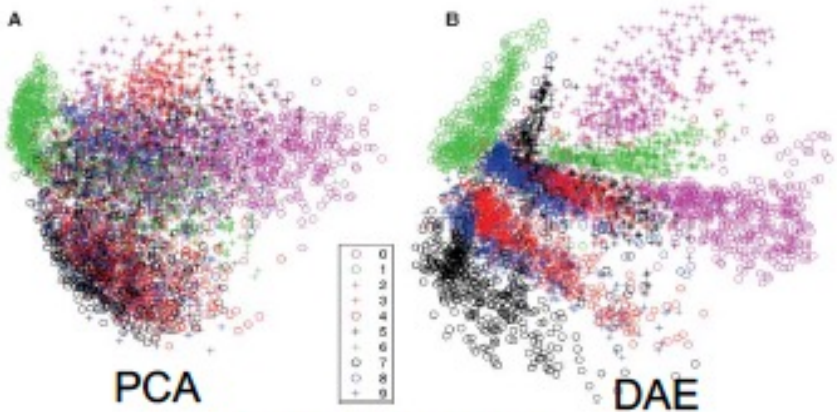
次元削減

- 潜在ベクトルを低次元に設定する事で特徴抽出(データ圧縮)を行うことが出来る
  - DAEは線形モデル（PCA : 主成分分析）よりも複雑なデータを上手く学習できる

Reconstruction

DAE: 28 x 28 → 1000 → 500 → 250→ 30

PCA: 28 x 28 → 30



2D Embedding

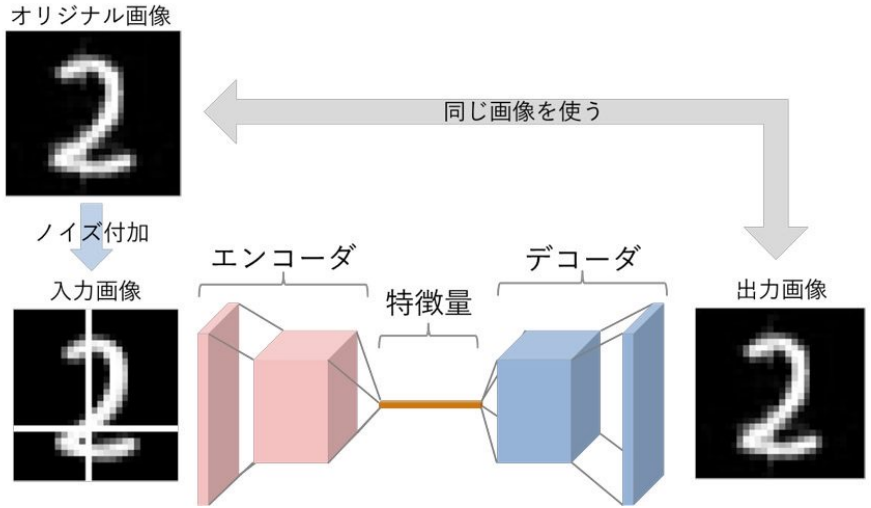
DAE: 28 x 28 → 1000 → 500 → 250→ 2

PCA: 28 x 28 → 2

(Hinton and Salakhutdinov, 2006)

ex. ノイズ除去（デノイジングオートエンコーダ）

- 入力データに含まれるノイズの除去
  - 人為的にノイズを加えたデータを入力として、
    ノイズ追加前のオリジナルのデータを出力するよう学習
  - ノイズが含まれたデータから、ノイズ除去後のデータが取り出せるようになる

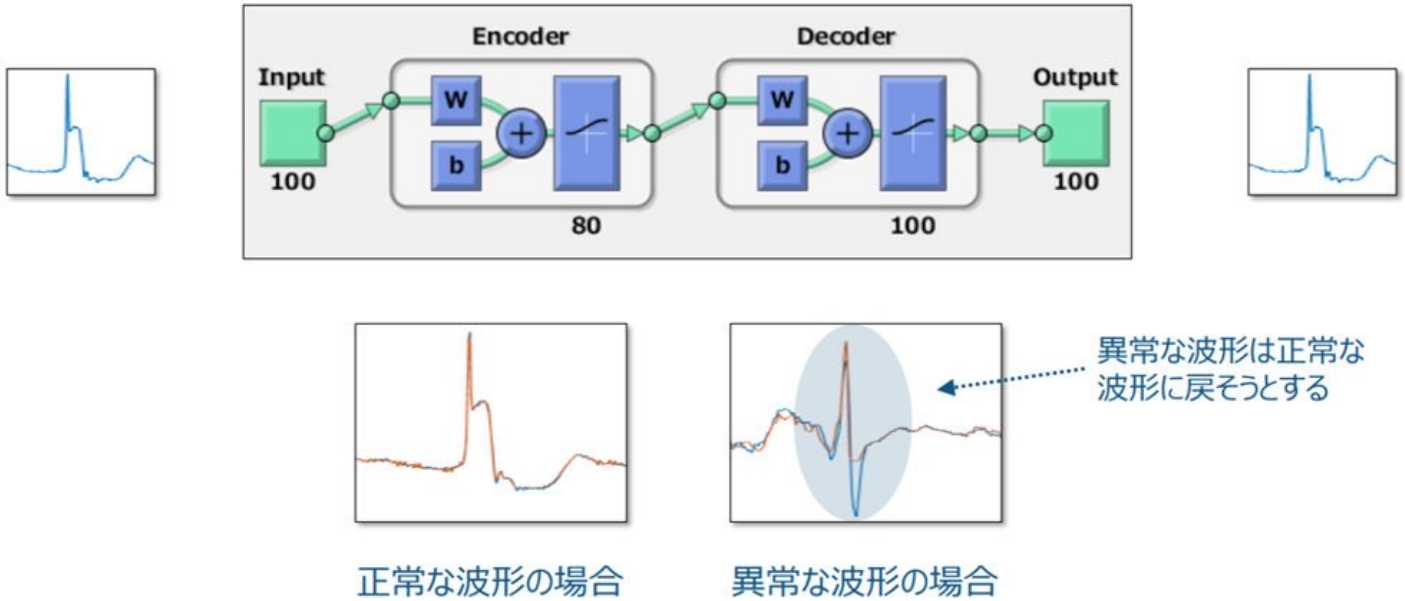


十字状のノイズが含まれた画像に対しての、オートエンコーダを使ったデノイズ例

https://jp.mathworks.com/discovery/autoencoder.html

ex. 異常検知

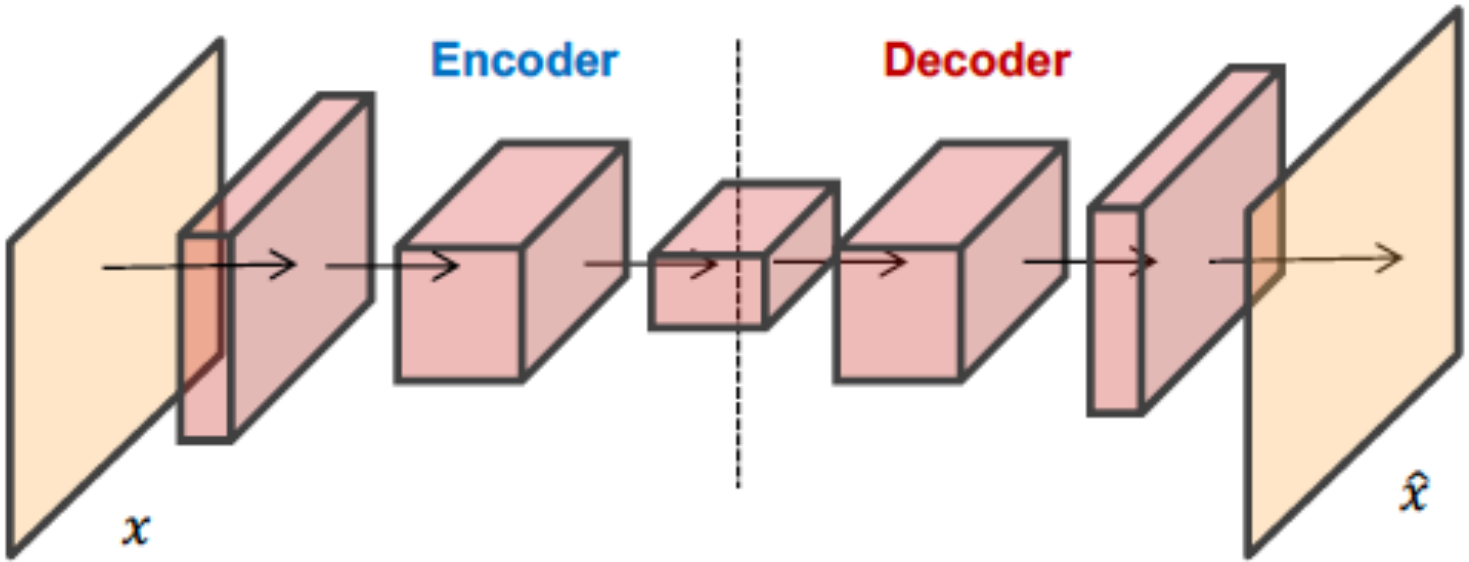
- 入出力に正常データを与えて学習、再構成可能にする
  - 異常データを入力した際、正常な波形に戻そうとモデルは再構成する
  - 入出力の誤差が大きくなるので異常である事がわかる



オートエンコーダを使った時系列信号の異常検知の例 https://jp.mathworks.com/discovery/autoencoder.html

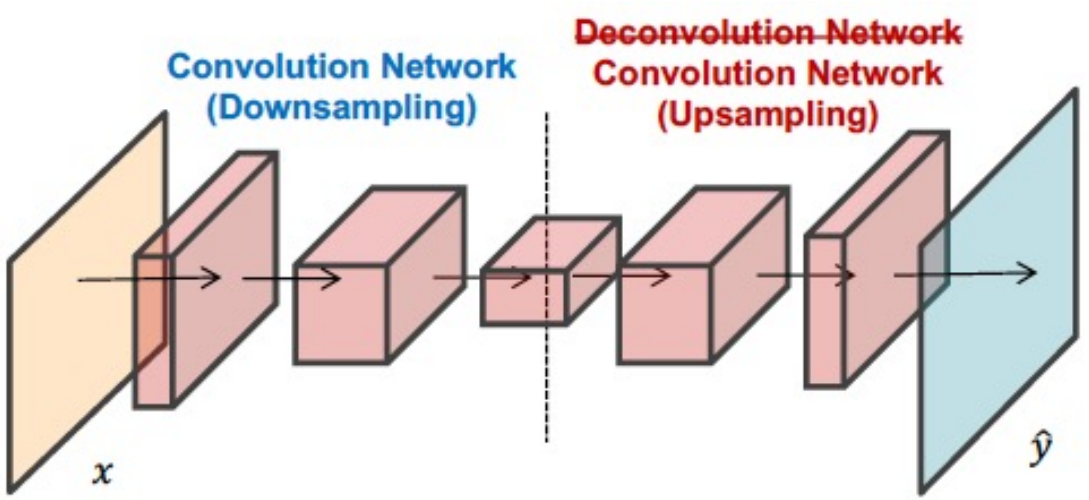
Convolutional Auto-Encoder

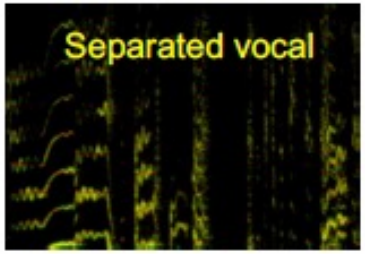
- オートエンコーダーに畳み込みニューラルネットを用いる
  - 高次元の非標識データから圧縮された特徴ベクトルを得るために使用される
  - この教師なし学習は最近あまり人気がないらしい…



Encoder-Decoder model (教師あり学習)

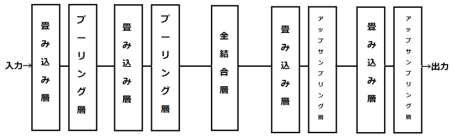
- semantic segmentation、音源分離などで教師あり学習として用いられる
  - ssは画素ごとに独立してクラス識別を行い，画像全体の意味的な領域に分割する問題
  - Deconvolution(逆畳み込み) はよく使われるが、正確ではない
    (入力データを復元するようなものではなく、誤解を招く）



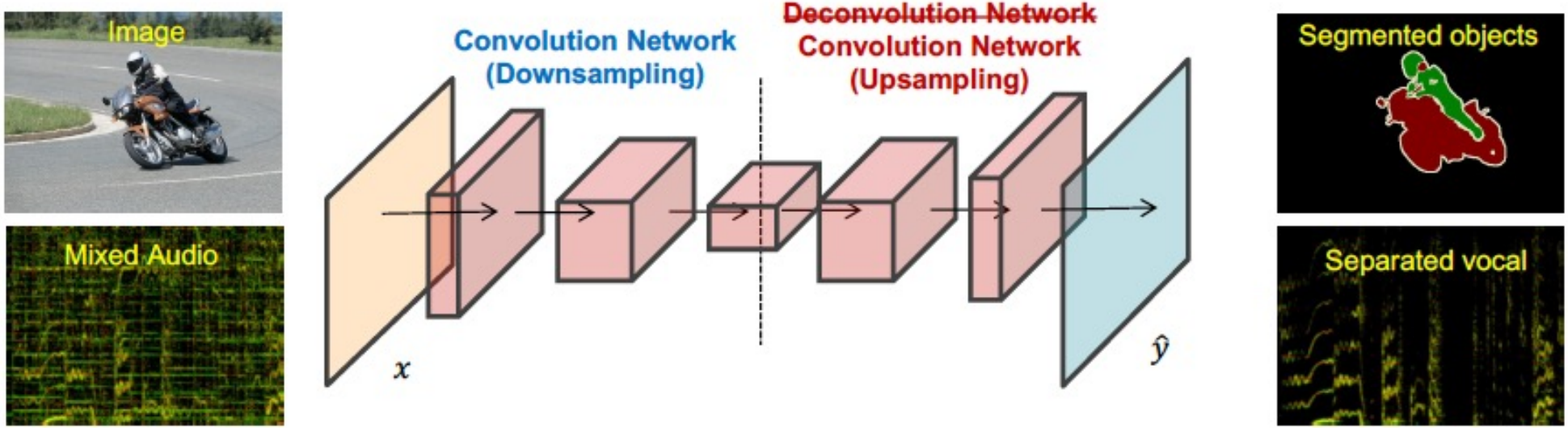


Encoder-Decoder Architecture (教師あり学習)

- Encoder-Decoder アーキ は通常、全層畳み込み(FCN)に設定されている
  - エンコーダー：畳み込み層→プーリング層（ダウンサンプリング）
  - デコーダー：畳み込み層→アンプーリング層（アップサンプリング）

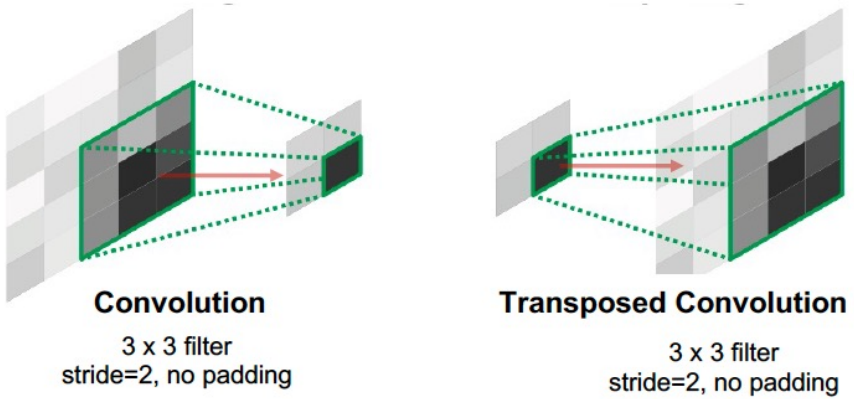


http://www.net.c.dendai.ac.jp/~tamura/

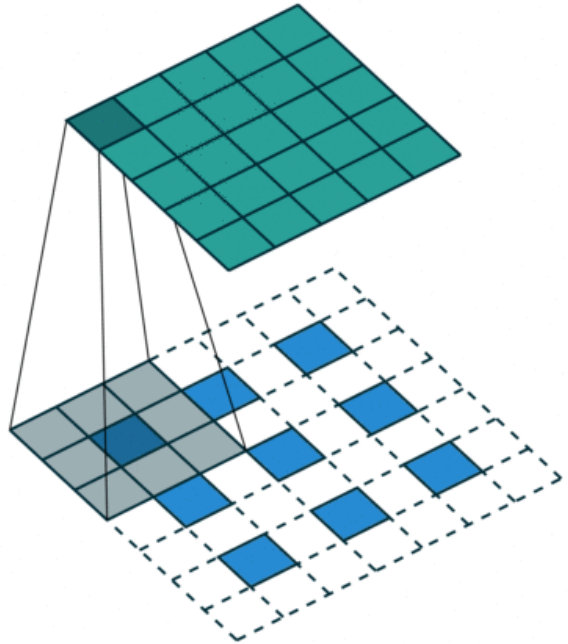


Transposed Convolution(逆畳み込み,転地畳み込み)

- 畳み込みで圧縮したデータをもとのサイズに戻す方法
  - しかし、畳み込みの逆プロセスではなく入力データを完全に復元する訳ではない
  - 入力データを拡大するためにデータを補完してから畳み込みを行う
  - ハイパーパラメータ：stride, padding



Learning Deconvolution Network for Semantic Segmentation, *Hyeonwoo Noh, Seunghoon Hong, Bohyung Han, 2015*

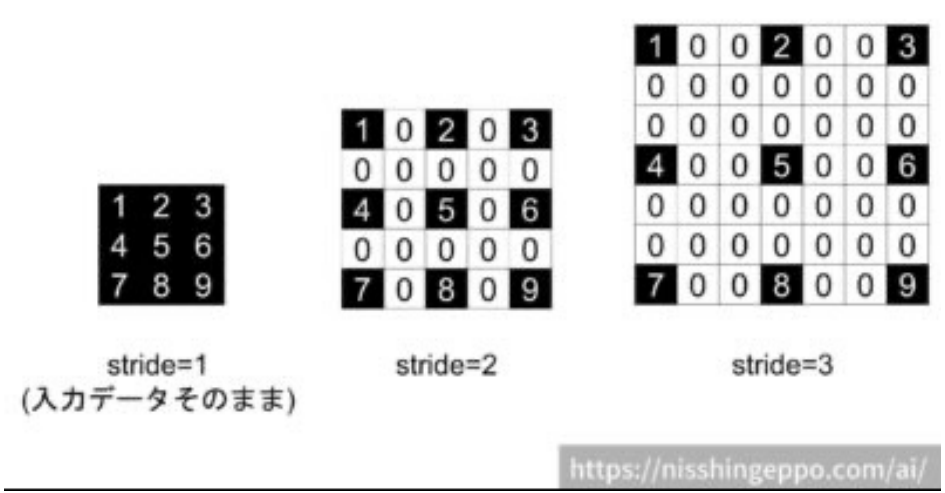


https://github.com/vdumoulin/conv_arithmetic

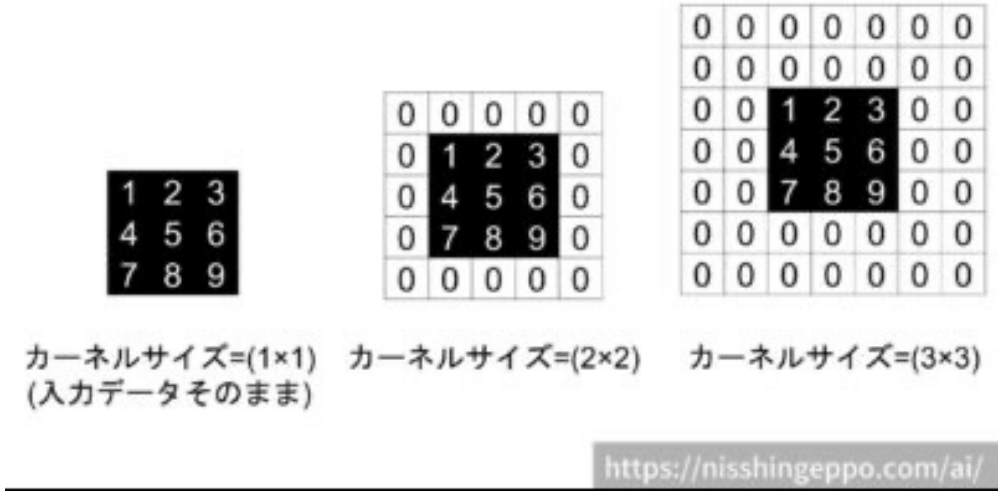
Transposed Convolution (逆畳み込み,転地畳み込み)

●.転置畳み込み(逆畳み込み)の処理

1. strideで指定した行数分(列数)だけ入力データのピクセル間に余白を入れる(3×3の場合)
2. カーネルのサイズより1行(1列)少ない数だけ余白を追加
3. paddingに応じて余白を削る
4. 畳み込み処理を行う



1.strideに応じて入力データを拡張



2.入力データの周囲に余白を取る

Transposed Convolution (逆畳み込み,転地畳み込み)

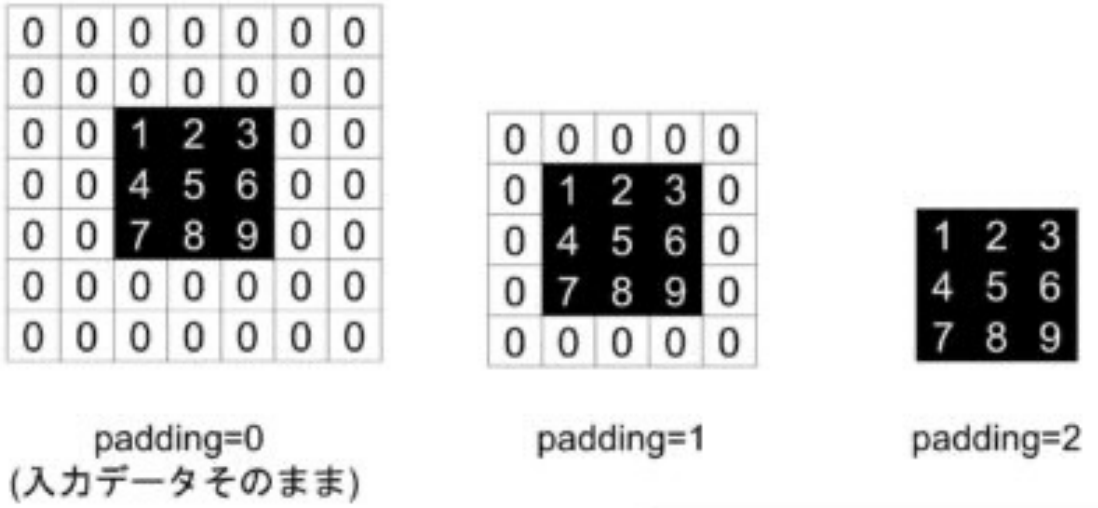
●.転置畳み込み(逆畳み込み)の処理

1. strideで指定した行数分(列数)だけ入力データのピクセル間に余白を入れる(3×3の場合)

2. カーネルのサイズより1行(1列)少ない数だけ余白を追加

3. paddingで指定した行数分(列数)だけ外側の余白を削る

4.畳み込み処理を行う



https://nisshingeppo.com/ai/

~~3. paddingに応じて余白を削る~~

Transposed Convolution (逆畳み込み,転地畳み込み)

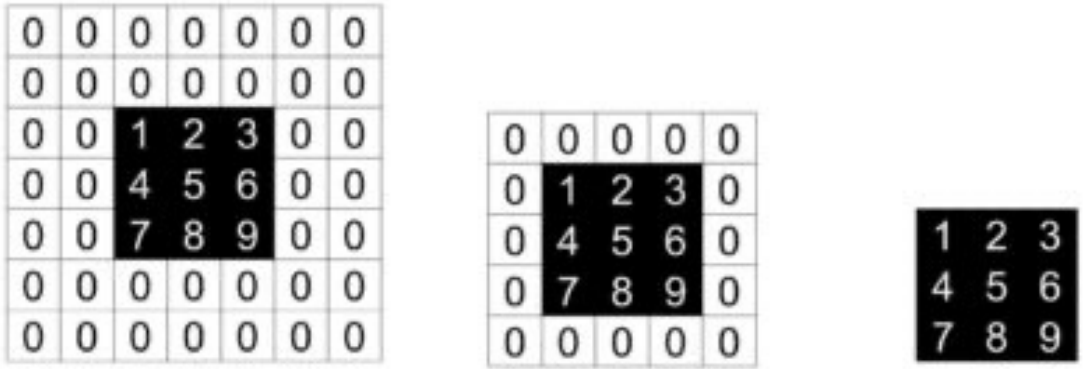
●.転置畳み込み(逆畳み込み)の処理

1. strideで指定した行数分(列数)だけ入力データのピクセル間に余白を入れる(3×3の場合)

2. カーネルのサイズより1行(1列)少ない数だけ余白を追加

3. paddingで指定した行数分(列数)だけ外側の余白を削る

4.畳み込み処理を行う



https://nisshingeppo.com/ai/

~~3. paddingに応じて余白を削る~~

Transposed Convolution (逆畳み込み,転地畳み込み)

入力データ (2×2)　　カーネル (2×2)

https://nisshingeppo.com/ai/

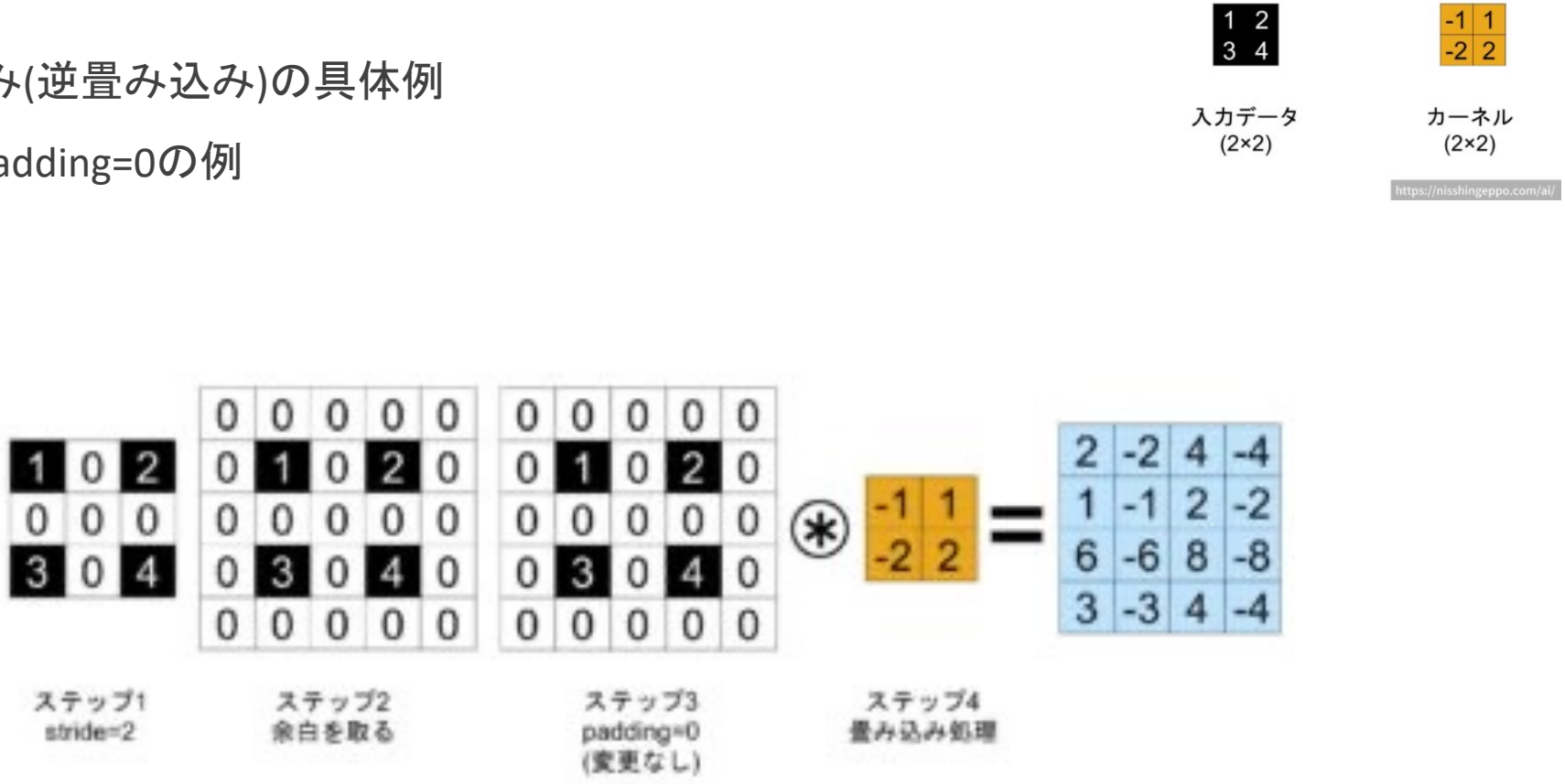
- .転置畳み込み(逆畳み込み)の具体例
  - stride=1, padding=0の例

https://nisshingeppo.com/ai/

Transposed Convolution (逆畳み込み,転地畳み込み)

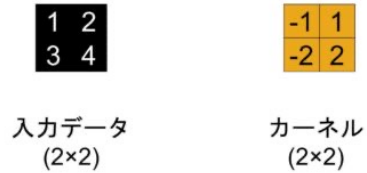
●.転置畳み込み(逆畳み込み)の具体例

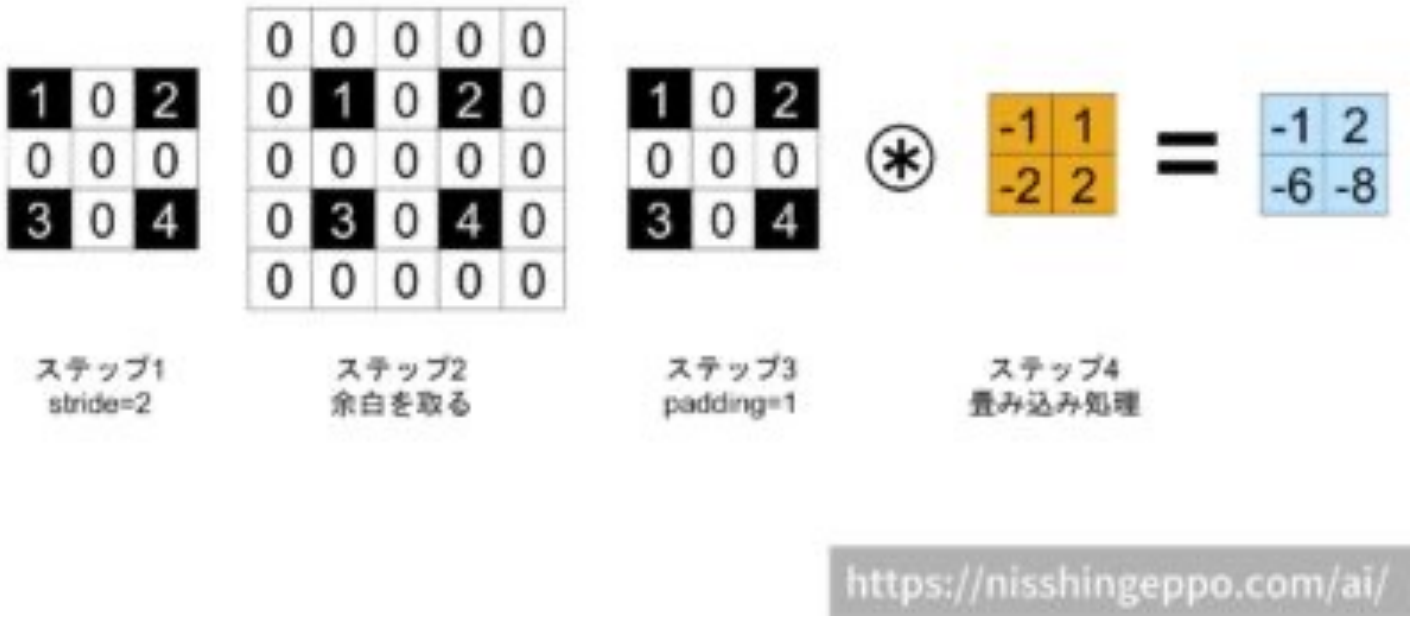
○ stride=2, padding=0の例



Transposed Convolution (逆畳み込み,転地畳み込み)

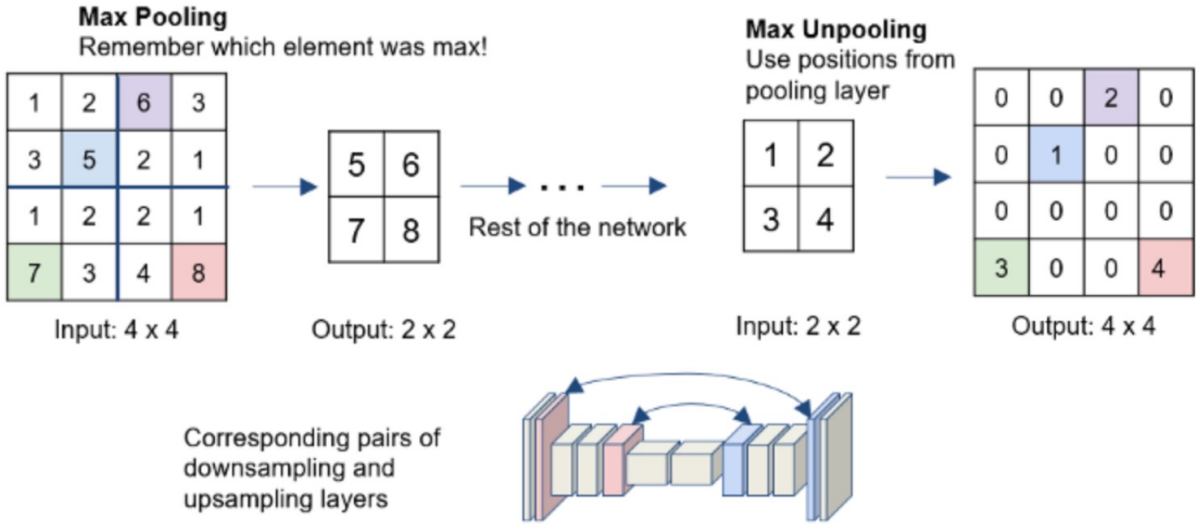
- .転置畳み込み(逆畳み込み)の具体例
  - stride=2, padding=1の例





Unpooling層

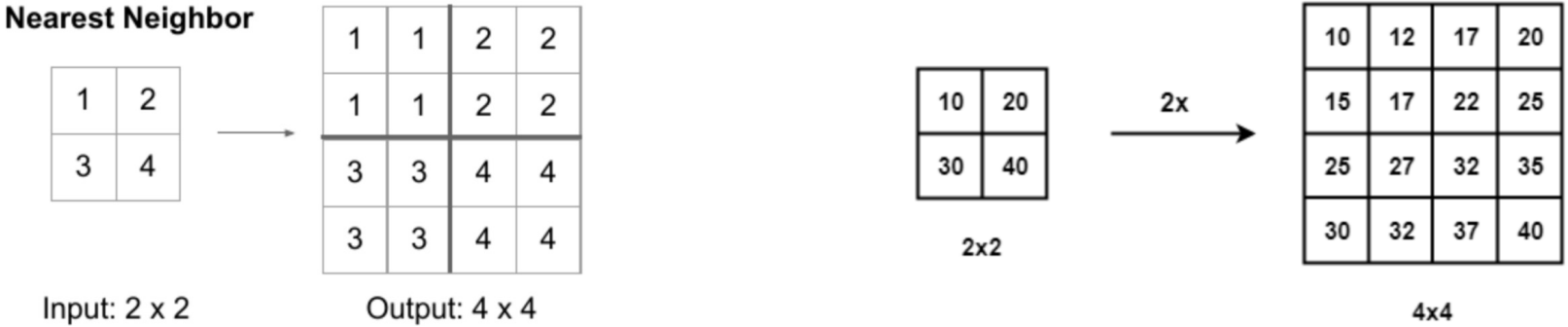
- プーリングを元に戻す(=アンプーリング)
  - encode-decoder CNNではMax-Unpooling（最大値アンプーリング）がよく用いられる
  - Encode部分のMax-Pooling層での最大値を取ったインデックスを保存しておき、
    Decode部分のMax-Unpooling層では位置情報に基づいて最大値を入力し、
    他は全てゼロを埋める。

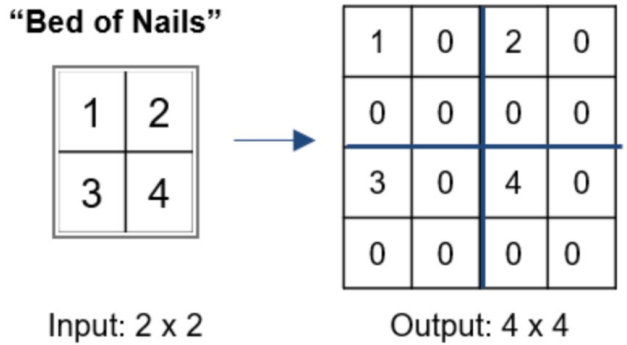


https://qiita.com/cv_carnavi/items/8fb0c795798ee88bd8b2

補足：Unpoolingのその他手法

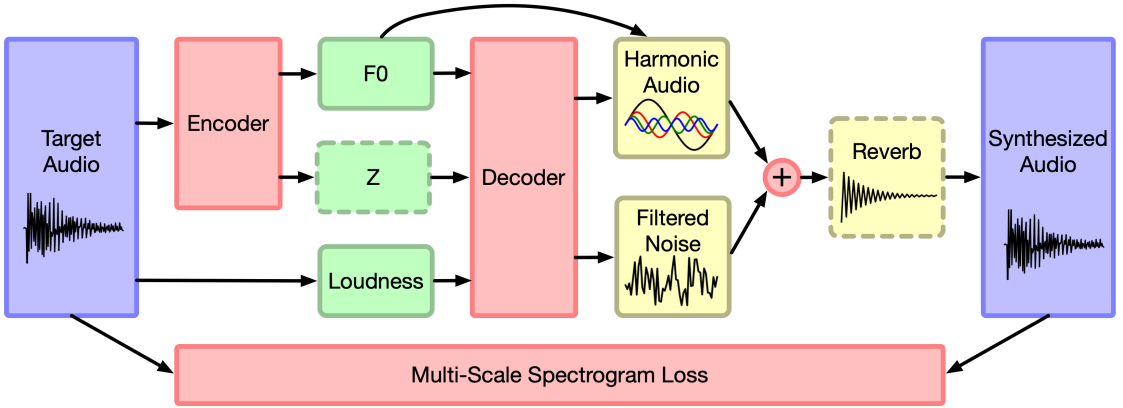
- 最近近傍補完、バイリニア補完、釘埋め等

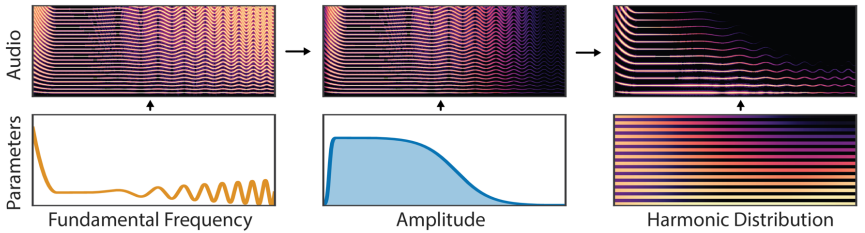




ハンズオン

- Timbre transfer (DDSP: Differentiable Digital Signal Processing)
  - 従来の信号処理要素をend-to-endで学習できるようにした音声生成モデル
  - F0の倍数(整数)の正弦波の合計をオーディオとして生成
  - ニューラルネットから(F0,振幅,高調波分布,ノイズ成分,残響成分)を供給



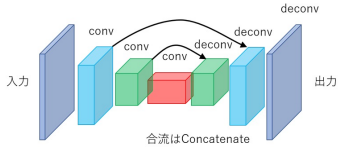


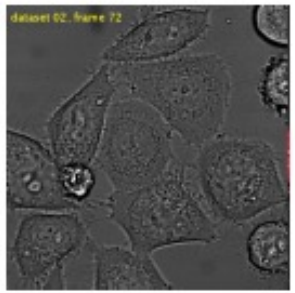
□リンク

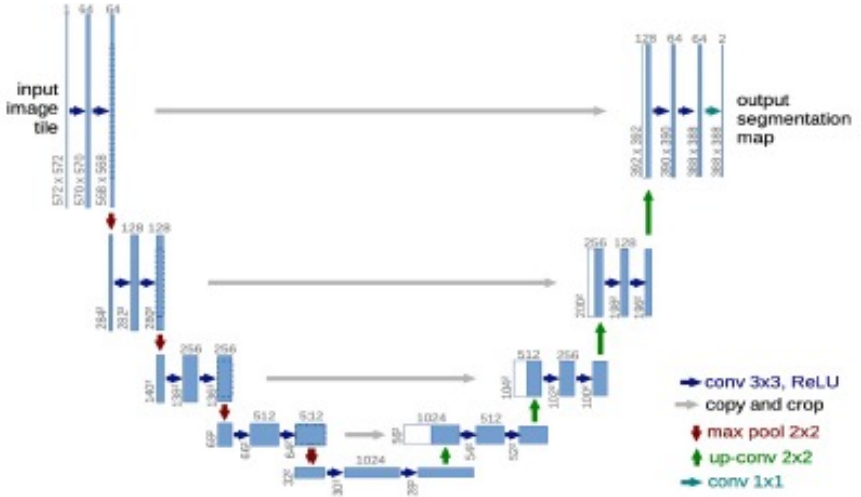
Google Colab、論文、解説

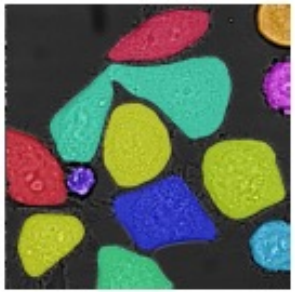
U-net

- セマンティックセグメンテーション等で用いられる
  (画素一つ一つに対してクラスを推測するネットワーク)
  - 対応するエンコーダー層からのスキップ接続により、
    “物体の局所的特徴と全体的位置情報“の両方を統合して学習させる
  - 医療用画像のセグメンテーションに初めて導入









U-Net: Convolutional Networks for Biomedical Image Segmentation: *Olaf Ronneberger, Philipp Fischer, Thomas Brox, 2015*

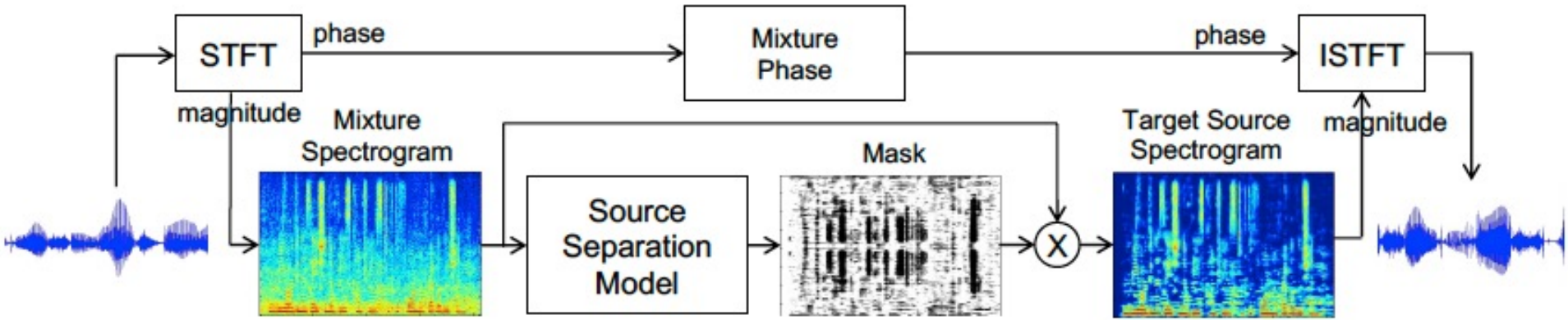
音源分離

- 複数音源がミックスダウンされたミックス音から、個々の音を分離する処理
  - 人間は音響パターン（音色、音域、ADSRなど）により
    個々の音源を認識することができる

Source: https://source-separation.github.io/tutorial/landing.html

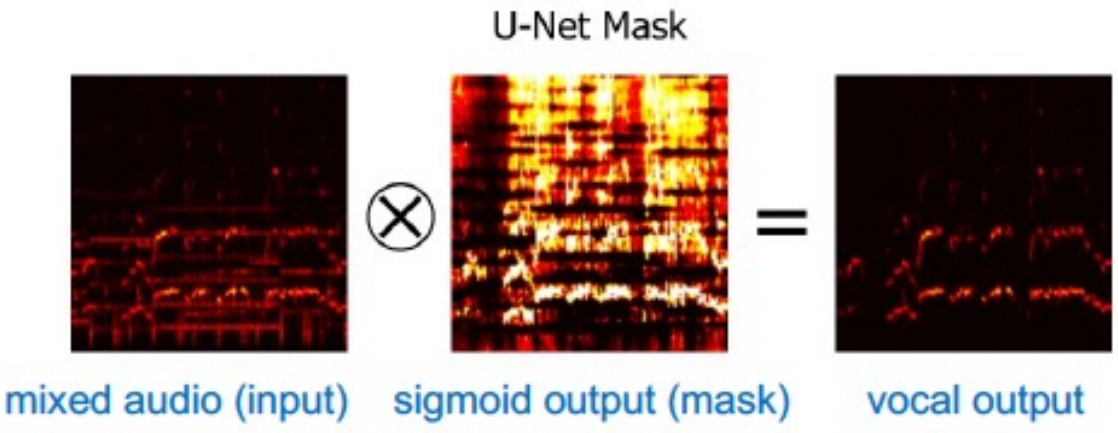
音源分離

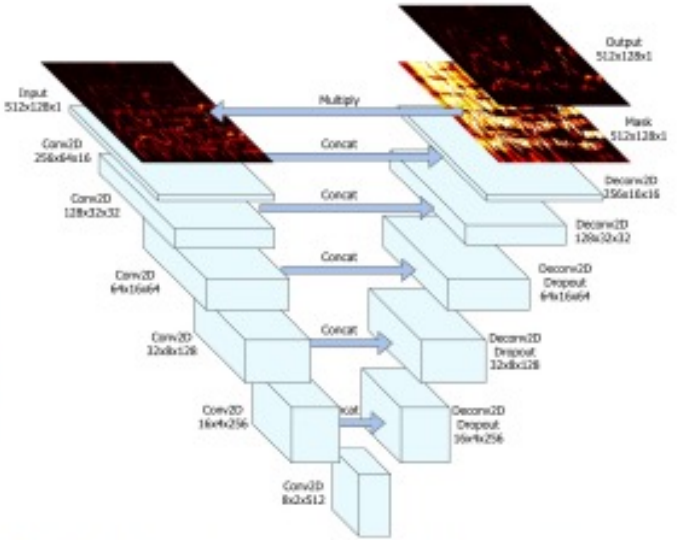
- 基本的なパイプライン
  - 音声表現としてスペクトログラムを使用する
  - 音源分離モデルは、2次元マスク($M \in [0.0, 1.0]$) を推定する
  - 混合スペクトログラムとマスクの要素ごとの乗算により、
    目標音源スペクトログラムを推定する
  - 混合位相をターゲット位相にコピーする



U-netを用いた歌声の分離

- U-Netによる正確なセグメンテーションが音源分離に有効
  - U-Netの入力は混合スペクトログラムで、出力はmask画像である
  - スペクトログラム上の、再構成された音のわずかなズレに音質は非常に敏感に反応する

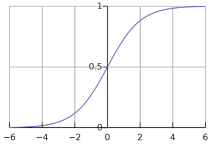




Singing Voice Separation with Deep U-Net Convolutional Networks, *Andreas Jansson, Eric Humphrey, Nicola Montecchio, Rachel Bittner, Aparna Kumar, Tillman Weyde, 2017*

U-netを用いた歌声の分離

- U-Netのアーキテクチャの詳細
  - 波形入力は8kHzにリサンプリングされ、スペクトログラムは[0,1]の範囲に正規化
  - カーネルサイズ(5,5)、ストライドサイズが (2,2) のフィルタ
  - プーリングなし
  - 出力はシグモイド関数(∈ [0.0, 1.0])

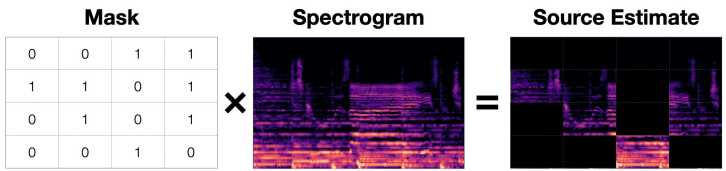


  - 損失関数として $L_{1,1}$ノルムを使用する。 $L(X, Y; \Theta) = \left|\left|f(X, \Theta) \odot X - Y\right|\right|_{1,1}$

音源分離したスペクトログラム(output)

mask(model output)

混合スペクトログラム(input)



メモ：ミックス音源からボーカル単体を引いて、
mask画像と要素積を取ったものを損失関数と置いている
・L_1,1ノルムは単純にその要素の絶対値の和

Singing Voice Separation with Deep U-Net Convolutional Networks, Andreas Jansson, Eric Humphrey, Nicola Montecchio, Rachel Bittner, Aparna Kumar, Tillman Weyde, 2017

U-netを用いた歌声の分離

メモ：
5年前の論文なので
今はもっと精度の良いものが
提案されているとは思います

- データセット
  - 2万組の音楽トラックとそのインストゥルメンタルバージョンを集める
  - 発声スペクトログラムは、T-F binで$\max(X - Y_i, 0)$により計算

Mixture spectrogram　　instrumental track spectrogram

- 結果
  - Baseline model：スキップ接続のない バージョン(それ以外は同じ)
  - Chimera model：ディープクラスタリングを用いたモデル

	U-Net	Baseline	Chimera
NSDR Vocal	**11.094**	8.549	8.749
NSDR Instrumental	**14.435**	10.906	11.626
SIR Vocal	**23.960**	20.402	21.301
SIR Instrumental	**21.832**	14.304	20.481
SAR Vocal	**17.715**	15.481	15.642
SAR Instrumental	**14.120**	12.002	11.539

iKala mean scores

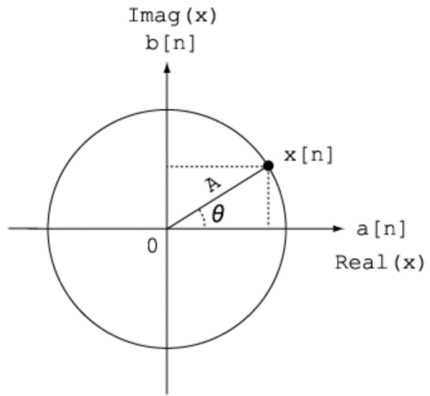
	U-Net	Baseline	Chimera
NSDR Vocal	**8.681**	7.877	6.793
NSDR Instrumental	**7.945**	6.370	5.477
SIR Vocal	**15.308**	14.336	12.382
SIR Instrumental	**21.975**	16.928	20.880
SAR Vocal	**11.301**	10.632	10.033
SAR Instrumental	**15.462**	15.332	12.530

MedleyDB mean scores

Singing Voice Separation with Deep U-Net Convolutional Networks, *Andreas Jansson, Eric Humphrey, Nicola Montecchio, Rachel Bittner, Aparna Kumar, Tillman Weyde, 2017*

Spectrogramの問題点

- STFTは、ウィンドウサイズ、ホップサイズなど、多くのパラメータに依存
  - 最適なパラメータ設定を見つけることが難しい
- ボーカル単体の出力にミックス音源の位相を使用して再構成すると、
  アーチファクト（ノイズというか何というか）が発生する場合がある
  - （特に異なる音源からの高調波成分が重なっている場合）
  - 振幅から位相を推定することもできる
    - Griffin-Limアルゴリズムは、最も一般的な位相復元手法の一つ
      (https://librosa.org/doc/latest/generated/librosa.griffinlim.html)
    - しかし、位相を完全に復元できるものではなく計算量も多い



Griffin-Limアルゴリズム

- 振幅スペクトログラムに対応する位相情報を乱数で作成

  ISTFTとSTFTを行い、残った情報（位相に矛盾があると情報が欠損する≒再構成できない）で元のスペクトログラムを置き換える。これを繰り返していく

(準備)位相スペクトログラムを乱数で選び初期値を生成

$$X(m,k) = A(m,k)\exp(j\phi(m,k))$$

(1)逆短時間フーリエ変換

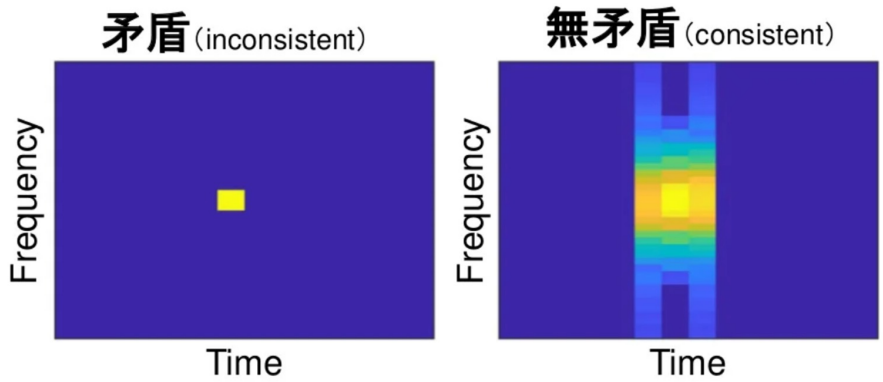
$$x(t) \leftarrow \mathrm{ISTFT}[X(m,k)]$$

(2) 短時間フーリエ変換を施す

$$X(m,k) \leftarrow \mathrm{STFT}[x(t)]$$

(3) 振幅スペクトログラムを置き換える

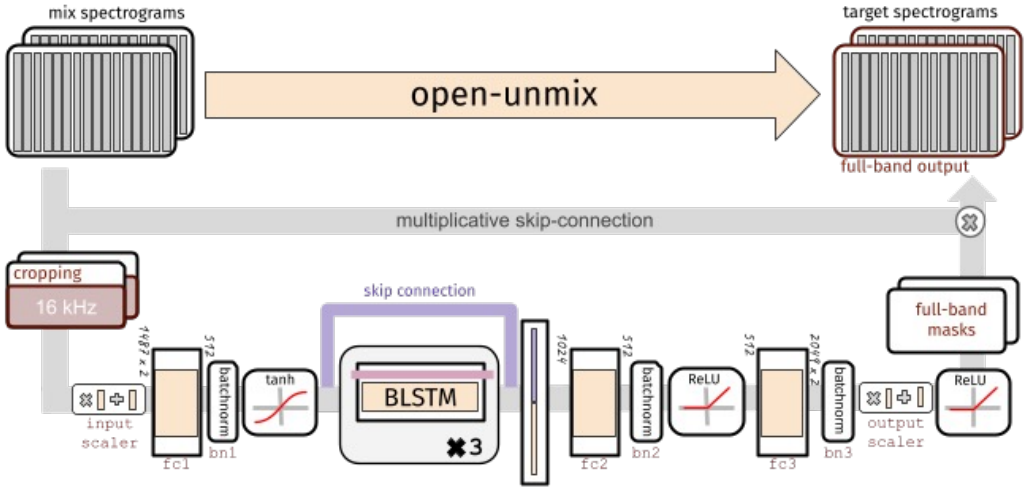
$$X(m,k) \leftarrow A(m,k)X(m,k)/|X(m,k)|$$



https://qiita.com/reiko_s/items/c53875e0cf688298218d
https://www.jstage.jst.go.jp/article/jasj/72/12/72_764/_pdf/-char/ja

ハンズオン

- Open-Unmix：音源分離のためのオープンソースDNN
  - ポップミュージックをボーカル、ドラム、ベース、その他の楽器の4つに分離
  - MUSDB18データセットで事前にトレーニング済
  - 3層の双方向ディープLSTMモデルを使用



Github

https://sigsep.github.io/

Google Colab

https://colab.research.google.com/drive/1pFa8GCJcOjF7xTRHS-sfZG6n4sjaPdle?usp=sharing

Open-Unmix - A Reference Implementation for Music Source Separation Fabian-Robert Stöter, Stefan Uhlich, Antoine Liutkus, and Yuki Mitsufuji,2019

参考文献

●Open Unmix

https://sigsep.github.io/

Google Colab

https://colab.research.google.com/drive/1mijF0zGWxN-KaxTnd0q6hayAlrID5fEQ