Generative Adversarial Networks

小泉 孝弥

1 Intoroduction

皆さんは, GAN というものを知っているだろうか. GAN[*1]とは敵対的生成ネットワーク (Generative Adversarial Networks) のことであり, 2014 年に登場した人工知能の技術である.「敵対的」という単語がついている通り, 2 つのニューラルネットワークという関数を競わせることで, 学習させていく手法である. 今回の方程では, まず, 機械学習についての説明をする. その後, GAN の数学的な定式化の説明をし, 最後に GAN の学習を安定させる方法について述べる.

2 Fundermetal concepts of Machine Learning

この節では機械学習の数学的な定式化を行う. $(\Omega, \mathcal{F}, \mathbb{P})$ を完備な確率空間とする. d, m を自然数とする.

Definition 2.1. (仮説空間, 仮説)
$\mathcal{X} := \mathbb{R}^d$ から $\mathcal{Y} := \mathbb{R}^m$ へのなんらかの条件を満たす写像の集まりのことを仮説空間といい $\mathcal{H}$ と表記する. すなわち,

$$\mathcal{H} = \{f : \mathcal{X} \to \mathcal{Y} \mid f \text{ が満たす条件 }\}$$

である. (条件の具体例は後に述べる). 仮設空間 $\mathcal{H}$ の元のことを仮説 (またはモデル) と呼ぶ. また, この時の $\mathcal{X}$ を入力空間, $\mathcal{Y}$ を出力空間と呼ぶ.

Definition 2.2. (データ)
$\mathcal{X}$ と $\mathcal{Y}$ の直積集合 $\mathcal{X} \times \mathcal{Y}$ の有限部分集合のことを教師ありデータという. また, $\mathcal{X}$ の有限部分集合のことを教師なしデータと呼ぶ.

Remark 2.3. (データについて)
データ D は確率分布 $\mathbb{P}_{\mathcal{X}\times\mathcal{Y}}$ にしてがって生成されているものとする. すなわち, $\mathbb{P}_{\mathcal{X}\times\mathcal{Y}}$ に従う確率変数列 $\{(X_n, Y_n)\}_{n=1}^N$ の観測値のことをデータと呼ぶ.

[*1] 「ぎゃん」と読む人と,「がん」と読む人がいるが, ここでは「ぎゃん」とすることにする.

Definition 2.4. (予測損失)
$\mathcal{H}$ を仮説空間とし, (X, Y) をデータの分布 $\mathbb{P}_{\mathcal{X}\times\mathcal{Y}}$ に従う確率変数とする. 以下で定義される $\ell : \mathcal{H} \to \mathbb{R}$ を予測損失と呼ぶ.

$$\ell(f) = \mathbb{E}[\|f(X) - Y\|].$$

予測損失が最小となるような $f \in \mathcal{H}$ を求めることが機械学習の目標である. しかし, 一般にデータの分布は未知であるためこの式を解くことができない. そこで、損失関数というものを定義し、それを最適化することを考える.

Definition 2.5. (損失関数)
$\mathcal{H}$ を仮説空間とする. $\mathcal{H}$ から $\mathbb{R}$ への写像 $\mathcal{L}_D : \mathcal{H} \to \mathbb{R}$ を損失関数 (Loss function) と呼ぶ.

Definition 2.6. (機械学習空間)
D をデータ, $\mathcal{H}$ を仮説空間, $\mathcal{L}_D$ を損失関数とする. この時 5 つ組 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を機械学習空間 (Machine Learning space) という.

Definition 2.7. (学習, 最適仮説)
$\mathcal{H}$ を仮設空間, $\mathcal{L}_D : \mathcal{H} \to \mathbb{R}$ を損失関数とする. 損失関数が最大または, 最小となるような $f^* \in \mathcal{H}$ を求めること[*2]を学習といい, $f^* \in \mathcal{H}$ を最適仮説と呼ぶ.

Definition 2.8. (機械学習)
機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ 上での学習を機械学習という. 特に, D が教師ありデータの時, 教師あり機械学習と呼び, 教師なしデータの時, 教師なし機械学習と呼ぶ.

Remark 2.9. (訓練データとテストデータ)
機械学習で学習を行う際は, データ D を全て使って学習させるのではなく, データを訓練データ D_{Train} とテストデータ D_{Test} に分ける. 訓練データはモデルの学習に用いて, テストデータはモデルの制度の確認のために用いる.

3 教師あり機械学習の具体例

前節では, 機械学習の抽象的な枠組みを紹介したが, この説では機械学習空間の具体例を述べる. $D = \{(x_i, y_i)\}_{i=1}^N \subset \mathcal{X} \times \mathcal{Y}$ を教師ありデータとする.

[*2] 機械学習においては厳密に損失関数が最大・最小となる関数が学習されるとは限らないが, そのような関数を求めることも学習の定義に含めるものとする.

3.1 単回帰分析・重回帰分析

最初に機械学習の最も基本的なモデルである単回帰分析を紹介する. その後, 多変数拡張である重回帰分析を紹介する.

Example 3.1. (単回帰分析)
機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を以下のように定義する.
$\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \mathbb{R}$,

$$\mathcal{H} = \{f : \mathcal{X} \to \mathcal{Y} \mid f(x) = wx, w \in \mathbb{R}\},$$
$$\mathcal{L}_D(f) = \sum_{i=1}^{N} |f(x_i) - y_i|^2.$$

この機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ 上で,

$$\arg\min_{f \in \mathcal{H}} \mathcal{L}_D(f)$$

を求める問題を単回帰分析という. 単回帰分析の最適仮説 $f^* \in \mathcal{H}$ は

$$f^*(w) = \frac{\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} x_i^2} w$$

となる. また, $\mathcal{H} \simeq \mathbb{R}$ である.

Example 3.2. (重回帰分析)
機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を以下のように定義する.
$\mathcal{X} = \mathbb{R}^d (N \geq d)$, $\mathcal{Y} = \mathbb{R}$,

$$\mathcal{H} = \{f : \mathcal{X} \to \mathcal{Y} \mid f(x) = W^\top x + b, W \in \mathbb{R}^d, b \in \mathbb{R}\},$$
$$\mathcal{L}_D(f) = \sum_{i=1}^{N} |f(x_i) - y_i|^2.$$

この機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ 上で

$$\arg\min_{f \in \mathcal{H}} \mathcal{L}_D(f)$$

を求める問題を重回帰分析という. $X \in \mathbb{R}^{N \times (d+1)}$ を

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ 1 & x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N1} & x_{N2} & \cdots & x_{Nd} \end{pmatrix}$$

と定義し, $\mathbf{y} = (y_1, y_2, \cdots, y_N)^\top \in \mathbb{R}^N$ とする. この時, $\mathbf{rank}(X) = d+1$ ならば, 最適仮説 $f^* \in \mathcal{H}$ は $[b\ W] = (X^\top X)^{-1} X^\top \mathbf{y}$ とすれば

$$f^*(x) = [b\ W]^\top x$$

である.

3.2 線形回帰

前小節で定義した重回帰分析は線形的なデータにしか適応できなかった. それを基底関数と呼ばれる関数列を定義する事で, 非線形データにも対応することができる.

Definition 3.3. (基底関数)
$\mathcal{X}$ を入力空間とし, 任意に $d \in \mathbb{N}$ をとる. $\mathcal{X}$ 上の連続関数列 $\{\phi_n\}_{n=1}^d \subset C(\mathcal{X}, \mathcal{X})$ が一次独立であるとき, $\Phi = (\phi_1, \phi_2, \cdots, \phi_d)^\top : \mathcal{X} \to \mathcal{X}^d$ を基底関数 (basis function) と呼ぶ.

Example 3.4. (多項式回帰)
機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を以下のように定義する.
$\mathcal{X} = \mathbb{R}, \mathcal{Y} = \mathbb{R}$,

$$\mathcal{H} = \{f : \mathcal{X} \to \mathcal{Y} \mid f(x) = W^\top \Phi(x), W \in \mathbb{R}^{d+1}\},$$
$$\mathcal{L}_D(f) = \sum_{i=1}^N |f(x_i) - y_i|^2.$$

ここで, $\phi_n(x) = x^n$, $n \in \{0, 1, \cdots, d\}$ とする.
この機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ 上で

$$\arg\min_{f \in \mathcal{H}} \mathcal{L}_D(f)$$

を求める問題を多項式回帰という. $X = [\phi_1(x_1), \phi_2(x_2), \cdots, \phi_N(x_N)]^T \in \mathbb{R}^{N \times d}$ とし, $\mathbf{y} = (y_1, y_2, \cdots, y_N)^T \in \mathbb{R}^N$ とする. この時, 最適仮説 f^* は重回帰分析と同様に $W = (X^\top X)^{-1} X^\top \mathbf{y}$ とすれば

$$f^*(x) = W^\top x$$

である.

Remark 3.5. (ハイパーパラメータと学習パラメータ)
多項式回帰の多項式の次数 $d \in$ や基底関数 $\{\phi_n\}_{n=1}^d$ のようにコンピュータに学習させるのではなく, 機械学習モデルの設計者が設定するパラメータのことをハイパーパラメータと呼ぶ. 一方, $W \in \mathbb{R}^{d+1}$ のように, データからコンピュータが自動で学習するパラメータのことを学習パラメータと呼ぶ.

3.3　過学習と正則化

前節の多項式回帰で学習を行うと以下のグラフのように, 訓練データに過剰適合してしまい, 未知のデータに関しての汎化性能が小さくなってしまうことを過学習 (overfitting) という.

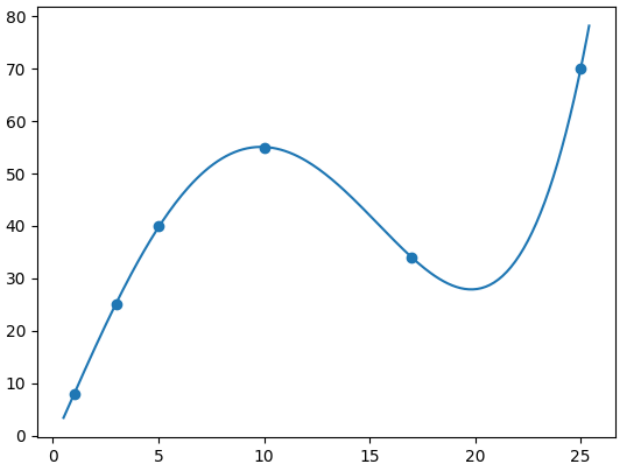

図 1　$d = 4$ の時の多項式回帰の過学習

過学習を防ぐための手法が正則化である.

Definition 3.6. (正則化)
$\mathcal{L}_D : \mathcal{H} \to \mathbb{R}$ を損失関数とする. $\mathcal{L}_D^R := \mathcal{L}_D + L^R$ とする. この時, $\mathcal{L}_D^R$ を $\mathcal{L}_D$ の正則化と呼び, L^R を正則化項と呼ぶ.

Example 3.7. (Ridge 正則化多項式回帰)
$\lambda \in \mathbb{R}^+$, $d \in \mathbb{N}$ を任意にとる. 機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を以下のように定義する.
$\mathcal{X} = \mathbb{R}, \mathcal{Y} = \mathbb{R},$

$$\mathcal{H} = \{f : \mathcal{X} \to \mathcal{Y} \mid f(x) = W^\top \Phi(x), W \in \mathbb{R}^{d+1}\},$$

$$\mathcal{L}_D(f) = \sum_{i=1}^{N} |f(x_i) - y_i|^2 + \lambda W^\top W. \qquad (W \text{ は } f \text{ のパラメータ})$$

ここで, $\phi_n(x) = x^n$, $n \in \{0, 1, \cdots, d\}$ とする.
この機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ 上で

$$\arg\min_{f \in \mathcal{H}} \mathcal{L}_D(f)$$

を求める問題を Ridge 正則化多項式回帰という. 通常の多項式回帰の時と同様に, $X = [\phi_1(x_1), \phi_2(x_2), \cdots, \phi_N(x_N)]^\top \in \mathbb{R}^{N \times d}$ とし, $\mathbf{y} = (y_1, y_2, \cdots, y_N)^\top \in \mathbb{R}^N$ とする. この時,

最適仮説 f^* は $W = (X^\top X + \lambda I)^{-1} X^\top \mathbf{y}$ とすれば

$$f^*(x) = W^\top x$$

である.

正則化の結果が以下のグラフである.

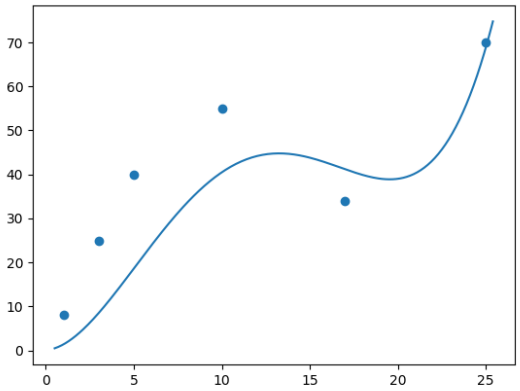

図 2　$\lambda = 700$, $d = 4$ の時の Ridge 正則化多項式回帰

このように正則化をすることで, パラメータが大きくなりすぎることを防ぐことで過学習を防ぐことができる.

Remark 3.8. (未学習について)
正則化を行う際にパラメータ $\lambda \in \mathbb{R}^{+}$[*3]$:= \{x \in \mathbb{R} \mid x \geq 0\}$ は自分で決める必要がある. その際に λ の値を大きくしすぎると未学習 (underfitting) という問題が発生する. 未学習とは訓練データに対してモデルが十分に適合できていないことをいう.

[*3] 正則化パラメータという.

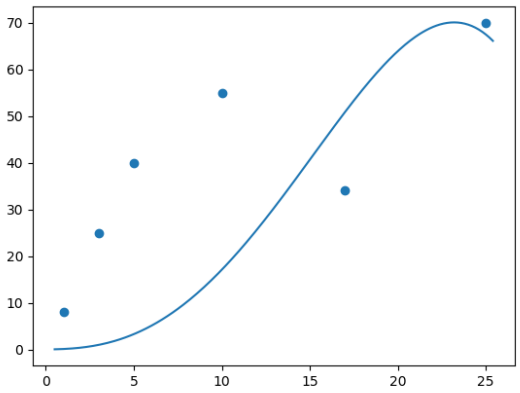

図 3　$\lambda = 30000$, $d = 4$ の時の Ridge 正則化多項式回帰の未学習

このように, 正則化パラメータは適切に選ぶことが大切である.

3.4　ロジスティック回帰と勾配降下法

最後の具体例として, ロジスティック回帰を紹介する. ロジスティック回帰は分類問題を解くための手法の 1 つである. まず, 実数値ベクトルを確率に変換するソフトマックス関数および, one-hot ベクトルというものを導入する.

Definition 3.9. (ソフトマックス関数)
$\psi : \mathbb{R}^d \to \mathbb{R}^d$ を以下で定義する.

$$\psi(x)_i = \frac{\exp(x_i)}{\sum_{k=1}^{d} \exp(x_i)}$$

この時, ψ をソフトマックス関数と呼ぶ.

Definition 3.10. (one-hot ベクトル)
$y \in \mathbb{R}^m$ が第 $c \in \{1, 2, \cdots, m\}$ 成分が 1 であり, 残りの成分が 0 である時, y を one-hot ベクトルという.

Example 3.11. (m 値分類ロジスティック回帰)
機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を以下のように定義する.
$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}^m$,

$$\mathcal{H} = \{f : \mathcal{X} \to \mathcal{Y} \mid f(x) = \psi(Wx + b), W \in \mathbb{R}^{m \times d}, b \in \mathbb{R}^m\},$$

$$\mathcal{L}_D(f) = -\sum_{n=1}^{N} \sum_{k=1}^{m} y_{nk} \log f(x_n)_k,$$

ここで $\psi : \mathbb{R}^m \to \mathbb{R}^m$ はソフトマックス関数であり, y_n は one-hot ベクトルである.
この機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ 上で

$$\arg\min_{f \in \mathcal{H}} \mathcal{L}_D(f)$$

を求める問題を m 値分類ロジスティック回帰という.

先ほどまでは, 損失関数の最小値を解析的に求めていたが, 今回の損失関数の最小値を解析的に求めることはせず, 勾配降下法という連続最適化アルゴリズムを用いて求めることにする. 勾配降下法とは C^∞ 関数 F の最小値を求めるためのアルゴリズムである.

Algorithm 1 Gradient Decent

Require: F: smooth function on $\mathbb{R}^d$
Require: $0 < \alpha < 1$: learning rate
Require: θ: Initial parameter vector
  while θ not converged **do**
    $\theta \leftarrow \theta - \alpha \nabla F(\theta)$
  end while
  return θ

勾配降下法の最適解への収束条件を述べておく.

Definition 3.12. (β-Lipschitz 連続)
(X, d_X), (Y, d_Y) を距離空間とする. $f : X \to Y$ が β-Lipschitz 連続であるとは,

$$\forall x, y \in X, d_Y(f(x), f(y)) \leq \beta d_X(x, y)$$

が成立することである.

Theorem 3.13. (勾配降下法の最適化収束条件 [6])
$\mathcal{H}$ をヒルベルト空間とし, $f : \mathcal{H} \to \mathbb{R}$ を C^∞ 関数とする. この時, ∇f が β-Lipschitz 連続ならば, 学習率 $\alpha \in \left(0, \frac{2}{\beta}\right)$ で最適解に弱収束する.

Example 3.14. (2 次関数における勾配降下法)
$y = x^2$ において勾配降下法を適応した結果を以下の図と表に示す. 本来の最適値である $x = 0$ に近づいているのが分かる.

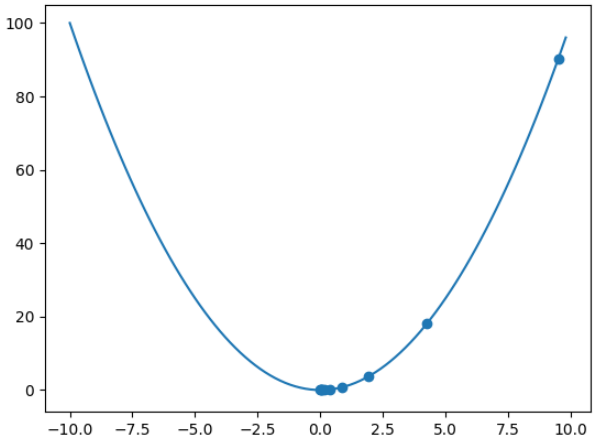

図 4　$\alpha = 0.001$ とした時の勾配降下法の結果.

イタレーション数	0	400	800	1200	1600	2000	2400	2800	3200	3600
θ の値	9.50	4.27	1.91	0.86	0.39	0.17	0.08	0.03	0.02	0.01

表 1　値の詳細

3.5　MNIST での実験

MNIST とは下記のような 0 から 9 までの自然数が書かれた 28×28 ピクセル画像のデータセットの名前である. 今回は, m 値分類のロジスティック回帰を用いて、画像に書かれている文字を予測する問題を考える. ロジスティック回帰の損失関数 $\mathcal{L}_D$ を勾配降下法を用いて最小化するため, まずは損失関数の勾配を計算する. ここで, $\mathcal{H} \simeq \mathbb{R}^{m \times d+1}$ だから, $\mathcal{L}_D$ を $\mathbb{R}^{m \times d+1}$ 上の関数と考える.

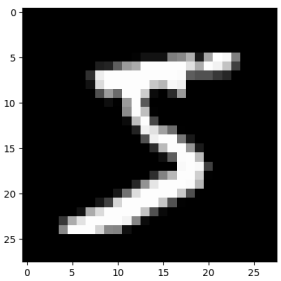

図 5　数字の 5

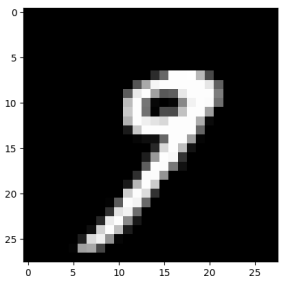

図 6　数字の 9

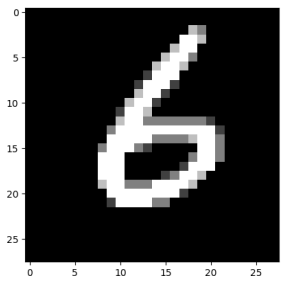

図 7　数字の 6

Lemma 3.15. ([1])
任意の $i \in \{1,2,\cdots,m\}, j \in \{1,2,\cdots,d\}$ に対して以下が成立する.

$$\frac{\partial \mathcal{L}_D}{\partial w_{ij}} = -\sum_{n=1}^{N} (y_{ni} - f(x_n)_i)\, x_{nj}$$
$$\frac{\partial \mathcal{L}_D}{\partial b_i} = -\sum_{n=1}^{N} (y_{ni} - f(x_n)_i)$$

この補題 3.15 より,

$$\begin{aligned}
\frac{\partial \mathcal{L}_D}{\partial W} &= -\begin{pmatrix} \sum_{n=1}^{N} (y_{n1} - f(x_n)_1)\, x_{n1} & \cdots & \sum_{n=1}^{N} (y_{n1} - f(x_n)_1)\, x_{nd} \\ \vdots & \ddots & \vdots \\ \sum_{n=1}^{N} (y_{nm} - f(x_n)_m)\, x_{n1} & \cdots & \sum_{n=1}^{N} (y_{nm} - f(x_n)_m)\, x_{nd} \end{pmatrix} \\
&= -\begin{pmatrix} y_{11} - f(x_1)_1 & \cdots & y_{N1} - f(x_N)_m \\ \vdots & \ddots & \vdots \\ y_{1m} - f(x_1)_m & \cdots & y_{Nm} - f(x_N)_m \end{pmatrix} \begin{pmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{Nd} \end{pmatrix} \\
&= -(Y - F)X^{\top},
\end{aligned}$$

ここで, $X = [x_1, x_2, \cdots, x_N]$, $Y = [y_1, y_2, \cdots, y_N]$, $F = [f(x_1), f(x_2), \cdots, f(x_N)]$ である. また, この計算より $\mathbf{1} = [1, 1, \cdots, 1]^{\top} \in \mathbb{R}^{N \times 1}$ とすれば,

$$\frac{\partial \mathcal{L}_D}{\partial b} = -(Y - F)\mathbf{1}$$

もいえる. これらを用いて学習した結果を以下に示す. また, この例からもわかるように機械学習においてパラメータの次元は非常に高次元なものとなる. (今回は $10 \times 784 + 10 = 7850$ 次元).

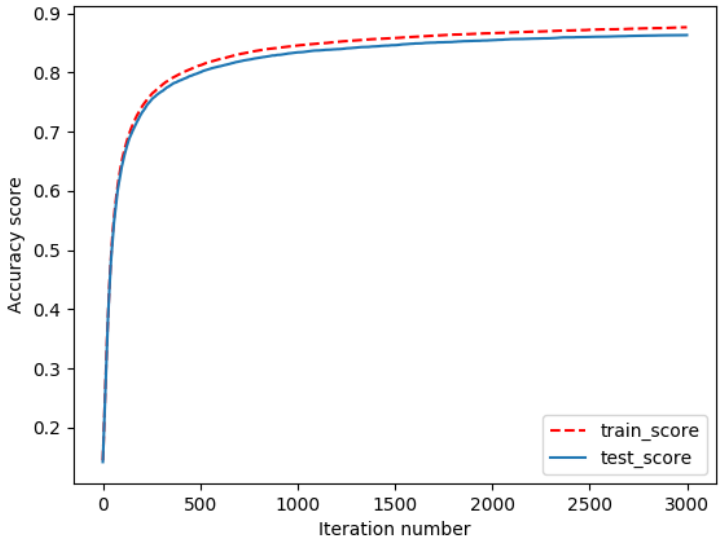

図 8　$\alpha = 0.0001$ とした時の, MNSIT 実験の結果

4 Deep Learning

ここから、高い表現能力を持つとされるニューラルネットワークについて説明する．ニューラルネットワークとはアフィン変換と活性化関数と呼ばれる関数の合成関数である．前節と同様に $D = \{(x_i, y_i)\}_{i=1}^N \subset \mathcal{X} \times \mathcal{Y}$ を教師ありデータとする．

4.1 Neural Network

Definition 4.1. (活性化関数)
$\sigma : \mathbb{R}^d \to \mathbb{R}^d$ を非線形かつ, Lipschitz 連続な関数とする．この時, σ を活性化関数 (activation function) という．

Example 4.2. (活性化関数の具体例)
活性化関数の具体例をいくつか述べる．

$$\sigma(x)_i = \frac{1}{1+\exp(-x_i)},$$
$$\mathrm{ReLU}(x)_i = \max(x_i, 0).$$

σ をシグモイド関数 (sigmoid function), ReLU を正規化線形関数 (Rectified Linear Unit) という．

Definition 4.3. (回帰型ニューラルネットワーク)
機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を以下のように定義する．$L = \{L_1, L_2, \cdots, L_K\}$ を自然数列とする．$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$,

$$\mathcal{H} = \left\{ f : \mathcal{X} \to \mathcal{Y} \;\middle|\; \begin{array}{l} f(x) = W^\top g_K \circ g_{K-1} \circ \cdots \circ g_1(x) - b, g_i = \eta(W_i x + b_i) \\ W_i \in \mathbb{R}^{L_i \times L_{i-1}}, b_i \in \mathbb{R}^{L_i}, W \in \mathbb{R}^{L_K}, b \in \mathbb{R} \end{array} \right\},$$
$$\mathcal{L}_D(f) = \frac{1}{N} \sum_{i=1}^N |f(x_i) - y_i|^2,$$

ここで η は活性化関数である．$f \in \mathcal{H}$ を回帰型 (全結合) ニューラルネットワーク (Full-connected Neural Network for Regression) と呼ぶ．

Definition 4.4. (分類型ニューラルネットワーク)
機械学習空間 $(\mathcal{X}, \mathcal{Y}, D, \mathcal{H}, \mathcal{L}_D)$ を以下のように定義する．$L = \{L_1, L_2, \cdots, L_K\}$ を自然数列とする．$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}^m$,

$$\mathcal{H} = \left\{ f : \mathcal{X} \to \mathcal{Y} \;\middle|\; \begin{array}{l} f(x) = \psi(W g_K \circ g_{K-1} \circ \cdots \circ g_1(x) - b), g_i = \eta(W_i x + b_i) \\ W_i \in \mathbb{R}^{L_i \times L_{i-1}}, b_i \in \mathbb{R}^{L_i}, W \in \mathbb{R}^{m \times L_K}, b \in \mathbb{R}^m \end{array} \right\},$$
$$\mathcal{L}_D(f) = -\sum_{n=1}^N \sum_{k=1}^m y_{nk} \log f(x_n)_k,$$

ここで η は活性化関数, y_i は one-hot ベクトル, ψ はソフトマックス関数である. $f \in \mathcal{H}$ を分類型 (全結合) ニューラルネットワーク (Full-connected Neural Network for Classification) と呼ぶ.

以下, ニューラルネットと書けば, 分類, 回帰ニューラルネットワークのいずれかを表すものとする.

Definition 4.5. (深層学習)
ニューラルネットワークが $K \geq 3$ の時に

$$\arg\min_{f \in \mathcal{H}} \mathcal{L}_D(f)$$

を求める問題を深層学習 (Deep Learning) と呼ぶ.

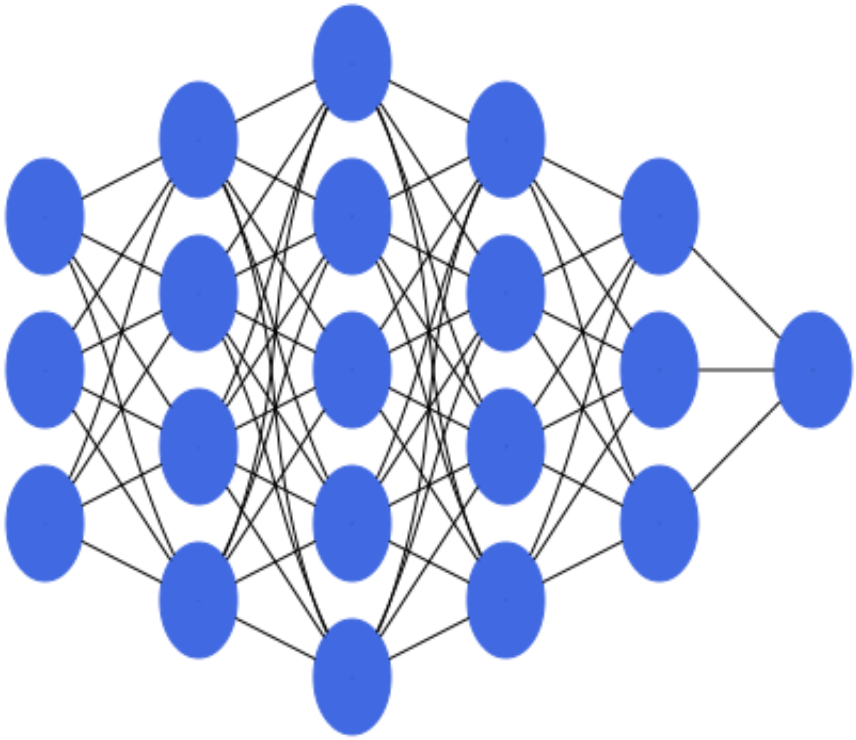

図 9　ニューラルネットワークの図

Remark 4.6. (フレームワーク)
現在はニューラルネットワークを簡単に実装するための深層学習フレームワークというものがいくつか存在する. 代表なものとして, Facebook の Pytorch や, Google の Tensorflow などがある.

4.2 The Universal Theorem of Neural Network

冒頭でも述べたが, 深層学習は前説まで紹介してきた機械学習の手法と比べて, 精度がよくなる. しかしながら, その数学的な理由については現在も研究途中である. 一方で, ニューラルネットの表現能力についてはいくつかの成果が出ており, その一つである普遍性定理を紹介する. 証明については [2] や [7] を見て欲しい.

Definition 4.7. (符号付き測度)
$(X, \mathcal{F})$ を可測空間とする. $\mu : \mathcal{F} \to \mathbb{R}$ が, 任意の互いに素な $\{A_n\}_{n\in\mathbb{N}} \subset \mathcal{F}$ に対して

$$\mu\left(\bigcup_{n\in\mathbb{N}} A_n\right) = \sum_{n\in\mathbb{N}} \mu(A_n)$$

を満たすとき, μ を符号付き測度という.

Definition 4.8. (正則 Borel 測度)
$(X, \mathcal{O})$ を位相空間とする. Borel 集合族 $\mathcal{B}(X)$ 上の Borel 測度 μ が

$$\begin{aligned}\forall A \in \mathcal{B}(X), \mu(A) &= \inf\{\mu(U) \mid U \supset A, U \in \mathcal{O}\} \\ &= \sup\{\mu(K) \mid K \subset A, K \text{ is compact and closed}\}\end{aligned}$$

を満たす時, 正則 Borel 測度という.

Definition 4.9. (正則符号付き Borel 測度)
$(X, \mathcal{O})$ を位相空間とする. $(X, \mathcal{B}(X))$ 上の符号付き測度 μ が正則符号付き Borel 測度であるとは, 以下で定義される $\mathcal{B}(X)$ 上の有限測度 μ^+, μ^- が共に正則 Borel 測度になることである.

$$\begin{aligned}\mu^+(A) &= \sup\{\mu(B) \mid B \in \mathcal{B}(X), B \subset A\}, \\ \mu^-(A) &= -\inf\{\mu(B) \mid B \in \mathcal{B}(X), B \subset A\}.\end{aligned}$$

また, X 上の正則符号付き測度全体の集合を $M(X)$ と表記する.

Definition 4.10. (Sigmoidal)
関数 $\sigma : \mathbb{R} \to \mathbb{R}$ がシグモイド的関数であるとは

$$\lim_{t\to+\infty} \sigma(t) = 1 \quad \lim_{t\to-\infty} \sigma(t) = 0$$

を満たすことである.

Definition 4.11. (discriminatory)
X を $\mathbb{R}^n$ の部分空間とする. 関数 $\sigma : \mathbb{R} \to \mathbb{R}$ が discriminatory 関数であるとは

$$\forall \mu \in M(X), \left(\forall y \in \mathbb{R}^n, \theta \in \mathbb{R}, \int_X \sigma(\langle x, y\rangle + \theta) d\mu(x)\right) \Longrightarrow \mu = 0$$

が成立することである.

Theorem 4.12. (普遍性定理)
$X \subset \mathbb{R}^n$ をコンパクト連結部分空間とし, $\mathcal{H}$ を $K = 1$ の時の回帰型ニューラルネットの仮説空間とする. この時, 活性化関数 η がシグモイド的関数ならば,

$$\forall f \in C(X), \forall \varepsilon > 0, \exists G \in \mathcal{H} \text{ s.t. } \sup_{x \in X} |f(x) - G(x)| < \varepsilon$$

が成立する.

5 Generative Adversarial Networks

GAN[3] は 2014 年に Goodfellow らによって提案された手法である. その後, WGAN, DCGAN といった様々な亜種が登場し, 現在も様々な手法が提案されている. 冒頭でも述べたが, GAN は 2 つのニューラルネットワークを戦わせる手法である. これは, よく, 警察と泥棒の偽札造りに例えられる. G を泥棒, D を警察とした時, 以下の作業を繰り返すことで, 偽札の本物具合をあげていくといったものである.

1. 泥棒 G が偽札を作る.
2. 警察 D が偽札であると見破る.
3. 泥棒 G はなぜ, 警察 D に偽札であるとバレたのかを考え, より本物に近い偽札を作れるようになる.

これらは以下のように min-max ゲームとして定式化される.

5.1 GAN の定式化

$(\Omega, \mathcal{F}, \mathbb{P})$ を完備な確率空間とする. $\mathcal{X}$ を d 次元線型空間とする.

$$\mathcal{H}_1 = \{G : \mathcal{Z} \to \mathcal{X} \mid G \text{ はニューラルネット } \},$$
$$\mathcal{H}_2 = \{D : \mathcal{X} \to [0,1] \mid D \text{ はニューラルネット } \}$$

ここで、$\mathcal{Z}$ は潜在空間と呼ばれる $\mathbb{R}^d$ の線型部分空間である. また, 確率変数 $Z : \Omega \to \mathcal{Z}$ に対し, $g(Z)$ が従う確率分布を $\mathbb{P}$

6 Applications of GANs

参考文献

[1] darden, 多クラス分類ロジスティック回帰,
http://darden.hatenablog.com/entry/2018/01/25/222201, Python と機械学習, 2018

[2] G. Cybenko, Approximation by Superpositions of a Sigmoidal Function, Mathmatics of control, signal and systems, vol. 2, no. 4, 1989

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville and Yoshua Bengio. Generative adversarial nets. In Advances in Neural Information Processing Systems, 2014.

[4] Martin Arjovsky and Leon Bottou, Towards Principled Methods for Training Generative Adversarial Networks, International Conference on Learning Representations, 2017.

[5] Takeru Miyato, Toshiki Kataoka, Masanori Koyama and Yuichi Yoshida, Spectral Normalization for Generative Adversarial Networks, International Conference on Learning Representations, 2018.

[6] 矢田部 浩平, 信号処理と最適化, http://www.sp.ipc.i.u-tokyo.ac.jp/~saruwatari/SP-Grad2018_09ProfYatabe.pdf, 早稲田大学 表現工学科, 2018.

[7] mochimochidog, ニューラルネットワークの普遍性定理, https://qiita.com/mochimochidog/items/ca04bf3df7071041561a, Qiita, 2020.

[8] Preferred Networks, ディープラーニング入門 Chainer チュートリアル, https://tutorials.chainer.org/ja/index.html, 2019