目次

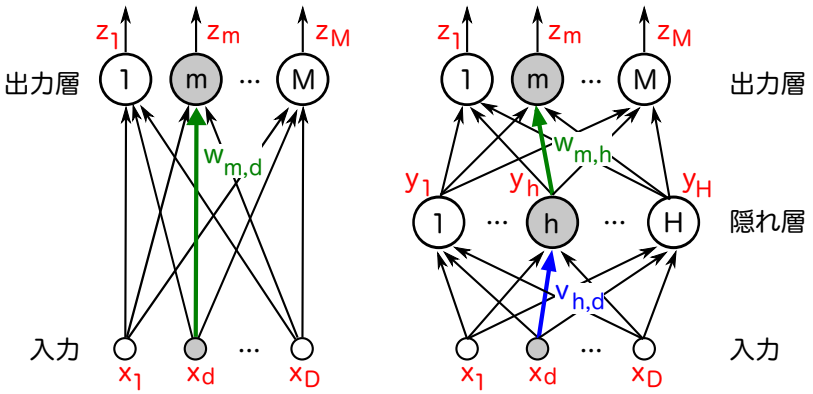
★14　パターン認識と機械学習 (5) — ニューラルネットワーク（承前）

前回説明したニューロンモデルを階層的につなぎあわせた構造のニューラルネットワークは，**多層パーセプトロン** (**MLP**) (☆ 1) と呼ばれる．

☆ 1) 多層パーセプトロン: Multi-Layer Perceptron. 階層型ニューラルネットと呼ぶこともある．

★14.2　多層パーセプトロンとその学習

下図に，2 種類のニューラルネットを示す．左右の図のいずれも，大きめの丸一つが一つのニューロンを表している．左のものは入力が直接出力とつながった構造をしている．一方，右の方は，入力と出力の間に「隠れた」ニューロンの層（これを「隠れ層」または「中間層」という）を有している．一般に，「多層」パーセプトロンという時は，このような隠れ層を 1 層以上有するものを指す．



図右の MLP の入出力は，次のような式で表される．

$$y_h = \sigma\left(v_{h,0} + \sum_{d=1}^{D} v_{h,d} x_d\right) \qquad (h = 1, 2, \ldots, H) \tag{1}$$

$$z_m = \sigma\left(w_{m,0} + \sum_{h=1}^{H} w_{m,h} y_h\right) \qquad (m = 1, 2, \ldots, M) \tag{2}$$

ここで，y_h は隠れ層の h 番目のニューロンの値であり，$v_{h,d}$ はこのニューロンと入力の d 番目の要素との結合の強さを表すパラメータである．同様に，z_m は出力層の m 番目のニューロンの値であり，$w_{m,h}$ はこのニューロンと隠れ層の h 番目のニューロンとの結合の強さを表すパラメータである．関数 $\sigma(s)$ は**活性化関数** (☆ 2) と呼ばれるものであり，前回資料式 (2) のシグモイド関数や，恒等関数 $\sigma(s) = s$ (☆ 3) などが用いられる．

MLP の特徴は，上述のように隠れ層を有することである．隠れ層ニューロンの活性化関数にシグモイドのように非線形なものを用いると，MLP 全体の入出力も非線形関数となる．そのため，入力と出力の間に複雑な関係があるようなデータの場合でも，うまく学習できると期待される．

☆ 2) 活性化関数: activation function.

☆ 3) シグモイドと違って出力が $(0, 1)$ に限定されないので，任意の実数を出力させたい場合に出力層の活性化関数として用いたりする．

MLP においては，ニューロン間の結合の強さを表す値が，学習すべきパラメータとなる．式 (1) と (2) で表される MLP の場合，入力—隠れ層のニューロン間の結合を表す値 $v_{h,d}$ と，隠れ層ー出力層のニューロン間の結合を表す値 $w_{m,h}$ がパラメータである．それらの学習には，ロジスティック回帰の場合と同様に，勾配法／最急降下法を用いることが多い．目的関数としては，出力の正解と実際の出力との間の二乗誤差（最小二乗法で説明した誤差関数と同じ形のもの，次節も参照）や交差エントロピー（前回資料参照）などを用いることができる．いずれの場合も，上記パラメータに関する微分が計算できるので，パラメータを適当な初期値から逐次修正していく形の学習アルゴリズムを構成することができる．

★ 14.3 多層パーセプトロンの応用例: 非線形回帰

多層パーセプトロンは，教師データと目的関数の与え方次第で，回帰／識別どちらの問題にも適用することができる．回帰の場合，通常は二乗誤差を目的関数とする．一方，識別の場合，交差エントロピーを用いることが多い．

最も単純な形の MLP として，上の図および式 (1) と (2) で $D = M = 1$ とした場合，つまり入力も出力も 1 つの場合を考える．出力層ニューロンの活性化関数は恒等関数（$\sigma(s) = s$）とする．このとき，入力 x に対するこの MLP の出力 z は，次式のようになる．

$$y_h = s(v_{h,0} + v_{h,1}x) \qquad (h = 1, 2, \ldots, H) \tag{3}$$

$$z = \sum_{h=1}^{H} w_h y_h \tag{4}$$

この MLP に対して, 入力と出力の正解のペア N 個から成る学習データ $\{(x_n, \tilde{z}_n)|n = 1, 2, \ldots, N\}$ を与え，x_n に対する MLP 出力 z_n と正解 $\tilde{z}_n$ との間の二乗誤差の和

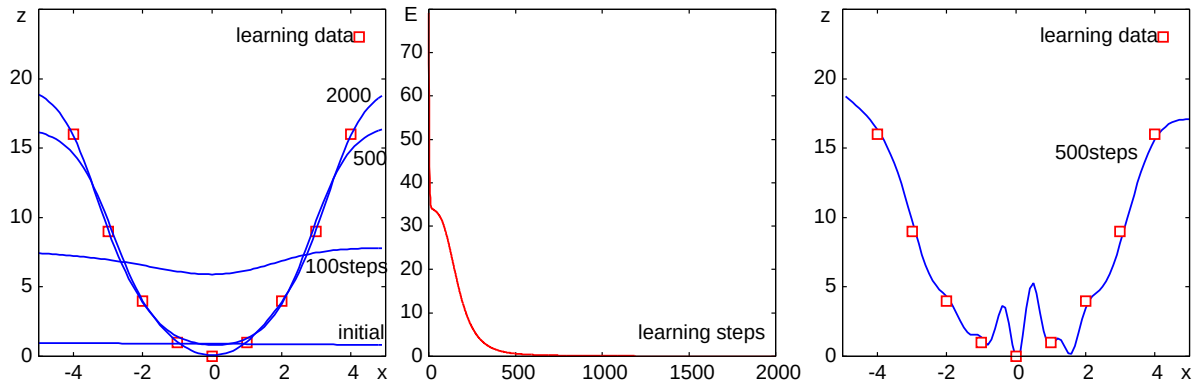
$$E = \frac{1}{2}\sum_{n=1}^{N}(\tilde{z}_n - z_n)^2 \tag{5}$$

を目的関数として学習させる．これは，入出力がともに 1 変数の場合の回帰問題への MLP の適用例となっている．MLP を用いると，非線形な活性化関数をもった隠れ層のおかげで複雑な曲線を当てはめることができる．

以下に，実際に学習を行なった実験の結果を示す．二次関数 $z = x^2$ を近似することを目標に，学習データを $(-4, 16), (-3, 9), \ldots, (4, 16)$ の $N = 9$ 個とした．また，中間層のニューロン数は $H = 10$ と $H = 100$ の二通りとした．

下図左は，$H = 10$ の場合の学習過程における MLP 出力の変化を示している．学習が進むにつれてうまく学習データを近似できるようになっていることがわかる．このことは，下図中に示した二乗誤差 E の変化の様子からもわかる．

一方，下図右は，$H = 100$ とした場合の学習結果の一例を示している．この場合，学習データに対する二乗誤差 E はほぼ 0 になっており，MLP 出力の曲線は全ての学習データ点を通っているが，その形は近似対象である放物線とはほど遠いものとなっている．このような現象は「過学習」と呼ばれている（次節参照）．



★ 14.4 ニューラルネットの課題と現在

上記実験結果の $H = 100$ の場合のように，機械学習の仕組みが学習データに過剰に適合し，未知のデータに対して**汎化**できなくなってしまう現象を，**過学習** (☆ 4) という．ニューラルネットに限らず，広く一般の機械学習で問題になる現象である．過学習は，学習モデルの自由度が高すぎる場合，すなわち，学習データの数に比べてパラメータの数が多すぎる場合に起こりやすい．最小二乗法による多項式のあてはめの例でも，次数を大きくした場合にやはり過学習が起こっていた（cf. ★ 12.2.2 多項式のあてはめ）．

過学習を回避するには，学習モデルの自由度を適切に設定すればよいのであるが，どの程度が適切かを前もって知るのは難しい．自由度が低すぎると学習データにすらうまく合わせられないし，自由度が高過ぎると過学習を起こしてしまう．機械学習の分野では，過学習を避けるための学習の理論とアルゴリズムの研究が長年続けられてきている．

近年になって機械学習が様々な分野で活用されるようになってきたのは，それらの研究の進歩と，コンピュータの性能向上のおかげである．特に最近では，隠れ層をいくつも有する多層のニューラルネットが注目されている．層の数を多くするとパラメータの数が非常に多くなるため，従来は過学習や計算速度の問題でまともに学習できなかったものが，学習アルゴリズムとコンピュータの進歩で改良され，画像認識や音声認識などで高い性能を発揮することがわかってきたからである．このようなニューラルネットの学習は**ディープラーニング（深層学習）** (☆ 5) と呼ばれている．

☆ 4) 過学習: overfitting, overtraining.

☆ 5) ディープラーニング: deep learning.