2 順伝播型ネットワーク

ニューラルネットワークを定式化

2.1 ユニットの出力

キーワード

順伝播型 (ニューラル) ネットワーク，多層パーセプトロン，重み，バイアス，活性化関数

ネットワークを構成する各ユニットは，入力 $x_1, ..., x_I$ を受けとり，総入力:

$$u = w_1 x_1 + \cdots + w_I x_I + b \tag{1}$$

を計算した上で，活性化関数 f を用いて

$$z = f(u) \tag{2}$$

を出力します．ここで $w_1, ..., w_I$ は**重み**，b は**バイアス**です．

1 つの層に複数のユニットが存在している場合，ベクトルを用いて次のように書けます．

$$\mathbf{u} = \mathbf{W}\mathbf{x} + \mathbf{b} \tag{3}$$
$$\mathbf{z} = \mathbf{f}(\mathbf{u}) \tag{4}$$

ただし，入力 (=第一層) のユニット数は I，第二層のユニット数は J とし，各記号は次のとおり：

$$\mathbf{u} = (u_1, ..., u_J)^\top, \mathbf{x} = (x_1, ..., x_I)^\top, \mathbf{b} = (b_1, ..., b_J)^\top, \mathbf{z} = (z_1, ..., z_J)^\top, \tag{5}$$

$$\mathbf{W} = \begin{pmatrix} w_{11} & \cdots & w_{1I} \\ \vdots & \ddots & \vdots \\ w_{J1} & \cdots & w_{JI} \end{pmatrix}, \mathbf{f}(\mathbf{u}) = (f(u_1), ..., f(u_J))^\top \tag{6}$$

2.2 活性化関数 (ユニット間，出力層ではない)

キーワード

ロジスティック (シグモイド) 関数，シグモイド関数，正規化線形関数，マックスアウト

- ロジスティック関数 $\subset$ シグモイド関数

$$f(u) = \frac{1}{1 + e^{-u}} \tag{7}$$

- 双曲線正接関数 (hyperbolic tangent) $\subset$ シグモイド関数

$$f(u) = \tanh(u) \tag{8}$$

- 正規化線形関数

$$f(u) = \max(u, 0) \tag{9}$$

この関数をもつユニットを **ReLU** と言います．

- 線形写像・恒等写像

- ロジスティック関数を区分線形近似したもの

$$f(u) = \begin{cases} -1 & u < -1 \\ u & -1 \leq u < 1 \\ 1 & u \geq 1 \end{cases} \tag{10}$$

- マックスアウト (K 個のユニットをまとめたようなもの)

$$u_{jk} = \sum_i w_{jik} z_i + b_{jk} \tag{11}$$

$$f(u_j) = \max_{k+1,\ldots,K} u_{jk} \tag{12}$$

2.3 多層ネットワーク

キーワード

入力層，中間層 (=隠れ層)，出力層

各層のユニットの入出力の計算を区別するために，層が $l = 1, 2, ..., L$ となっているときに各変数の肩に (l) と記すことにします．すると，層 2 のユニットの出力 $\mathbf{z}^{(2)}$ は，入力層 $\mathbf{x}$ から

$$\mathbf{u}^{(2)} = \mathbf{W}^{(2)}\mathbf{x} + \mathbf{b}^{(2)} \tag{13}$$

$$\mathbf{z}^{(2)} = \mathbf{f}(\mathbf{u}^{(2)}) \tag{14}$$

と計算され，その後の層 $l + 1$ のユニットの出力 $\mathbf{z}^{(l+1)}$ は，1 つ下の層の出力 $\mathbf{z}^{(l)}$ から

$$\mathbf{u}^{(l+1)} = \mathbf{W}^{(l+1)}\mathbf{z}^{(l)} + \mathbf{b}^{(l+1)} \tag{15}$$

$$\mathbf{z}^{(l+1)} = \mathbf{f}(\mathbf{u}^{(l+1)}) \tag{16}$$

と計算される．最後に層 L まで計算が終わったら，ネットワークの最終的な出力を

$$\mathbf{y} = \mathbf{z}^{(L)} \tag{17}$$

と表記する．各層の重み $\mathbf{W}^{(l)}$ とバイアス $\mathbf{b}^{(l)}$ をユニットのパラメータと呼び，出力 $\mathbf{y}$ は入力とパラメータによって決まることから，

$$\mathbf{y}\left(\mathbf{x}; \mathbf{W}^{(2)}, ..., \mathbf{W}^{(L)}, \mathbf{b}^{(2)}, ..., \mathbf{b}^{(L)}\right) \tag{18}$$

あるいは簡単に

$$\mathbf{y}(\mathbf{x}; \mathbf{w}) \tag{19}$$

と書きます．

2.4 出力層の設計と誤算関数

キーワード

訓練サンプル，訓練データ，誤算関数，回帰，二値分類，多クラス分類，最尤推定，ソフトマックス関数，交差エントロピー

学習に使用する入出力データ

$$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{d}_1), (\mathbf{x}_2, \mathbf{d}_2), ..., (\mathbf{x}_N, \mathbf{d}_N)\} \tag{20}$$

を訓練データといいます．$\mathbf{x}_n$ が入力，$\mathbf{d}_n$ が出力．

2.4.1 回帰

出力層の活性化関数には恒等写像を用い，誤算関数には二乗誤算の和

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} ||\mathbf{d}_n - \mathbf{y}(\mathbf{x}_n; \mathbf{w})||^2 \tag{21}$$

を使用し，$E(\mathbf{w})$ がもっとも小さくなる $\mathbf{w}$ を求めます．

2.4.2 二値分類

ここでは $\mathbf{x}$ を指定したときに $d = 1$ となる事後確率 $p(d = 1|\mathbf{x})$ を，ネットワーク全体の出力 $y(\mathbf{x}; \mathbf{w})$ でモデル化し，出力層の活性化関数にロジスティック関数を使うことで，

$$p(d = 1|\mathbf{x}) \approx y(\mathbf{x}; \mathbf{w}) \tag{22}$$

とします．パラメータ $\mathbf{w}$ は最尤推定で求めることにします．$p(d = 1|\mathbf{x}) = y(\mathbf{x}; \mathbf{w})$ より $p(d = 0|\mathbf{x}) = 1 - y(\mathbf{x}; \mathbf{w})$ であることと，

$$p(d|\mathbf{x}) = p(d = 1|\mathbf{x})^d p(d = 0|\mathbf{x})^{1-d} \tag{23}$$

より，N 個の入力に対する尤度は

$$L(\mathbf{w}) = \prod_{n=1}^{N} p(d_n|\mathbf{x}_n; \mathbf{w}) = \prod_{n=1}^{N} y(\mathbf{x}_n; \mathbf{w})^{d_n} \{1 - y(\mathbf{x}_n; \mathbf{w})\}^{1-d_n} \tag{24}$$

となります．誤差関数としてはこれに対数をとって符号を反転させた

$$E(\mathbf{w}) = -\sum_{n=1}^{N} [d_n \log y(\mathbf{x}_n; \mathbf{w}) + (1 - d_n) \log \{1 - y(\mathbf{x}_n; \mathbf{w})\}] \tag{25}$$

を用て，パラメータ $\mathbf{w}$ を求めます．

2.4.3 多クラス分類

出力層の活性化関数

ネトワークの出力層にクラス数 K と同じ K 個のユニットを並べ，出力層の各ユニット $k(= 1, ..., K)$ の出力を

$$y_k = z_k^{(L)} = \frac{\exp(u_k^{(L)})}{\sum_{j=1}^{K} \exp(u_j^{(L)})} \tag{26}$$

とします．この関数はソフトマックス関数と呼ばれます．この式のスコア $u_k^{(L)}$ に対して

$$u_k^{(L)} = \mathbf{W}^{(L)}\mathbf{z}^{(L)} + \mathbf{b}^{(L)} \tag{27}$$

と出力層のユニットを当てはめ，$\mathbf{W}$ を求めます．

誤差関数の導出

分類する K 個のクラスを $C_1, ..., C_K$ を表すとき，ユニット k の出力値 y_k を用いて，クラス C_k に属する確率を

$$p(C_k|\mathbf{x}) = y_k = z_k^{(L)} \tag{28}$$

でモデル化します．n 個めのサンプルがクラス C_k に入っているとき，出力値であるラベル $\mathbf{d}_n$ が k 番目の要素が 1 でそれ以外は 0 であるベクトルで表されているとします．たとえば，合計 10 クラスで 3 番目のクラスに属しているとき，

$$\mathbf{d}_n = [0\,0\,1\,0\,0\,0\,0\,0\,0\,0]^\top \tag{29}$$

と表されます．このとき $\mathbf{d}_n$ の事後分布は

$$p(\mathbf{d}_n|\mathbf{x}) = \prod_{k=1}^{K} p(C_k|\mathbf{x})^{d_k} \tag{30}$$

となります．これをもとにした尤度関数 $L(\mathbf{w})$ は

$$L(\mathbf{w}) = \prod_{n=1}^{N} p(\mathbf{d}_n|\mathbf{x}_n;\mathbf{w}) = \prod_{n=1}^{N}\prod_{k+1}^{K} p(C_n|\mathbf{x}_n)^{d_{n,k}} = \prod_{n=1}^{N}\prod_{k+1}^{K}(y_k(\mathbf{x};\mathbf{w}))^{d_{n,k}} \tag{31}$$

となるので，対数と負をとったものを誤差関数とします：

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k+1}^{K} d_{n,k}\log y_k(\mathbf{x};\mathbf{w}) \tag{32}$$

この誤差関数は交差エントロピーと呼ばれます．

2.4.4 出力層の活性化関数

二値分類でロジスティック関数，多クラス分類でソフトマックス関数を使うことの妥当性を確認する計算

二値分類

事後確率 $p(d=1|\mathbf{x})$ は条件付確率の定義から

$$p(d=1|\mathbf{x}) = \frac{p(\mathbf{x},d=1)}{p(\mathbf{x})} = \frac{p(\mathbf{x},d=1)}{p(\mathbf{x},d=0)+p(\mathbf{x},d=1)} \tag{33}$$

と計算できます．ここでスコア u を

$$u := \log\frac{p(\mathbf{x},d=1)}{p(\mathbf{x},d=0)} \tag{34}$$

で表せば，先ほどの事後確率は

$$p(d=1|\mathbf{x}) = \frac{p(\mathbf{x},d=1)}{p(\mathbf{x},d=0)+p(\mathbf{x},d=1)} \tag{35}$$

$$= \frac{\mathrm{e}^{\log p(\mathbf{x},d=1)}}{\mathrm{e}^{\log p(\mathbf{x},d=0)}+\mathrm{e}^{\log p(\mathbf{x},d=1)}} \tag{36}$$

$$= \frac{1}{1+\mathrm{e}^{\log p(\mathbf{x},d=0)-\log p(\mathbf{x},d=1)}} \tag{37}$$

$$= \frac{1}{1+\mathrm{e}^{-u}} \tag{38}$$

と計算でき，ロジスティック関数に一致することを確認できます．

多クラス分類

ロジスティック関数と同様クラス C_k の事後確率は条件付確率の定義から

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}, C_k)}{p(\mathbf{x})} = \frac{p(\mathbf{x}, C_k)}{\sum_{j=1}^{K} p(\mathbf{x}, C_j)} \tag{39}$$

となります．ここではクラス k のスコアを $u_k := \log p(\mathbf{x}, C_k)$ と置くことで，この事後確率を

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}, C_k)}{\sum_{j=1}^{K} p(\mathbf{x}, C_j)} \tag{40}$$

$$= \frac{\mathrm{e}^{\log p(\mathbf{x}, C_k)}}{\sum_{j=1}^{K} \mathrm{e}^{\log p(\mathbf{x}, C_j)}} \tag{41}$$

$$= \frac{\mathrm{e}^{u_k}}{\sum_{j=1}^{K} \mathrm{e}^{u_j}} \tag{42}$$

と計算でき，ソフトマックス関数に一致することを確認できます．

3 確率的勾配降下法と誤差逆伝播法

3.1 勾配降下法

キーワード

大域解，局所解，勾配効果法，勾配，学習係数

やることまとめ

順伝播型ネットワークの学習のまとめ：与えられた訓練データ

$$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{d}_1), (\mathbf{x}_2, \mathbf{d}_2), ..., (\mathbf{x}_N, \mathbf{d}_N)\} \tag{43}$$

を元に計算される誤差関数 $E(\mathbf{w})$ をネットワークのパラメータ $\mathbf{w}$ について最小化すること．学習のゴールは $\arg\min_{\mathbf{w}} E(\mathbf{w})$(=選んだ $E(\mathbf{w})$ に対し最小値を与える $\mathbf{w}$) を求めることです．ただし，$E(\mathbf{w})$ は一般に凸関数ではないため，大域的最小解を直接求めるのは不可能です．代わりに，$E(\mathbf{w})$ の局所的な最小解 $\mathbf{w}$ を求めます．この $E(\mathbf{w})$ の局所的最小解 $\mathbf{w}$ を求める方法の代表例が勾配降下法です．

勾配降下法とは

勾配降下法とは，現在の $\mathbf{w}$ を，負の勾配方向 $-\nabla E$ に少し動かすことを繰り返す方法です[*1]．ここで勾配とは微分を並べたベクトル：

$$\nabla E = \frac{\partial E}{\partial \mathbf{w}} = \left[\frac{\partial E}{\partial w_1} \cdot \frac{\partial E}{\partial w_M}\right]^{\top} \tag{44}$$

です．現在の重みを $\mathbf{w}^{(t)}$，動かした後の重みを $\mathbf{w}^{(t+1)}$ と表せば，

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t+1)} - \epsilon \nabla E \tag{45}$$

[*1] ∇E は，変数である $\mathbf{w}$ を動かしたとき E は ∇E だけ増加するということを表す量のことです．これを負の方向に動かせば E は ∇E だけ減少します．

で重みを更新することが勾配降下法であると書き直すことができます．ϵ は $\mathbf{w}$ の更新量の大きさを決めるパラメータ (**学習係数**) です[*2]．実際の更新手順は，初期値 $\mathbf{w}^{(1)}$ を適当に決め，式 (45) で $\mathbf{w}^{(2)}, \mathbf{w}^{(3)}, \ldots$ と順次計算していきます．

3.2 確率的勾配降下法

キーワード

バッチ学習，確率的勾配降下法

確率的勾配降下法とは

前節の誤差関数は全サンプルを用いたものでした[*3]．n 番目のサンプル 1 個だけの誤差を $E_n(\mathbf{w})$ と表せば

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}) \tag{46}$$

となります．確率的勾配効果法は更新式 (45) の勾配 ∇E を ∇E_n に置き換えた更新式：

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t+1)} - \epsilon \nabla E_{\mathbf{n}} \tag{47}$$

を用いる方法です．

確率的勾配降下法のメリット

- 訓練データに冗長性がある場合，計算効率が上昇する
- 一つの局所的最小解にハマるリスクが下がる
- 学習の経過を細かく監視できる
- オンライン学習が可能

3.3 勾配計算の難しさ

前節の勾配を試しに，二乗誤差：$E_n = 1/2||\mathbf{y}(\mathbf{x}_n) - \mathbf{d}_n||^2$ に対して，第 l 層の重み $w_{ij}^{(l)}$ に関する微分で計算してみましょう．合成関数の微分から

$$\frac{\partial E_n}{\partial w_{ij}^{(l)}} = (\mathbf{y}(\mathbf{x}_n) - \mathbf{d}_n)^\top \frac{\partial \mathbf{y}}{\partial w_{ij}^{(l)}} \tag{48}$$

と計算できます．さらに中身の微分 $\partial \mathbf{y}/\partial w_{ij}^{(l)}$ も計算できれば，この微分を数値的にも計算できます．だがしかし，第 L 層である出力層から遡って計算するため，微分をする対象の $\mathbf{y}$ は

$$\begin{aligned}
\mathbf{y} &= \mathbf{f}\left(\mathbf{u}^{(L)}\right) && (49)\\
&= \mathbf{f}\left(\mathbf{W}^{(L)}\mathbf{z}^{(L-1)} + \mathbf{b}^{(L)}\right) && (50)\\
&= \mathbf{f}\left(\mathbf{W}^{(L)}\mathbf{f}(\mathbf{W}^{(L-1)}\mathbf{z}^{(L-2)} + \mathbf{b}^{(L-1)}) + \mathbf{b}^{(L)}\right) && (51)\\
&= \mathbf{f}\left(\mathbf{W}^{(L)}\mathbf{f}(\mathbf{W}^{(L-1)}\mathbf{f}(\cdots\mathbf{f}(\mathbf{W}^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{(l)})\cdots)) + \mathbf{b}^{(L)}\right) && (52)\\
& && (53)
\end{aligned}$$

[*2] これは人間が決めます．

[*3] 全サンプルを使用する学習を**バッチ学習**と言ったりします

となり，上式の $\mathbf{W}^{(l)}$ の一要素である $w_{ij}^{(l)}$ で微分しなければならないですが，ご覧のとおり多くの活性化関数の入れ子の中に $w_{ij}^{(l)}$ があるので連鎖規則を何度も適用する必要があり，この微分を計算するのは容易ではありません．これを解決するのが次節から説明する**誤差逆伝播法**です．

3.4 回帰の 2 層ネットワークでの誤差逆伝播法の計算例

3.5 多層ネットワークでの誤差逆伝播法 (一般化)

3.6 勾配降下法のアルゴリズム完全版

4 パラメータ推計に関するトピック

4.1 ミニバッチの利用

4.2 汎化性能と過剰適合

4.3 過剰適合の緩和

4.4 学習のトリック

4.5 勾配消失問題