解説 基専

# ディープラーニングによるパターン認識

久保陽太郎（日本電信電話（株）NTT コミュニケーション科学基礎研究所）

## ディープラーニングとは何か

パターン認識は音声や画像などの入力信号から，それに対応するタグを推定する処理である．具体的には，音声認識の場合は音声信号から単語タグの列を，文字認識の場合は文字画像から対応する文字タグを推定するといった処理を表し，応用先に応じた入力信号／タグの定義を用いる．近年このようなパターン認識の分野において，ディープラーニング（Deep Learning：深層学習）と呼ばれる技術がブレイクスルーを起こし，大きな注目を集めている．たとえば音声認識の分野では，ディープラーニングを利用することによって，これまでの最先端技術を結集して作られた音声認識器の精度をさらに超えることが可能であるといった報告があり，現在では高精度な音声認識器を構築するための重要技術であると考えられている．また，コンピュータビジョンの分野においても，IMAGENET Large Scale Visual Recognition Challenge（ILSVRC）という国際コンペティションで，ディープラーニングに基づく認識器は2位の研究グループが利用した認識器の性能を大きく引き離してトップの性能を示した．

ディープラーニングという言葉は，非常に広い意味を持つ言葉であり，今日までさまざまな意味でのディープラーニングが提案されている．本稿では以降，ディープニューラルネットワーク（Deep Neural Network：DNN）によるパターン認識という意味でディープラーニングという語を用いるが，ほかにも潜在変数が階層的に深い生成モデルの学習や，再帰的にカーネル法による処理を適用するものなどもディープラーニングと呼ばれることがある．

ディープラーニングは，機械学習の技術の1つである．特に本稿ではDNNによるパターン認識を取り上げるため，入力ベクトル $\boldsymbol{x}$ があったとき，それに対応する出力クラス $\hat{k}$ を推定する，いわゆる「教師有り学習」と呼ばれているものを対象とする．教師有り学習の問題は一般的に，入力の一例 $\boldsymbol{x}$ と出力の一例 $k$ の対応スコアを計算する関数 $q(\boldsymbol{x}, k)$ をどのようにして設計するかという点に帰着される．$q(\boldsymbol{x}, k)$ の設計の指針は大別して3通りあり，1つ目は生成モデルアプローチと呼ばれる．生成モデルアプローチでは $q(\boldsymbol{x}, k)$ を $\boldsymbol{x}$ と $k$ の同時分布 $P(\boldsymbol{x}, k)$ であるとして，$q(\boldsymbol{x}, k)$ の設計を行う．2つ目は識別モデルアプローチと呼ばれ，$q(\boldsymbol{x}, k)$ を $\boldsymbol{x}$ を観測した上での $k$ の条件付き分布 $P(k|\boldsymbol{x})$ であるとして，$q(\boldsymbol{x}, k)$ の設計を行う．3つ目は識別関数法と呼ばれるもので，$q(\boldsymbol{x}, k)$ に確率分布としての制約をいっさい置かずに直接設計する方法である[☆1]．どのアプローチの場合であっても，$q(\boldsymbol{x}, k)$ があれば，入力に対する出力の推定値 $\hat{k}$ は，すべての出力候補 $k$ について $q(\boldsymbol{x}, k)$ を評価することによって，$\hat{k} = \mathrm{argmax}_k q(\boldsymbol{x}, k)$ のように求めることができる．本稿で取り扱うディープラーニングではDNNを用いてこの $q(\boldsymbol{x}, k)$ を設計することを考える．上述の3通りの区分ではDNNは一般的に識別モデルアプローチか直接関数法に属するとされることが多く，そのため同じアプローチであると見なされるロジスティック回帰モデルやサポートベクタマシン（Support Vector

☆1 文献によってはこの識別関数法と識別モデルによるアプローチを区別せず，識別アプローチと呼ぶ場合もある．

Machine : SVM）と比較されることが多い．

DNN とは，層数の多い多層パーセプトロン（Multi-Layer Perceptron : MLP）のことを指す．これまで層数の多い MLP はその最適化が難しく，有効に利用することが難しいとされてきたが，近年，学習法の進展や計算機性能の向上などによってそれが可能になり，先述したようなさまざまな実験でその有効性が検証されるようになってきている．本稿ではまず，なぜ DNN のパターン認識への応用，すなわちディープラーニングが難しかったのかについて解説し，続けてなぜそれが可能になったのかについて解説した後，実際にどのように応用されているかについての事例を紹介する．

## 多層パーセプトロン（MLP）の基礎

2 層 MLP の模式図を**図 -1** に示す．MLP は $D$ 次元入力ベクトル $\boldsymbol{x} \overset{\text{def}}{=} [x_1, x_2, \cdots x_D]^{\mathsf{T}}$ が入力されたときの $D'$ 次元出力ベクトル $\boldsymbol{y}(\boldsymbol{x}) \overset{\text{def}}{=} [y_1(\boldsymbol{x}), y_2(\boldsymbol{x}), \cdots y_{D'}(\boldsymbol{x})]$ を以下のように再帰的な関数で表すモデルである．

$$
\begin{aligned}
&y_i(\boldsymbol{x}) = h_i^{(L)}(\boldsymbol{x}) \\
&h_i^{(\ell)}(\boldsymbol{x}) = \sigma^{(\ell)}\left(z_i^{(\ell)}(\boldsymbol{x})\right),\ h_d^{(0)}(\boldsymbol{x}) = x_d, \\
&z_i^{(\ell)}(\boldsymbol{x}) = \sum_{j=1}^{D^{(\ell-1)}} \boldsymbol{w}_{i,j}^{(\ell)} h_j^{(\ell-1)}(\boldsymbol{x}) + b_i^{(\ell)}.
\end{aligned}
\tag{1}
$$

ここで $L$ は層数でありこれを大きく設定した MLP を DNN と呼ぶ（たとえば $L \geq 3$）．$\sigma^{(l)}(.)$ は活性化関数（activation function）と呼ばれる関数であり，目的に応じてさまざまなものが用いられる．$j$ 個目の出力 $y_j(\boldsymbol{x})$ は最終層，すなわち $L$ 番目の層の $j$ 番目の活性化状態 $h_j^{(L)}(\boldsymbol{x})$ であり，活性化状態は活性化関数 $\sigma^{(L)}$ を入力量 $z_j^{(L)}$ に適用することによって得られる．入力量はその 1 つ前の層の活性化状態 $h_i^{(L-1)}(\boldsymbol{x})$ の重み付き和とバイアス項 $b_j^{(L)}$ の和で与えられ，その重み係数は $\boldsymbol{w}_{j,i}^{(L-1)}$ のように表される．$D^{(\ell)}$ は $\ell$ 層目のユニット数と言われる数であり，各層において何次元の活性化状態を持つかということを示す．入力および出力の次元数も同様に $D^{(0)} = D$, $D^{(L)} = D'$ と表す．MLP を用いたパターン認識では，入力ベクトルが $\boldsymbol{x}$，タグが $k$ のとき，$q(\boldsymbol{x}, k)$ を $q(\boldsymbol{x}, k) = y_k(\boldsymbol{x})$ のようにして定義することでパターン認識を行う（ここでタグは自然数で表されているとする）．

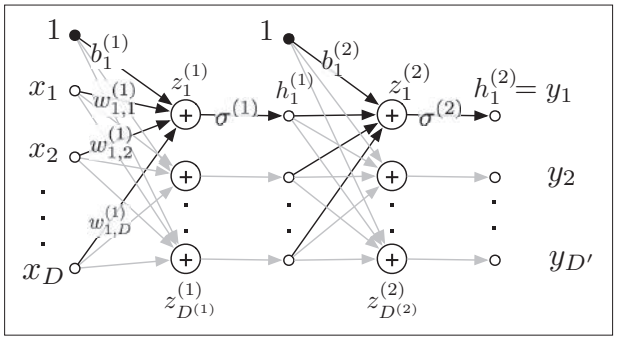

**図 -1　2 層パーセプトロン**

識別モデルアプローチでは $q(\boldsymbol{x}, k)$ は確率分布 $P(k|\boldsymbol{x})$ であることが要請される．出力ベクトルとして確率分布を表現するためには，出力値が非負値であり総和が 1 になるよう，最終層の活性化関数 $\sigma^{(L)}$ として，以下で示すようなソフトマックス活性化関数（softmax activation function）を利用する．

$$
\sigma^{(L)}\left(z_i^{(L)}(\boldsymbol{x})\right) = \frac{\exp\left\{z_i^{(L)}(\boldsymbol{x})\right\}}{\sum_{i'} \exp\left\{z_{i'}^{(L)}(\boldsymbol{x})\right\}}\ . \tag{2}
$$

また，ほかの層に対応する活性化関数としては，以下に示すシグモイド活性化関数（sigmoid activation function）が広く用いられている．

$$
\sigma^{(L)}\left(z_i^{(L)}(\boldsymbol{x})\right) = \left(1 + \exp\{-z_i^{(L)}(\boldsymbol{x})\}\right)^{-1}. \tag{3}
$$

DNN の学習は一般的に，層数 $L$ やユニット数 $D^{(\ell)}$，活性化関数に関する設定を事前に決め，重み係数 $\boldsymbol{w}_{j,i}^{(\ell)}$ およびバイアス項 $b_j^{(\ell)}$ を自動最適化することによって行われる．

最適化の基準として，広く用いられているものは最大相互情報量基準と呼ばれる基準である．最大相互情報量基準では学習用データ $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots \boldsymbol{x}_n \cdots, \boldsymbol{x}_N\}$ と，それに対応するラベルデータ $\mathcal{K} = \{k_1, k_2, \cdots k_n \cdots, k_N\}$ を用いて，以下の最適化を実行することにより，最も適切なパラメタを得る．

$$
\text{minimize}_{\Theta} \sum_n \underbrace{-\log q(\boldsymbol{x}_n, k_n) + \frac{1}{N}\Omega(\Theta)}_{\mathcal{L}_n}. \tag{4}
$$

ここで $\Theta$ はパラメタ集合であり，$\Theta \stackrel{\text{def}}{=} \left\{ w_{j,i}^{(\ell)},\ b_j^{(\ell)} \mid 1 \le i \le D^{(\ell)},\ 1 \le j \le D^{(\ell-1)},\ \forall \ell \right\}$ である．また，$\mathcal{L}_n$ は $n$ 番目の学習データに対応する目的関数の項であり，サンプル $n$ に関するロス関数と呼ばれる．$\Omega(\Theta)$ は正則化項と呼ばれるもので，パラメタに対する事前知識を反映するよう設計することで，パラメタの値が不自然なものになることを避けるために導入される．具体的には $w_{i,j}^{(\ell)}$ は大きな値ではないという事前知識を反映させ，小さな正の定数 $\gamma$ を用いて $\Omega(\Theta) = \gamma \sum_{\ell,i,j} | w_{i,j}^{(\ell)} |^2$ のような設計をすることが多い．

この最適化は非凸関数の制約なし最適化問題であり，さまざまなアルゴリズムで局所最適解を求めることができる．確率的勾配降下法（Stochastic Gradient Descent：SGD）では，以下に示す更新則でパラメタを逐次更新していくことで局所最適解の近似値を得る．

$$w_{j,i}^{(\ell)} \leftarrow w_{j,i}^{(\ell)} - \frac{\eta}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \frac{\partial \mathcal{L}_r}{\partial w_{j,i}^{(\ell)}},\ b_j^{(\ell)} \leftarrow b_j^{(\ell)} - \frac{\eta}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \frac{\partial \mathcal{L}_r}{\partial b_j^{(\ell)}} . \tag{5}$$

ここで $\mathcal{R}$ はトレーニングデータのインデックス番号（$\{1,\cdots, N\}$）からランダムに少数個，各更新ステップのたびに抽出したインデックスの集合であり，$\eta$ は学習率と呼ばれるハイパーパラメタである．

## ■DNN と関係の深い機械学習法

本節では DNN とロジスティック回帰の関係や，DNN とカーネル法の関係について述べる．

最終層（$L$ 層目）の活性化関数をソフトマックス活性化関数（式（2））とした場合，MLP は $L$–1 層目の活性化状態を素性（入力）とした多クラスロジスティック回帰モデルであると考えることができる．また，シグモイド活性化関数 $\sigma_i^{(\ell)}(.)$（式（3））はその前の活性化状態を素性とした 2 クラスロジスティック回帰モデルのどちらか片方のクラスの確率であるとみなすことができる（もう片方のクラスの確率は $1-\sigma_i^{(\ell)}(.)$）．2 クラスロジスティック回帰モデルは入力ベクトルの線形識別処理を行っていると解釈することができるため，DNN は階層を増やすことによって入力と出力の間の関係を再帰的な線形識別処理によって表現しようとしていると考えることができる．

ロジスティック回帰や SVM が，十分に前処理（特徴抽出）を行った表現力の高い入力ベクトル $\boldsymbol{x}$ を仮定し，代わりに $q(\boldsymbol{x}, k)$ を単純な線形関数で表現することによって，理論的見通しの良さや，過学習と呼ばれる問題の回避を行っていたのに対し，ディープラーニング技術では，画像であれば画素値，音声であればバンドパスフィルタの出力値といったような，ほぼ前処理を行わない入力を仮定し，代わりに $q(\boldsymbol{x}, k)$ として再帰的に定義された複雑な関数を用いて複雑な入出力関係を表現する．こうした違いのため，ほかの手法のように，大域最適解が求まるという保証や，汎化性能の上界の評価といったようなことは難しくなっているが，アプリオリに導入した特徴抽出法と異なり，特徴抽出の方法も含め，データに合わせて最適化できるという利点を手に入れている．

$q(\boldsymbol{x}, k)$ の複雑さと，理論的解析の見通しのよさを両立する手法としてはカーネル法がある．カーネル法は最初に $\boldsymbol{x}$ を関数 $\phi$ によって超高次元に写像し，$q(\boldsymbol{x}, k) = g(\phi(\boldsymbol{x}), k)$ を $\phi(\boldsymbol{x})$ に関して線形の関数としてモデル化することによって $g$ を線形識別の関数に保ったまま，複雑な識別処理を可能とする．活性化関数と重み付き和を合成した特徴抽出関数 $\psi^{(.)}(\boldsymbol{x})$ を導入すると DNN も同様に，$q(\boldsymbol{x}, k) = g(\psi^{(L-1)}(\psi^{(L-2)}(\cdots \boldsymbol{x} \cdots)), k)$ と，特徴抽出関数と線形識別の関数を用いて記述することができる．カーネル法と DNN の大きな違いは特徴抽出関数の再調整を行うか行わないかという点と，それをネストするかしないかという点である．ここで特筆すべきは，DNN のようなネストした特徴抽出処理を行わなくても $q(X, Y)$ は $\phi$ の次元数が十分に高ければ任意の入出力関係を十分な精度で近似できることが証明されている点である．このことは，近似の精度という観点ではネスト構造は不要であるということを表している．しかし，ネストした線形識別処理によって入力から出力が生成されるようなデータに関して，ネスト構造を無視して関数を近似しようとすると，生成

に用いられた識別器の数に対し，必要な識別器の数が指数的に増加することが知られている[1]．線形識別関数 $g$ における入力次元数やパラメタ数の増加は，汎化性能の悪化に繋がることがさまざまな文献で指摘されており，本質的にネストした構造を持つデータを扱う際にはDNNが有効だということが推察できる．

## なぜ今ディープラーニングなのか

ニューラルネットワークによるパターン認識の研究の起源は1950年代まで遡るが，特徴抽出と識別処理を統一的に扱うことが可能となったのは1986年にRumelhartがバックプロパゲーション法（Back Propagation（BP）法）を提案して以降のことである．BP法は，活性化関数として従来使われてきたステップ関数を微分可能な関数で置き換え，目的関数を全パラメタについて微分可能にした上で，勾配降下法によってパラメタを調整する方法である．

BP法によって学習されるDNNは，特徴抽出のプロセスと識別のプロセスをすべて1つの最適化基準で最適化できる柔軟なモデルである．特に層数が十分に存在する場合，高度な特徴抽出の処理を，中間表現の次元数を高くすることなく高度な識別を行うことができるという点で，汎化性能の面からも期待できるモデルであった．しかし，実際は，後述するような問題点があり，その真価を発揮できないまま，SVMに代表されるような，より高度な“浅い”識別器にとって代わられた．本章では，なぜ，これまで層数の多いDNNが学習できなかったのかと，なぜ近年これらが可能になり再注目を集めるに至ったのかについて解説する．

### ■なぜ学習ができなかったのか

これまでDNNの学習が難しかった理由には計算機性能の不足や学習データセットの不足といった学習に利用可能な資源の問題もあったが，本節ではアルゴリズムの面で大きな問題であったと考えられるVanishing Gradientについて解説する．

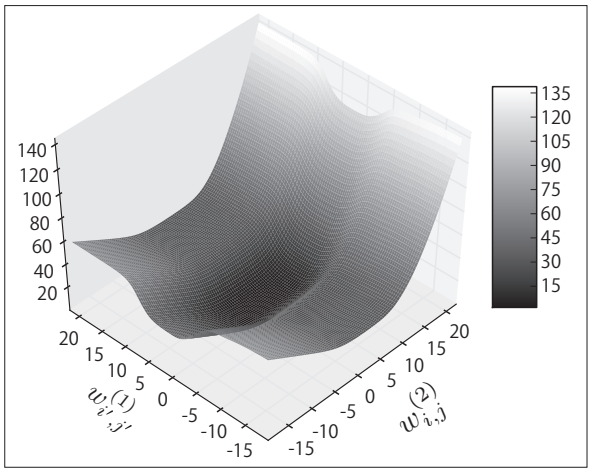


図-2　2層パーセプトロンの目的関数

**図-2**に2層パーセプトロンの局所解の近傍の目的関数の値を，2つのパラメタを変化させプロットした図を示す．図中 $w^{(1)}_{i',j'}$ と示した軸が入力信号を直接処理する層（$\ell$=1）の1つの重み係数であり，$w^{(2)}_{i,j}$ と示した軸が出力値を算出するのに直接使われる層（$\ell$=2）の1つの重み係数である．図で示される通り，2層パーセプトロンの目的関数は出力に近い層の重み係数については大きな変動を示すが，入力に近い層の重み係数については大きな変動を示さない．また，変動は小さいとはいえ，$w^{(1)}_{i',j'}$ の軸に関しても中心部に明確に谷があり，入力に近い層も適切に最適化しなければ局所最適解を得ることができない．このようにパラメタが張る空間で目的関数の勾配が大きい方向と小さい方向が混在している場合，勾配降下法は現実的な時間で収束しないことが知られている[2]．

このような目的関数となってしまう原因はDNNおよびその最適化の目的関数の定義にある．BP法では目的関数のパラメタについての偏微分を評価し，それに基づいてパラメタを式（5）のように更新する．この偏微分係数を具体的に計算するための式は以下のようになる（簡単のため重みパラメタ $w^{(\ell)}_{i,j}$ についてのみ書く）．

$$\frac{\partial \mathcal{L}_r}{\partial w^{(\ell)}_{i,j}} = \frac{\partial \mathcal{L}_r}{\partial z^{(\ell)}_j(\boldsymbol{x})} h^{(\ell)}_i(\boldsymbol{x}) + \frac{\gamma}{N} w_{i,j}\ ,$$
$$\frac{\partial \mathcal{L}_r}{\partial z^{(\ell)}_j} = s^{(\ell)}(z^{(\ell)}_j) \sum_i w^{(\ell)}_{j,i} \frac{\partial \mathcal{L}_r}{\partial z^{(\ell-1)}_i(\boldsymbol{x})}\ . \tag{6}$$

この式のように，ある層 $\ell$ に対応する偏微分係数は

前の層 $\ell-1$ の偏微分係数を現在の重み行列の値で重み付き和を取った後，活性化関数の導関数の値をかけることによって得られる．これまで一般的に使われてきた活性化関数の導関数では，活性関数の導関数の値は 1 未満のものが多く，層を重ねるごとに勾配が減衰してしまうということが起こる．仮に大きな導関数を持つ活性化関数が用いられていたとしても，今度は勾配の振動が起こってしまうという問題を抱えることになる．

この問題はこれまでもさまざまな形で議論され，たとえば入力量を単なる重み付き和から高次の関数に拡張することでそれを防ぐという試み（Sigma-Pi ネットワークや Long-Short Term Memory（LSTM）ネットワーク），自然勾配法や Resilient Prop（RProp）に代表されるような勾配のスケールに頑健な最適化法，また入力層のパラメタを複数の活性化状態で共有する畳み込みニューラルネットワーク（Convolutional Neural Network：以下 CNN）が考案されてきた．しかし，これらの手法がより深く検討される前に，ほかの手法が注目を集めることとなり，DNN を有効に使うことの意義は薄れていった．

## ■なぜ再注目に至ったのか

DNN が再注目されるきっかけの 1 つとなったのが事前学習手法の発達である．事前学習法とは BP 法に先立ってパラメタの値を推定しておくことであり，これまでランダムに設定していた最適化の初期値を別の手法で推定することによって行われる．事前学習法自体は初期値を与えるだけであり，続けて行われる実際の学習法によっては，その効果がほとんど失われてしまうヒューリスティクスである．より洗練された最適化アルゴリズムを使えば，事前学習の効果は薄れ，場合によっては不要になるかもしれないという意見もあるが，DNN 研究の再注目のきっかけとなったのは事前学習法に基づく DNN の登場であった．

事前学習にはさまざまな手法が提案されているが，大別して Autoencoder を用いたものと Restricted Boltzmann Machine（RBM）を用いたものに分けられる．これら 2 つの手法で共通していることは，事前学習時にラベル情報，すなわちシステムの出力 $k$ を無視し，入力 $\boldsymbol{x}$ の情報を損なわないような多層の特徴抽出器を教師なし学習するという点，層数の多い特徴抽出器を直接最適化することを避け単層特徴抽出器を 1 つずつ最適化することで初期値を得ようとしている点である．

事前学習がディープラーニングに効果的である理由としてはさまざまな検討がなされている．具体的には，SGD との組合せにおいて収束先の一意性を向上させるといったことや，汎化性能の向上に役立つということが実験を通して示されている [3]．SGD では Vanishing Gradient の影響を排除できないことから，入力に近い層の重み係数はほぼ初期値がそのまま利用されてしまうことも多い．しかし，初期値として入力ベクトルの情報を保った別の表現を得ておけば，入力に近い層の調整が不可能であったとしても，他層の学習に悪影響を与えることがない．このようなことが，収束先の一意性の向上に貢献していると考えられる．

$L$ 層の DNN を学習データ $\mathcal{X}$ から学習する際の事前学習を用いた学習アルゴリズムの概略を**図 -3** に示す．アルゴリズム中の $h(.)$，$\tilde{\mathrm{W}}(.)$，$\tilde{\boldsymbol{b}}(.)$，$\Phi(.)$ は特徴抽出器として RBM を用いるか Autoencoder を用いるかで定義が変化する部分であり，以降に詳述する．これら 2 つの事前学習アルゴリズムではニューラルネットワークと同じ構造を持つ単層特徴抽出器を導入し，トレーニングデータ $\mathcal{X}$ に対応する特徴ベクトル集合 $\mathcal{F}$ を，学習された単層特徴抽出器 $\hat{\Lambda}$ を利用することによって計算する．そうして得られた特徴ベクトル集合 $\mathcal{F}$ に対応する特徴を用いてまた別の単層特徴抽出器を学習し，再度特徴抽出を行うということを繰り返す．このとき，各ステップで利用した単層特徴抽出器を DNN の初期値として利用する．一般的に識別器の出力を計算する層，すなわち最終層は事前学習を行わずに小さな分散 $\epsilon$ を持つ正規乱数によって初期化することが多い．

Algorithm1　事前学習を含めた学習法

1：$\mathcal{F} \leftarrow \mathcal{X}$
2：**for** $\ell = 1$ to $L-1$ **do**
3：単層特徴抽出器の学習：$\hat{\Lambda} = \mathrm{argmin}_{\Lambda}\, h(\Lambda; \mathcal{F})$
4：単層特徴抽出器から DNN のパラメタを抽出：
$\mathrm{W}^{(\ell)} \leftarrow \tilde{\mathrm{W}}(\hat{\Lambda})$, $\boldsymbol{b}^{(\ell)} \leftarrow \tilde{\boldsymbol{b}}(\hat{\Lambda})$
5：単層特徴抽出器の適用：$\mathcal{F} \leftarrow \Phi(\mathcal{F})$
6：**end for**
7：$w_{i,j}^{(L)} \sim \mathcal{N}(0, \epsilon), b_j^{(L)} = 0 \quad \forall i, \forall j$
8：$\hat{\Theta} = \mathrm{argmin}_{\Theta}\ \sum_n \mathcal{L}_n$ を得られた初期値
$\mathrm{W}^{(\ell)}$ および $\boldsymbol{b}^{(\ell)}$ からスタートする BP 法で計算

**図 -3　Algorithm1**

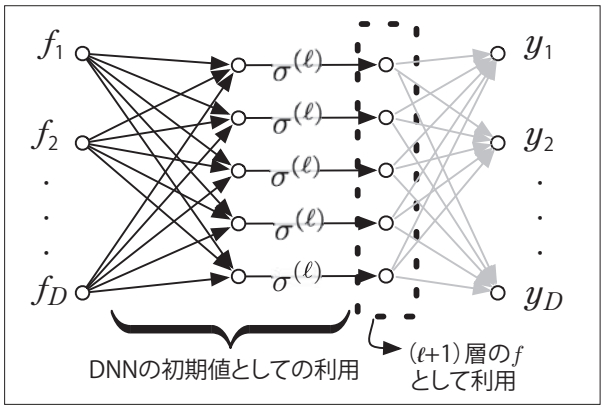


**図 -4　Autoencoder**

## Autoencoder

Autoencoder は MLP を用いて構成された特徴抽出器であり，ディープラーニングが再注目される以前より，非線形主成分分析のためのニューラルネットワークとして研究されていたものに近い．

**図 -4** に Autoencoder の模式図を示す．

Autoencoder では式（1）の定義に従い，$L = 2$ のニューラルネットワークを考え，そのパラメタを $\Lambda = \{\boldsymbol{w}_{i,j}^{(\ell)}, b_i^{(\ell)}\}$ と置く．ここで，最初の隠れ層の活性化関数 $\sigma^{(1)}$ として一般的にシグモイド関数 (Eq. (3))，また出力層の活性化関数として恒等関数 $\sigma^{(2)}(z) = z$ を考える．また，出力層の次元数 $D' = D^{(2)}$ は入力の次元の数 $D = D^{(0)}$ と同じにしておく．

このようにして定義したニューラルネットワークを以下の基準で最適化する．

$$\mathrm{minimize}_{\Lambda} \sum_n D(\boldsymbol{f}_n, \boldsymbol{y}(\boldsymbol{f}_n)). \tag{7}$$

ここで $D(\boldsymbol{f}_n, \boldsymbol{y}(\boldsymbol{f}_n))$ は $\boldsymbol{f}_n$ と $\boldsymbol{y}(\boldsymbol{f}_n)$ の非類似度を測る関数である．一般的に $D$ はユークリッド距離が用いられる．Autoencoder 自身も BP 法によって最適化されるが，層数が多くない ($L = 2$) ため，ランダムに設定した初期値からでも有効な局所解を得ることができると考えられる．

このようにして得られた Autoencoder は，入力信号を活性化状態で示される特徴ベクトルに非線形変換した上で，その特徴から再度，元の入力信号と十分に近い出力を再構成することができる．これは後述する RBM と同様，入力ベクトルと同じ情報を持つ特徴ベクトルを作る方法をニューラルネットワークの一層分と同じ形で得ていることになる．元の表現 $\boldsymbol{f}_n$ に対応する特徴表現は $\ell = 1$ における活性化状態を並べたベクトル $[h_1^{(\ell)}(\boldsymbol{f}_n), h_2^{(\ell)}(\boldsymbol{f}_n), \cdots]^{\mathsf{T}}$ となり，対応するニューラルネットワークパラメタは $\tilde{\mathrm{W}}(\Lambda) = (\mathrm{w}^{(1)})_{i,j}$, $\tilde{\boldsymbol{b}} = (b^{(1)})_i$ となる．

Autoencoder は，その柔軟な定式化を活かし Denoising Autoencoder や Sparse Autoencoder，Contractive Autoencoder などといったようなさまざまな拡張が提案されているのが特徴である．応用事例にあわせて適切な Autoencoder を用いることができればパターン認識の精度向上にも有用であると考えられる．

## RBM

RBM は観測変数ベクトル $\boldsymbol{v}$ と隠れ変数ベクトル $\boldsymbol{h}$ の関係を記述する確率モデルであり，事前学習の文脈では $\boldsymbol{v}$ の情報を十分に持つ隠れ変数 $\boldsymbol{h}$ を特徴とするために導入される．RBM では $\boldsymbol{v}$ と $\boldsymbol{h}$ の同時分布を以下のように定義する．

$$P(\boldsymbol{v}, \boldsymbol{h}) = \frac{\exp\{-E(\boldsymbol{v}, \boldsymbol{h})\}}{\sum_{\boldsymbol{v}'} \sum_{\boldsymbol{h}'} \exp\{-E(\boldsymbol{v}', \boldsymbol{h}')\}}. \tag{8}$$

観測変数や隠れ変数として連続変数を仮定する場合は上式の総和の操作を積分の操作に置き換えて定義する．$\boldsymbol{v}$ と $\boldsymbol{h}$ の対応を表すエネルギー関数 $E(\boldsymbol{v}, \boldsymbol{h})$ は任意のものが利用できるが，特に $P(\boldsymbol{h}|\boldsymbol{v})$ がニューラルネットワークと同様の計算によって計算できるものを RBM と呼ぶ．

パターン認識の文脈で広く用いられているエネルギー関数は以下の2種類である．最初の形はBernoulli-Bernoulli RBMと呼ばれ観測変数が0から1の値を取る場合に利用される．

$$E(\boldsymbol{v},\boldsymbol{h}) = -\boldsymbol{v}^{\mathsf{T}}\boldsymbol{c} - \boldsymbol{h}^{\mathsf{T}}\boldsymbol{b} - \boldsymbol{v}^{\mathsf{T}}\mathrm{W}\boldsymbol{h}. \quad (9)$$

次の形はGaussian-Bernoulli RBMと呼ばれる形で，観測変数が実ベクトル，隠れ変数がバイナリベクトルである場合に利用される．

$$E(\boldsymbol{v},\boldsymbol{h}) = \| \boldsymbol{v} - \boldsymbol{c} \|^2 - \boldsymbol{h}^{\mathsf{T}}\boldsymbol{b} - \boldsymbol{v}^{\mathsf{T}}\mathrm{W}\boldsymbol{h}. \quad (10)$$

これらのエネルギー関数を用いた場合，RBMのパラメタは$\Lambda = \{\boldsymbol{b}, \boldsymbol{c}, \mathrm{W}\}$であり，学習の目的関数（Algorithm 1, Line 4）は最尤推定に基づいて以下のように示される．

$$h(\Lambda;\mathcal{F}) = -\sum_n \log \sum_{h'} P(\boldsymbol{v} = \boldsymbol{f}_n, \boldsymbol{h}'). \quad (11)$$

この目的関数についての最適化はすべての取り得る隠れ変数の値に関する総和を取り扱わなければならないため，SGD等では効率的に計算できない．そのため，Contrastive Divergence法[4]がRBMの学習によく用いられる．

このモデルからMLPのパラメタを取り出す処理（Algorithm 1, Line 5）は，$\tilde{\mathrm{W}}(\hat{\Lambda}) = \mathrm{W}$, $\tilde{b}(\hat{\Lambda}) = b$と定義する．また，このモデルから特徴ベクトルを取り出す処理（Line 5）は以下のようにサンプリングを用いて定義する．

$$\Phi(\mathcal{H}) = \{\boldsymbol{h}'_n \sim P(\boldsymbol{h} \mid \boldsymbol{v} = \boldsymbol{f}_n) \forall n\}. \quad (12)$$

サンプリングを行う代わりに期待値を計算して特徴とすることもよく行われる．以上のようにしてRBMは，入力の情報を十分に反映する特徴抽出器をニューラルネットワークと同様の構成，すなわち線形変換と活性化関数のペアで実現する．

RBMの利点は，確率的なプロセスが最適化時や特徴抽出時に利用されるため，頑健な特徴が得られていることが期待できる点である．実際，RBMによる事前学習は多くの実験で単純なAutoencoderによる事前学習と比較して良い性能を達成している．

## ディープラーニングの応用事例

上述の手法を用いて構成されたDNNが，さまざまな応用分野で高い性能を発揮している．本章では，その例として音声認識と一般物体認識の2つを取り上げ，紹介する．

### ■音声認識

音声認識は音声分析結果である$D$次元ベクトルの時系列$\mathrm{X}=\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots \boldsymbol{x}_t \cdots | \boldsymbol{x}_t \in \mathbb{R}^D\}$から，対応する単語の列$\boldsymbol{w}$を推定する問題である．入力Xと出力$\boldsymbol{w}$の関係を記述するため，確率分布$P(\boldsymbol{w}|\mathrm{X})$を導入し，音声Xが観測されたときの出力$\hat{\boldsymbol{w}}$を以下のように定義する．

$$\hat{\boldsymbol{w}} = \operatorname*{argmax}_{\boldsymbol{w}} P(\boldsymbol{w} \mid \mathrm{X}) = \operatorname*{argmax}_{\boldsymbol{w}} P(\mathrm{X} \mid \boldsymbol{w}) P(\boldsymbol{w}). \quad (13)$$

ここで$P(\mathrm{X}|\boldsymbol{w})$を隠れマルコフモデル（Hidden Markov Mode：以下HMM）と呼ばれる確率モデルでモデル化することを考える．Xの系列長と同じ長さの潜在離散変数の系列$s$が1次のマルコフモデルに従って生成されることを仮定し，以下のように表す．

$$P(\mathrm{X} \mid \boldsymbol{w}) = \sum_{s_1, s_2, \cdots} \prod_t P(\boldsymbol{x}_t | s_t) P(s_t \mid s_{t-1}, \boldsymbol{w}). \quad (14)$$

ここで，$P(\boldsymbol{x}_t|s_t)$を以下のように変形し，下式中の$P(s_t|\boldsymbol{x}_t)$を$s_t$を出力タグ，$\boldsymbol{x}_t$を入力としてDNNでモデル化することによってDNNによる高度な識別能力をHMMに統合する．

$$P(\boldsymbol{x}_t \mid s_t) = \frac{P(s_t \mid \boldsymbol{x}_t)}{P(s_t)} P(\boldsymbol{x}_t). \quad (15)$$

潜在変数のユニグラム確率$P(s_t)$は別途最尤推定などによって推定する．また$P(\boldsymbol{x}_t)$は認識時に影響を与えないため定数として扱う．DNNは特徴抽出の処理を内包するため，DNNを用いる場合，入力ベクトル$\boldsymbol{x}_t$として従来手法で用いていたものより単純なものを利用することができる．

このようにして作った音声認識器を用いることで，Switchboardと呼ばれる，広く用いられているデータセットにおいて，従来手法のエラー率27.4%

を 18.5% まで下げることができるという報告があり，音声認識の研究者を驚かせた．音声認識でこれだけの精度改善を起こした理由の 1 つに，従来のモデルでは扱えなかった長い区間の音声情報を，明示的な特徴抽出によって縮約することなく，識別器と一体化した最適化の枠組みによって利用できるようになったという点が考えられる．

音声認識で DNN を利用する場合，Kaldi ツールキットが有望である．Kaldi における DNN 機能は 2013 年 2 月現在，開発版（trunk）のみでの提供であり，また利用方法に関するドキュメントも整備されていないが，近いうちに利用可能になるだろうと考えられる．既存のプログラムを利用せずに，一から DNN のプログラムを構成することを考える場合は Theano による行列演算の利用も有効である．Theano は Python で書かれたライブラリであり，Python の枠組みを利用して簡易的な数式処理を行い，CPU と GPU で動作する最適なコードを生成／実行する．数式処理によるコード生成を普通の Python のプログラムと混ぜて記述するには多少の慣れが必要であるが，十分に高速な計算ルーチンを比較的簡単に記述できることは，特に DNN の実装においては重要である．

## ■一般物体認識

冒頭に述べたように画像パターンの認識でも，たとえば ILSVRC コンペティションのシステムのようにディープラーニングが応用されている．ILSVRC のシステムの場合，DNN の利用方法の面では単純であるが，DNN の構成そのものに画像特有の構造を導入した点と，新しい学習法が導入されている点が重要である．ILSVRC で用いられた DNN は CNN と呼ばれるもので，線形フィルタの畳み込み処理に相当するパラメタ共有構造を持つ．また活性化関数も，ひろく用いられているシグモイド活性化関数ではなく，Rectied Linear Unit（ReLU）が用いられている．さらに各活性化状態を縮約するプーリング層および活性化状態を正規化するコントラスト正規化が導入されている．学習においては，Dropout 法と呼ばれるアンサンブル学習を効率的に近似実行するための枠組みが導入され，効果を上げている．CNN，プーリング層，コントラスト正規化については画像処理の知見がニューラルネットワークに統合された興味深い事例となっているが，紙幅の都合上本稿では取り上げない．文献 5）に詳細な解説があるので，それを参照されたい．本節では，ほかの分野での応用も期待される，ReLU および Dropout 法について紹介を行う．

ReLU は活性化関数として，以下のヒンジ型関数を使うものである．

$$\sigma^{(L)}\left(z_i^{(L)}(\boldsymbol{x})\right) = \max\{0, z_i^{(L)}(\boldsymbol{x})\}. \tag{16}$$

この単純な活性化関数が導入することで活性化状態をスパースな状態に保つことができる上に，勾配の計算を単純化できることから，学習の計算量を大幅に削減できることが報告されている [6]．

Dropout 法は複数の挙動の異なる複数の識別器を平均化することによって頑健な識別を行うアンサンブル学習のアイディアを DNN に導入したものであると考えることができる．アンサンブルを構成する手段としてはさまざまな方法が考えられるが，Dropout 法では 1 つの DNN から，重み係数を変化させずに，一部の隠れユニットを利用しないようにすることで複数の DNN のバリエーションを作り出す．こうした識別器群の同時学習は，単に学習時の各ステップにおいてランダムに選ばれたユニットを存在しないものとして学習するだけでよい．このようにして学習した DNN 群を，利用時にはすべて用いることで，識別結果の平均化を行い，認識精度を高めることができる．Dropout 法は，ほかのさまざまな識別タスクにおいても，その有効性が検証されており，今後さらにさまざまな分野で使われていくことが予想される．

一般物体認識や，その他画像に関連する各種パターン認識のために DNN を用いる場合，“cuda-convnet” というツールキットが有用である．cuda-convnet は C++ と CUDA を用いて書かれた，画像認識用の DNN のツールキットであり，本節で紹介した

事例で用いられたCNN，プーリング層，コントラスト正規化，ReLUの実装を提供している．このツールキットの主題は名前にあるように，CNNの実装であるが，一般のDNNの学習も可能である．cuda-convnetはnVidiaのFermiアーキテクチャに準ずるGPUを搭載した計算機でしか動作しないが，大規模なDNNを構築するにあたりGPUを用いることは重要である．

## ディープラーニングの課題と展望

ディープラーニングが抱えている最大の課題は，その理論的取り扱いの難しさであろう．ニューラルネットワークは層を深くしていく以外にもさまざまな構成をとることが可能であり，また応用分野に応じて，適した構成をとるといった試みも古くからなされている．しかし，一般的にどのような問題にどのような構成が優れているのか，といったことを探求する取り扱いの容易なモデル選択アルゴリズムはいまだ提案されていないというのが現状である．

同様に，内部的にどのような操作が行われているのかを理解することが困難であるという点もある．ディープラーニングの各種応用では，人間の脳の構造を模して各層の結合トポロジを設計するという試みも多く，実験的に良い性能が出ているものも少なくない．しかし，各層の活性化状態が実際に何を意味しているのかという点について，推測の域を超えて議論することは現状難しく，設計したネットワークがなぜ有効なのかを説明することが難しいという問題がある．

このような課題を含みながらも，ディープラーニングがさまざまなパターン認識関連タスクにおいて，最高の性能をマークしていることは事実であり，今後もさまざまな応用分野で利用されていくことが予測される．特に，パターンの識別と特徴抽出が同時に一枚岩のモデルとして最適化できるのは好ましく，これまで各応用分野に特化した知識が必要であった前処理の部分が単純化されることから，異なる分野間の技術の交流も従来より容易になることが期待される．さまざまな応用分野にわたる多面的な検討を通して，ディープラーニングの根幹にかかわる理論にも新たな知見がもたらされることを期待したい．

**参考文献**

1) Bengio, Y. : Learning Deep Architectures for AI, Foundations and Trends in Machine Learning, Vol.2, No.1, pp.1-127 (2009).
2) 中野良平：ニューラル情報処理の基礎数理，数理工学社 (2005).
3) Erthan, D., Bengio, Y., Courville, A., Manzagol, P. A., Vincent, P. and Bengio, S. : Why Does Unsupervised Pre-training Help Deep Learning?, Journal of Machine Learning Research, Vol.11, pp.625-660 (2010).
4) Carreira-Perpinan, M. A. and Hinton, G. E. : On Contrastive Divergence Learning, Proc. AISTATS (2005).
5) 岡谷貴之，齋藤真樹：ディープラーニング，信学技法，Vol.2013-CVIM-185, No.19, pp.111-127 (2013).
6) Glorot, X., Bordes, A. and Bengio, Y. : Deep Sparse Rectifier Neural Networks, Proc. AISTATS (2011).

(2013年2月5日受付)

■ **久保陽太郎（正会員）** kubo.yotaro@lab.ntt.co.jp

2010年早稲田大学基幹理工学研究科博士課程修了，博士（工学）．RWTHアーヘン大学客員研究員を経て，日本電信電話（株）に入社．音声認識の研究に従事．