

ソフトコンピューティング(2/3)

- ソフトコンピューティングとは、計算機科学、人工知能、機械学習などの総称
- 複雑な事象の研究・モデル化・解析を行うもの
- 従来の手法では計算コスト等の面で対処できないこと、あるいは従来手法では解析できないこと、完全な解法が見つかっていないことなどを対象とする

従来手法 = 人がルールや解法を設計する手法

遺伝的アルゴリズム

- 遺伝的アルゴリズムは、生物の進化（遺伝）の機構を模倣した確率的探索手法

選択

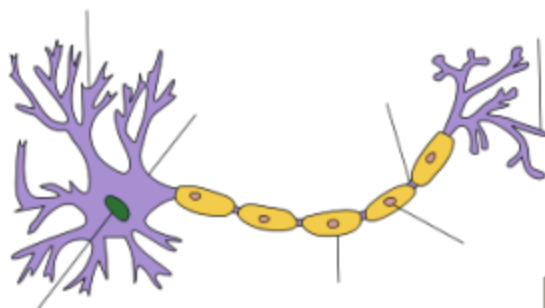
交叉

突然変異

- 大規模組合せ最適化問題、多峰性の探索問題に有効
- 遺伝的アルゴリズム、遺伝的プログラミング、
進化的プログラミング、進化戦略を総称して進化的計算と呼ぶ

ニューラルネットワーク(1/2)

- ニューラルネットワークとは神経回路網を意味する言葉
- (脳の) 神経回路の基本機能の一部をモデル化し、工学的に役立てることが狙い
- パターン認識などの分野で利用されることが多い



神経細胞 (ニューロン)

ファジィ論理

- ファジィ論理は人間の知恵をコンピュータで利用するための道具
制御問題に対応
- 制御問題に利用されることが多い
- ファジィ制御は「もし～のときには～のように制御せよ」という形式の、曖昧性を含む言語ルールを用いる
(IF～THENルール)

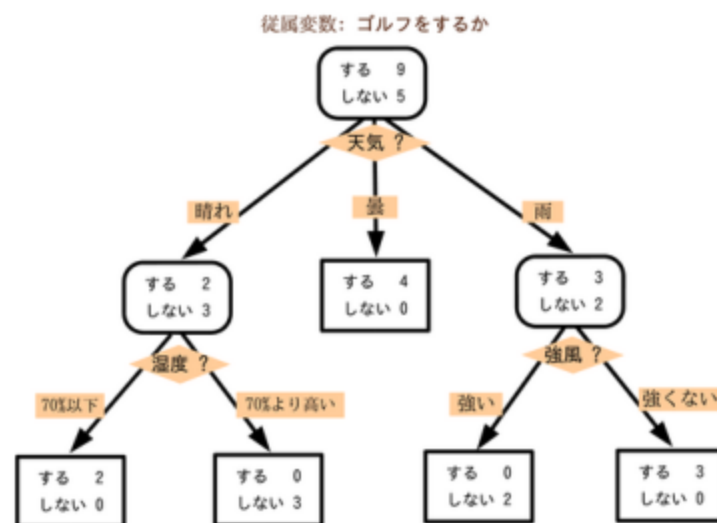
決定木、強化学習

- 決定木

- データにおける規則を木の形で表現
- 教師あり学習の機能を実現

- 強化学習

- 試行錯誤による経験の積み重ねの模倣
- 教師なし学習、探索の機能を実現



決定木の例

ソフトコンピューティングの特徴

- 生物（特に人）が発想の原点にある技術
 - 遺伝的アルゴリズム：生物の進化をモデル化
 - ニューラルネットワーク：生体の神経回路網を模倣
- 生物（特に人）の考え方をモデル化
 - ファジィ理論：人のあいまいな論理を模倣
 - 決定木：人が考えるルールを木構造でモデル化
 - 強化学習：人の試行錯誤プロセスをモデル化

最適化問題(1/2)

- 最適化問題とは、特定の集合上で定義された実数値関数また整数値関数について、その値が最小（もしくは最大）となる状態を解析する問題
（数理計画問題、数理計画も含む）
- 物理学やコンピュータビジョンにおける最適化問題は、考えている関数をモデル化された系のエネルギーを表すものと見なす
（エネルギー最小化問題とも呼ばれる）

最急降下法(4/4)

- 利点

- 実装が容易
- 解空間が単峰性である場合、誤った方向への探索がない

- 欠点

- パラメータに依存
- 解空間が多峰性である場合、局所解に陥りやすい

焼きなまし法(1/5)

- 大域的最適化問題解決のための汎用的な乱択アルゴリズム
 - 広大な探索空間内で、与えられた関数の大域的最適解に対して、よい近似を取得するための手法
-

制約充足問題

- 一つ、もしくは複数の制約条件を満たす解を見つける問題
- 最適化問題とは異なり、制約条件を満たす解であるならば、その解の最適性は問わない
- すべての解の制約充足が困難な場合は、制約違反量もしくは制約違反確率の最小化を図る
- 各制約条件違反それぞれにペナルティ（重み）を課し、ペナルティ合計の最小化を実施する

復習： 最適化問題

$$\begin{array}{ll} \min_x f(x) & \leftarrow \text{目的関数} \\ \text{s.t. } x \in F & \leftarrow \text{制約条件} \\ & F \subseteq X \end{array}$$

- x : 決定変数 (対象問題で決定すべき量)
一般には複数あるので、ベクトルで表現
(そのベクトル空間を X とする)
- $f(x)$: 目的関数 (x の良さ/悪さを与える値)
- F : 可能解領域 (空間 X の中で解が許される範囲)

最適化 = 目的関数の最大化/最小化

組合せ最適化問題の解き方

- 基本的に、列挙する（全ての解を挙げる）ことでしか最適解は得られない



- でも解の数は非常に多い（NP困難な問題）
- 厳密な最適解が必要か？
 - 厳密解を得たい → すべて列挙するしかない

厳密解法 = 効率的な全列挙法：分枝限定法、動的計画法

- 近似的な解（最適に近そうな解）で妥協する

近似解法：部分的列挙法／発見的方法／探索的方法

欲張り法

遺伝的アルゴリズム、
粒子群最適化法

欲張り法

- 貪欲法、グリーディ算法ともいう
- 欲張り法は近似アルゴリズムの一種
- 欲張り法は問題の要素を複数の部分問題に分割し、それぞれを独立に評価し、評価値のもっとも高いものを取り込んで解を得る手法

組合せ最適化問題の典型例

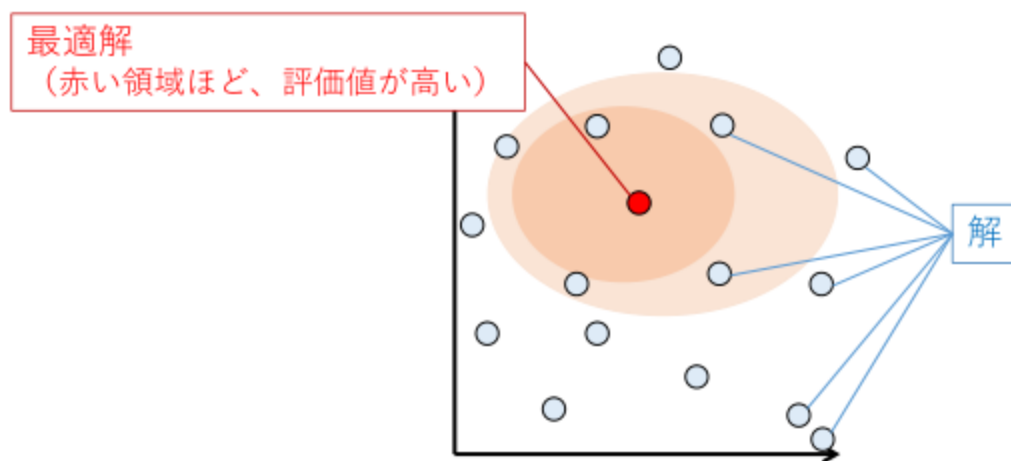
(色々な問題がこれらのいずれかになっていることが多い)

- 巡回セールスマン問題
 - ナップザック問題
 - 1 機械スケジューリング問題
 - 最大充足可能性問題と充足可能性問題
 - 一般化割当問題
 - グラフ彩色問題
 - 整数計画問題
-

粒子群最適化法(2/8)

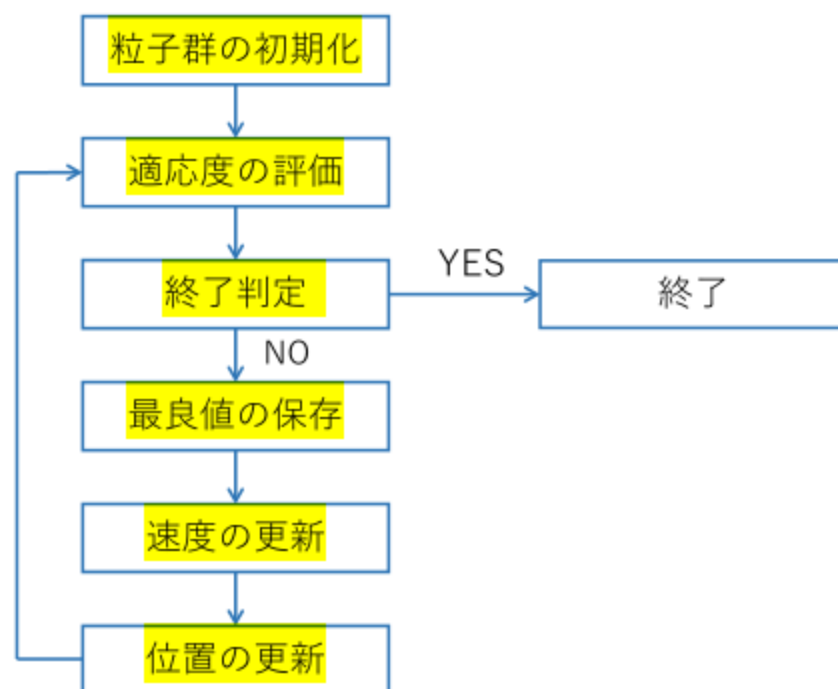
- 群（解集団）に含まれる複数個の探索個体が、群で情報を共有しながら最良値を探索

1. 解空間に群（解集団）をランダムに配置



粒子群最適化法(6/8)

- 群（解集団）に含まれる複数個の探索個体が、群で情報を共有しながら最良値を探索



粒子群最適化法(7/8)

- 群（解集団）に含まれる複数個の探索個体が、群で情報を共有しながら最良値を探索
- 位置と速度を持った粒子群としてモデル化される
位置 x : 解の現在位置（値）、速度 v : 解の移動量
- 各粒子が最良値に移動するため、収束性が高い

PSOの更新式

$$(1) v = wv + c_1 r_1 (G_{best} - x) + c_2 r_2 (P_{best} - x)$$

$$(2) x = x + v$$

w : 慣性定数

v : 探索個体の速度

x : 探索個体の位置

c_1, c_2 : 学習係数

r_1, r_2 : $[0, 1]$ の一様乱数

P_{best} : 各探索個体の最良位置

G_{best} : 個体群全体での最良位置

粒子群最適化法(8/8)

- 利点

- 実装が容易
- 各粒子が最良値に移動するため、収束性が高い
(永遠に探索が続くことはほとんどない)

- 欠点

- 初期解を十分にランダムに配置しなければ局所解に陥る
- 解空間が離散である場合には使用できない

第3回 進化型計算

- 遺伝的アルゴリズム
 - 選択
 - 交叉
 - 突然変異
- 応用例：建築物の構造最適化

遺伝的アルゴリズム (G A) (1/3)

- 生物の進化過程をモデル化した最適化手法の一つ

ダーウィンの進化論

「生物は交叉、突然変異、淘汰を繰り返しながら、環境に適合するように進化していく」



コンピュータ上に仮想生命を生成、かつ、その環境に対する適応度を最適化問題の目的関数に一致させ、進化の過程をシミュレーションすることで最適化問題を解く

1つの仮想生命

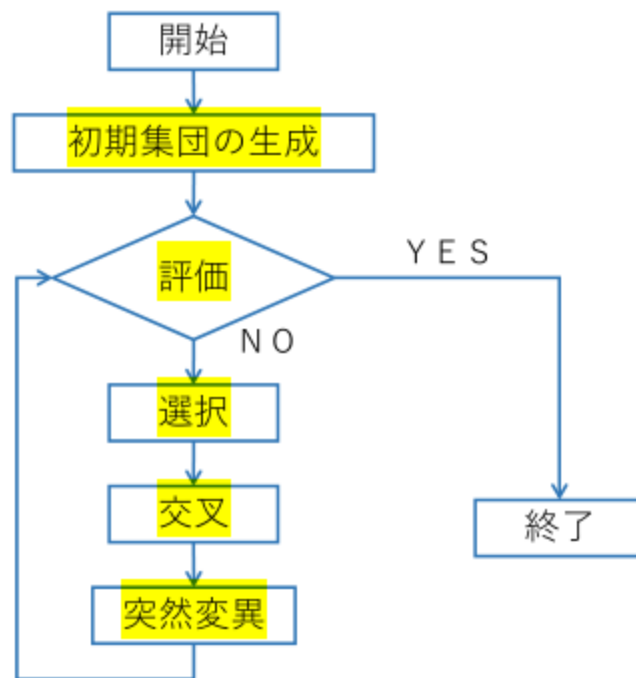
染色体



各遺伝子

遺伝的アルゴリズム（G A）（3/3）

- 生物の進化過程をモデル化した最適化手法の一つ



選択(1/2)

「適応度の高い生命が次の世代により多くの子孫を残す」

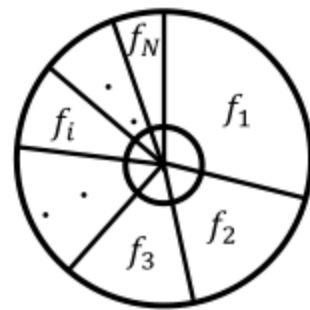
- エリート保存戦略

集団の中で最も適応度の高い生命を無条件でそのまま次世代に残す

- ルーレット選択

各個体の適応度とその統計を求めて、適応度の統計に対する各生命の割合を統計確率として生命を選択する

$$p_i = \mathbf{w} \left(\frac{f_i}{\sum_{k=1}^N f_k} \right)$$



選択(2/2)

- 選択における確率を適応度に比例させるのではなく、適応度の2乗に比例させると、進化の進み方はどう変わるか？
逆に適応度の平方根（ルート）に比例させると、どう変わるか
- 個体群の収束を抑制する、逆に、促進する、には、適応度の差をどのように調整すべきか

交叉(1/2)

- 選ばれた2つの生命の遺伝子をランダムな位置で部分的に入れ替える
- 1点交叉
ランダムに選んだ親2つに対し、ランダムに選んだ交叉点の前後で遺伝子を入れ替える

親	01001 101	→	子供	01001110
	01100 110			01100101

- 2点交叉
2点の交叉位置をランダムに選択し、各部分を入れ替える

親	010 011 01	→	子供	01000101
	011 001 10			01101110

突然変異(1/2)

- 一定の確率で染色体の一部をランダムに操作する

遺伝子型： 10010110
 ↓ 突然変異
 10110110

- 他の遺伝子に置き換えることにより、交叉だけでは生成できない子を生成できるため、集団の多様性を維持する働きがある

3つの遺伝的操作

個体の遺伝子型（記号列）に対してほどこす操作

[収獲 -exploitation-] より優れたものに近づくため

- 選択：評価（適応度）の高い個体をより選ぶ

[探索 -exploration-] まだ調べていない新しいものを試す

- 交叉：2つの個体を組み替えて、新たな個体を作る
- 突然変異：1つの個体を少し改変することで、新たな個体を作る

方針：

新しいものを作って評価し、良ければ選ぶ（自然選択による進化）

いずれも確率的な操作で行う（より良くなる組み換え方、改善の仕方は実は分からずに操作している）

最適化手法（探索）としての G A

- 解空間の探索法としての二大特徴は何か
 - 多点探索であること、確率的な探索であること
- 最適化問題を解くために G A を使おうと考えた際、決めるべき要素は何か
 - 解の表現（遺伝子型の表し方、符号化法）
 - その遺伝子型で交叉を行っても、大丈夫か
 - 交叉で生まれた個体は、制約条件を満たすか
 - 適応度の与え方

情報処理素子としてのニューロン

- 生体ニューロン（神経細胞）
 - 細胞体：信号を処理する細胞本体部
(閾値作用)
 - 樹状突起：入力信号を受け取る部分
 - 軸索：出力信号伝達部
 - シナプス：他ニューロンとの接合部分
(興奮性と抑制性)
- ニューロンのモデル
多入力1出力の情報処理素子

人工ニューロンのモデル

2つの本質的な特徴

- 多入力：他の多数のニューロンから信号を受け取り、膜電位を決定
 - 閾値素子：膜電位が閾値を超えた途端に興奮し、スパイク発火
-

任意 (i 番目) のニューロンのモデル

$$u_i = \sum_{j=1}^n w_{ij} x_j$$

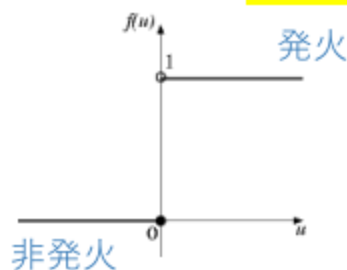
ニューロンの状態（膜電位）は、
複数入力の重み付き和で決定
（ネット値）

$$y_i = f_i(u_i - \theta_i)$$

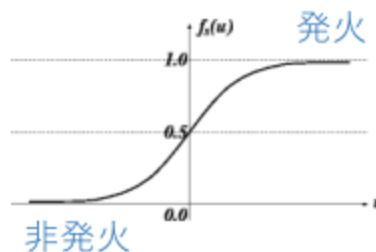
閾値を超えると発火

第 i ニューロンの活性化関数: f_i

e.g. 階段関数、線形関数、シグモイド関数




$$f(x) = \frac{1}{1 + \exp(-x)}$$



しきい素子（階段関数）のモデル

f_i が階段関数のとき

$$y_i = f_i(u_i - \theta_i) = \begin{cases} 1, & \text{if } \sum_{j=1}^n w_{ij}x_j - \theta_i \geq 0 \\ 0, & \text{if } \sum_{j=1}^n w_{ij}x_j - \theta_i < 0 \end{cases}$$


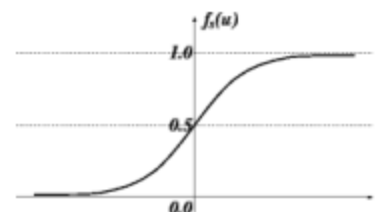
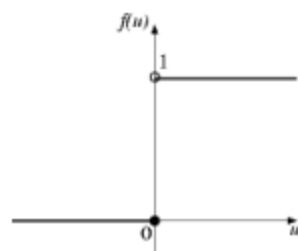
The diagram illustrates the McCulloch-Pitts model. A central equation defines the output y_i as a step function of the net input $u_i - \theta_i$. The function is 1 if the net input is non-negative and 0 otherwise. To the right, a graph of $f(u_i)$ shows a horizontal line at 0 for $u_i < \theta$ and a horizontal line at 1 for $u_i \geq \theta$, with a vertical jump at the threshold θ .

McCulloch-Pittsモデル：生体ニューロンを極めて単純化し、2つの本質的な特徴をとりいれた

- ニューロンは、他の多数のニューロンからの信号を受け取り、その膜電位（内部状態）が定まる
- ニューロンは、その膜電位が閾値を超えなければ何もせず、超えれば興奮（発火）する

しきい素子（階段関数）のモデル

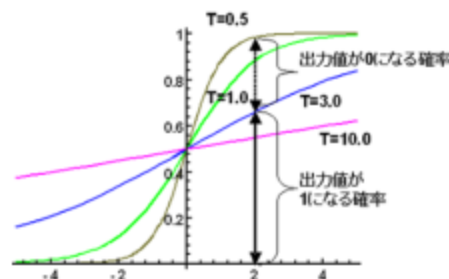
Threshold



階段関数： $T \rightarrow 0$ の極限

シグモイド（S字型）関数

$$f(x) = \frac{1}{1 + \exp(-x/T)}$$



T 値を変えると、階段関数とS字型関数は、ほぼ一樣な関数に相当

単純パーセプトロンの学習

誤り訂正学習法（教師あり学習）：

出力値と教師信号が一致しない「誤り」の場合のみ、
結合荷重と閾値が修正される

修正量

$$\Delta \mathbf{w} = \eta \cdot (t - o) \cdot \mathbf{x}$$

η ：学習率
(小さな正值)

\mathbf{w} : A層とR層の間の結合荷重と閾値

\mathbf{x} : A層の n 個のユニットの出力値、0,1の二値

o : R層のユニットの出力値、0,1の二値

t : 教師信号、0,1の二値

$$\text{修正後の}\mathbf{w} = \text{修正前の}\mathbf{w} + \text{修正量}\Delta \mathbf{w}$$

デルタ則 (Widrow-Hoff則)

別称： Δ ルール (教師あり学習)

連続値をとるA層とR層の2層からなるネットワークに、
誤り訂正学習法を拡張して適用

$$\Delta w_{ji} = \eta \cdot (t_j - o_j) \cdot o_i$$

η : 学習率
(小さな正值)

w_{ji} : A層の第*i*ユニットからR層間の第*j*ユニットへの結合荷重と閾値

o_i : A層の第*i*ユニットの出力値、連続値 ($i = 1, \dots, n$)

o_j : R層の第*j*ユニットの出力値、連続値 ($j = 1, \dots, m$)

t_j : R層の第*j*ユニットに対する教師信号、連続値

デルタ則は最急降下法

最急降下法：関数 $E(\mathbf{w})$ の最小点（実際は極小点）を探す方法

$$\Delta \mathbf{w} = -\eta \cdot \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} \propto -\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}}$$

ここで、 A 層- R 層の2層のネットワークに対する出力誤差の2乗和

誤差（の総和）

$$E(\mathbf{w}) = \frac{1}{2} \sum_{j=1}^m (t_j - o_j)^2$$

を考えると、デルタ則は最急降下法となっている

- j を出力層ユニットとしたとき

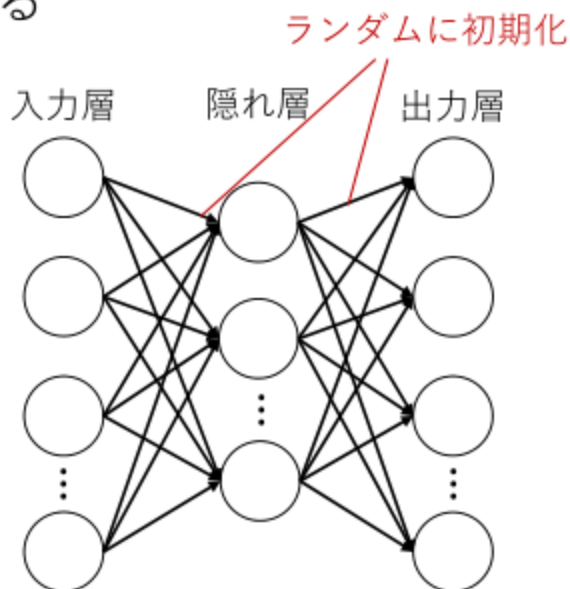
$$\delta_j = (t_j - o_j) \cdot f'(u_j)$$

- j が隠れ層ユニットのとき

$$\delta_j = f'(u_j) \cdot \sum_k \delta_k \cdot w_{kj}$$

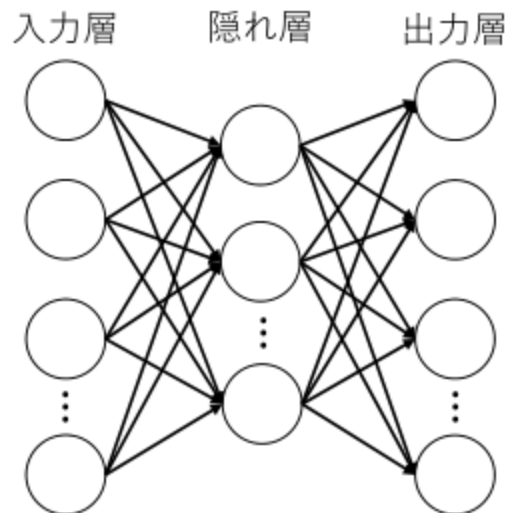
誤差逆伝搬法 (Back Propagation法) (2/6)

2. 入力層、中間層、出力層の順に、各ユニットの入出力を計算する
この操作は信号伝送過程、あるいは、誤差逆伝搬法における
前進型処理とよばれる



誤差逆伝搬法 (Back Propagation法) (4/6)

4. 得られた誤差を最小化するように、出力層から入力層に向かって、ネットワークの各層間の結合荷重を修正する
この操作は学習過程、あるいは、誤差逆伝搬法における後進型処理とよばれる



$$E(\mathbf{w})$$

畳み込みニューラルネットワーク

- 畳み込みニューラルネット (Convolutional Neural Network, CNN) は、畳み込み層とプーリング層という特別な2種類の層を含む順伝搬型ネットワークで、画像認識・[クラス分類](#)に応用される
- 通常の順伝搬型ネットワークと同様に、誤差逆伝搬法を用いて学習する
- 局所受容野および重み共有と呼ばれる特別な層間結合を持つことがこれまでのネットワークとの違い

畳み込み層(1/7)

- 畳み込みの演算を行う単層ネットワーク

画像

1	2	2	4	3	4	5	4	5	5
1	2	3	4	3	4	5	5	4	6
2	2	2	3	3	4	4	5	6	6
3	3	3	4	4	5	6	5	6	7
4	4	4	3	4	6	7	8	7	8
3	3	3	4	4	5	6	8	7	8
4	3	5	4	6	7	7	8	8	9
4	5	6	5	7	6	7	9	8	9

フィルタ($w=1$)

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

積和

結果画像

プーリング層(1/4)

- 局所における最大値を抽出 (MAXプーリング)

特徴マップ

	2	3	4	3	4	5	5	4	
	2	2	3	3	4	4	5	6	
	3	3	4	4	5	6	5	6	
	4	4	3	4	6	7	8	7	
	3	3	4	4	5	6	8	7	
	3	5	4	6	7	7	8	8	

プーリング結果

3	4						

局所コントラスト正規化(2/2)

- 減算正規化

画素 (i, j) を中心とする $H \times H$ の正方領域における平均値を $\overline{x_{ij}}$ とし、各画素値 x_{ij} から差し引く

$$z_{ij} = x_{ij} - \overline{x_{ij}}$$

- 除算正規化

各領域内での画素の分散を σ_{ij}^2 とし、減算正規化ののちに各領域内での分散を揃える

$$z_{ij} = \frac{x_{ij} - \overline{x_{ij}}}{\sigma_{ij}}$$

ソフトマックス関数(2/2)

- ニューラルネットワークによる多クラス分類に使用される
- 出力層の k 番目のユニットにおける総入力 u_k をもとに出力層の全ユニットの合計出力が1となるよう値を調整

$$y_k \equiv \frac{\exp(u_k)}{\sum_{j=1}^K \exp(u_j)}$$

ファジィ論理(1/3)

- あいまい性についての数学的な理論

ex. 「非常に背が高い」

- 主観の科学的利用 ⇔ 客観
- 言語で表現された知識の利用 ⇔ 数値

- 提唱者：ザデー（1965 カリフォルニア大）
- 応用：日本で開花（1987 家電、地下鉄）

ファジィ制御

ファジィ・エキスパートシステム

が、代表的

様々な学習

- ニューラルネットワーク

非線形関数 $y = f(x; w)$ の数値パラメータ w の値を調整

関数近似

- 強化学習

状態 s で行動 a をとるときの報酬の推定値 $Q(s, a)$ を調整

非線形方程式

数値の学習

- 決定木

属性値から概念を識別するための木構造を生成

- 説明に基づく学習

将来の探索を削減するための効率良いルールを生成

- 帰納論理プログラミング

与えられた入出力関係を満たす数学的機能パターンを生成

構造の学習

決定木の学習(2/2)

- 決定木の学習は、具体的な判断事例から決定木を生成すること

事例

入力 = {属性1=値1, ..., 属性n=値n} : 出力 = 判別結果のような形式のデータで、入力の属性値に応じた判別結果を指定

- このようなデータの集まりを、学習アルゴリズムに対する訓練例という

複数の決定木(2/2)

- 属性をどのような順番でテストするかによって、生成される決定木が異なる
- すべての訓練例に矛盾のない決定木の中でもっともサイズの小さなものを求める（オッカムのかみそり）
しかし、これには膨大な計算量が必要となる



- 単純なヒューリスティックにより最小でなくともそれに近い決定木の取得法が提案されている

情報理論の利用(3/3)

平均情報量(エントロピー)

$$I(P_1, \dots, P_n) = \sum_{i=1}^n P_i \log_2 \frac{1}{P_i} = - \sum_{i=1}^n P_i \log_2 P_i \quad (\text{ビット})$$

例: コイン投げ

表 $P_1 = 1/2$

裏 $P_2 = 1/2$

$$I\left(\frac{1}{2}, \frac{1}{2}\right) = \frac{1}{2} \log_2 2 + \frac{1}{2} \log_2 2 = 1$$

◆ 正例が p 個, 負例が n 個のときの平均情報量

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = \frac{p}{p+n} \log_2 \frac{p+n}{p} + \frac{n}{p+n} \log_2 \frac{p+n}{n}$$

ID3アルゴリズム

- これまでのルールをまとめて

決定木 $ID3(S, A, \text{default})$ {

S : 訓練例の集合 A : 属性の集合
 default : Yes / No の既定値

if (S が空集合) return default
else if (S がすべて正例) return Yes
else if (S がすべて負例) return No
else if (A が空集合) return 多数決(S)
else {
 $\text{bestA} = A$ のうちでゲインが最大の属性;
 $\text{tree} = \text{new 決定木}(\text{bestA})$;
 $\text{bestAdomain} = \text{bestA}$ の取りうるすべての値;
 for each v in bestAdomain do {
 $S' = S$ のうち $\text{bestA}=v$ となっている全データ;
 $\text{subtree} = ID3(S', A - \text{bestA}, \text{多数決}(S))$;
 tree の下に subtree を v の枝で連結する;
 }
 return tree ;
}

ノイズと過剰一致(1/2)

- 訓練例にはノイズと呼ばれる誤りの一種が含まれる
- ノイズまでも再現するような細かな分類は不適切な決定木を生み出すことが多い (過学習)



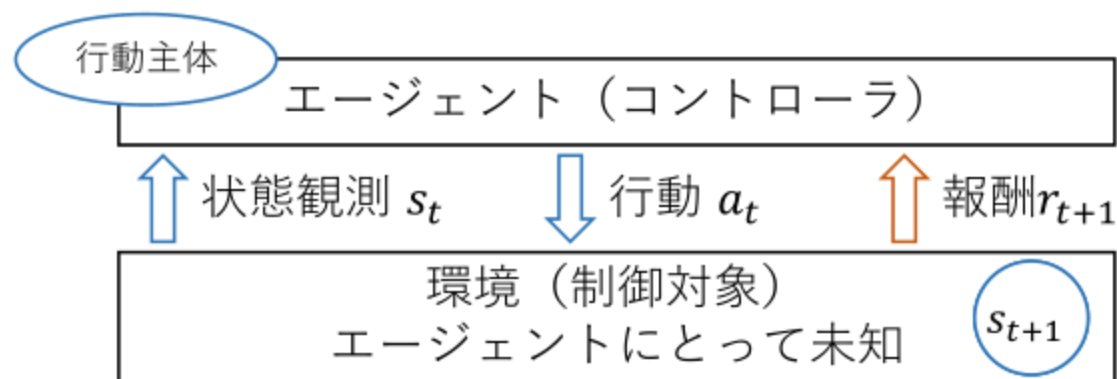
- 平均情報量のゲインが十分小さいときにはそれ以上の分類をやめ、終端ノードとする (枝刈り)

アルゴリズムの性能評価(1/2)

1. 例となるデータを多数収集
 2. 例となるデータを訓練集合と検査集合に分割
 3. 訓練集合を学習アルゴリズムに入力して決定木を生成
 4. 検査集合を決定木に入力し、正答率を取得
- 決定木に限らず、学習アルゴリズムで一般的に使用されている

強化学習とは(1/5)

- 試行錯誤を通じて環境に適応する学習制御の枠組み
- 生体の「脳」のシステムを模倣



- 教師付き学習とは異なり、状態入力に対する正しい行動出力を明示的に示す教師が存在しない

マルコフ決定過程(3/4)

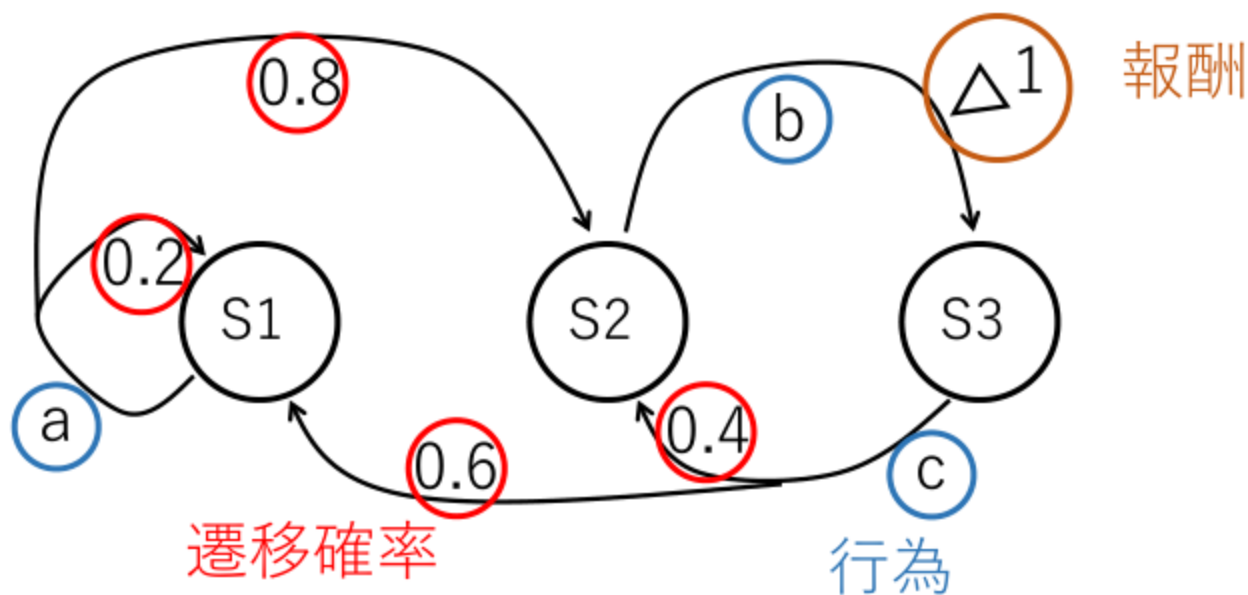
- エージェントの各時刻における意志決定は、
政策関数 $\pi(s, a) = P_r\{a_t = a | s_t = s\}$ (ただし全状態 s 、全行動 a において定義される) によって表される (政策 π)

2つの性質

- マルコフ性: 状態 s' への遷移が、そのときの状態 s と行動 a にのみ依存し、それ以前の状態や行動には関連しない
- エルゴート性: 任意の状態 s からスタートし、無限時間経過した後の状態分布確率は 最初の状態とは無関係

マルコフ決定過程(4/4)

- 状態遷移確率は現在の状態のみに依存する
- 状態遷移確率は時間的に変動しない



行動はup,down,left,right

S 1	2	3	4	5
6	7	8	9	G 1 0

$$Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a')]$$

$$r = 1 \text{ 或 } 0 \quad \alpha = 0.5 \quad \gamma = 0.1$$