

脳が行なっているような画像処理 AI を作ってみよう

電気通信大学 大学院情報理工学研究科
庄野 逸

概要

この講義では、脳のような AI を作ってみるということで、視覚機能を模倣したモデルを計算機の上でつくってみます。ヒトの脳はおよそ 1000 億個のニューロンと呼ばれる計算素子から構成されています。本テーマでは、まず古代生物の視覚構造を模倣した側抑制モデルを構成し、画像処理を行ってみます。その後に、側抑制効果を一般化したフィルタ演算を模したモデルを構築し、パターン認識実験を行ってみます。

1 はじめに

ヒトの脳はおよそ 1000 億個のニューロンと呼ばれる計算素子から構成されています。そのうちのおおよそ半分は、視覚に関する計算を行っていると言われています。本テーマでは、視覚のモデルを通して、我々の視覚野がどのような計算を行っているかを考え、脳を模倣して数字画像を認識するような AI を作ることを目的としています。これを段階的に行っていくものとして、

- 古代生物の視覚の側抑制モデルを実現してみる
- 側抑制モデルを一般化した畳み込み計算を実現してみる
- 現代 AI の中核をなす畳み込みネットワークを実現してみる

といったサブテーマにそって実験を勧めていきましょう。ここではおおまかな時間として以下のようない流れを考えています。

13:00-13:10 開会式
 13:10-13:25 計算機室へ移動 & 自己紹介
 13:25-13:50 計算機使用方法の説明
 13:50-14:50 1 時間目:古代生物の視覚 —側抑制モデル—
 15:00-16:00 2 時間目:側抑制の一般化 —フィルタ計算の概要—
 16:10-17:10 3 時間目:現代 AI の中核技術 —畳み込みネットワーク—
 17:20-17:30 閉会式

2 実験用計算機へのログインとノートブックの使用方法

プログラミング言語としては Python をもちいており、配布資料は、jupyter notebook と呼ばれるものを用います。これは、当日の配布資料で。

3 1 時間目:古代生物の視覚 —側抑制モデル—

最初のサブテーマでは視覚が発生したころの計算機シミュレーションを行ってみましょう。5 億年ほど前の生物である三葉虫などは複眼を持ち、光学情報をもちいることで生存性を高めていたと言われています。この複眼システムは、現代でもカブトガニの一種に残っています。この複眼システムでは、光を感じる細胞群が並んでおり、各細胞

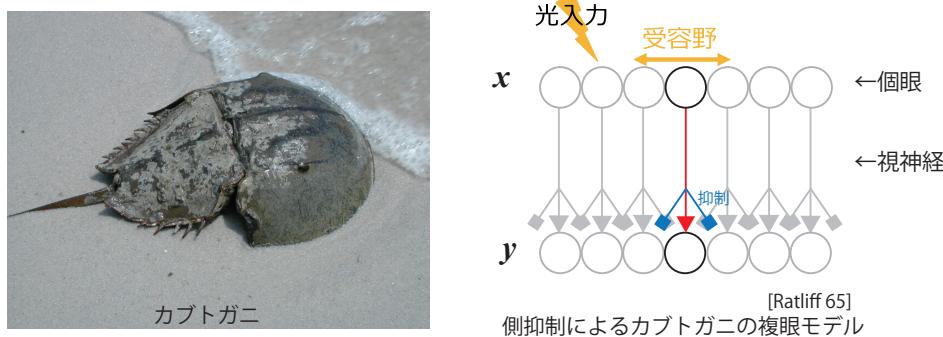


図 1 カブトガニの視細胞の概要。並んで視細胞が近傍の視細胞に抑制をかける

は、自分の周辺の細胞に対し、出力を抑制させる効果をもたらす。このような効果は“側抑制”と呼ばれます。この構造は丁度、図 1 のような形に視細胞が並んでいて、互いが互いに抑制をかけるという構造になっています。それでは、側抑制は視覚において、どのような役割を持っているのでしょうか？

ここでは簡単のために複眼システムを 1 次元に並べたような状況を考えてシミュレーションを行ってみます。実際の計算は配布資料の中を覗いてみてください。このような計算機実験を行う場合、個眼の一個一個の出力は、“配列”と呼ばれる構造に保存するのが一般的です。配列は添字で複数のデータを表現する仕組みです。入力は各個眼に入ってくる光信号で、場所によって値が異なるはずなので、場所を n という添字で表すこととします、ここでは入力の光の分布を $\mathbf{x} = \{x_n\}$ という配列で表すものとします。一方、出力は側抑制を受けたもので $\mathbf{y} = \{y_n\}$ として表すものとします。側抑制の計算方法は比較的簡単で、例えば $n = 3$ 番目の位置の出力 y_3 を求めたければ

$$y_3 = (-1) \times x_2 + (2) \times x_3 + (-1) \times x_4 \quad (1)$$

といった形で求めることになります。ここでは括弧の中にかかっている係数が入力から出力を伝える側への重みを表していて、負の値は抑制効果を表すことになります。すなわち、出力と同じ位置 $n = 3$ の入力 x_3 は、少し顎眞（係数が 2）して大きく見て、周囲の入力 (x_2, x_4) はやや割り引いた抑制効果（係数が -1）をもたらしていることになります。なお係数の値そのものは、適当な例なので、あとで適宜いじってどのような出力が得られるかを実験してみてください。

また、これは $n = 3$ の場合だけでなく、他の場所でも

$$y_n = (-1) \times x_{n-1} + (2) \times x_n + (-1) \times x_{n+1} \quad (2)$$

とかけることになります。

それでは、この出力 y_n は、入力に応じてどのような性質を持つのでしょうか？まず、 x_n が一様で一定の入力だった場合を考えてみます。この場合、周辺から入ってくる抑制効果と、顎眞にみている個眼の効果が相殺して、値は常に 0 になります。一方、値が変化している場所では、 x_n と周辺入力 x_{n-1} は異なる値となるため、差が生じ、0 でない値が発生することとなります。すなわち、この複眼システムにどんなに強い光を当てても、位置的に光の強度が変化していかなければ、このシステムは反応を示さず、光の強度が変化している場合のみ反応を示すということになります。すなわち、外界において光の強度“変化”が起こるような場合を検知するためのシステムと考えることができます。これは外敵が物理的に近づいてきた場合や、捕食するための獲物が近づいた場合に反応をしていることを示していると考えられており、このような視覚システムが生物に装備されたことによって生存性が上がったと言われています。

我々の視覚システムは、これほど単純な側抑制のシステムから成り立っているわけではありませんが、この側抑制で我々の知覚と合致するマッハバンド効果を説明することができます。マッハバンドは濃淡の微妙に異なるグレーの領域が接しているときに暗い方の境界部分はより暗く、明るい方の境界部分はより明るく見えるような錯視現象です。図 2 にマッハバンドの概要を示します。これは黒から白に徐々に変化していく画像ですが、黒から白に変わ

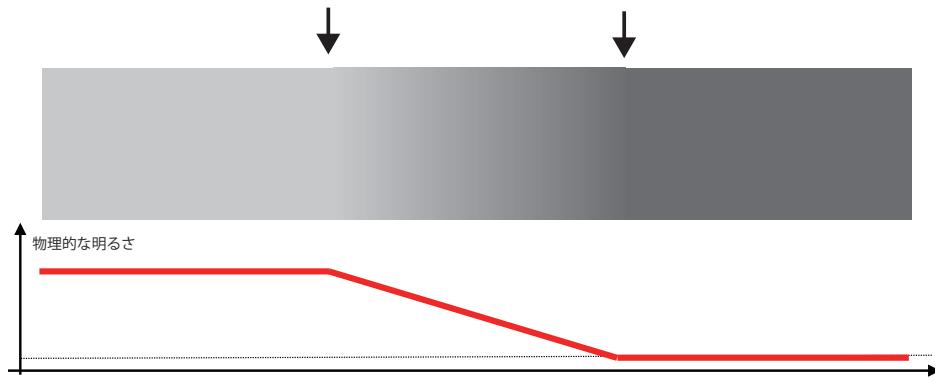


図 2 マッハバンドの概要

る場所で黒く見えたり白く見えたりするのが（見える人には）見えるかと思います。これは、光の強度を、1次元の入力で表すと図 2(b) のようになります。これに $\{(-1), (3), (-1)\}$ といった側抑制をかけた場合、変化する場所が強調されることになります。

課題 1

配布した計算機試料に対して側抑制効果を適用してみましょう。まずは、上の式で示した $(-1), (2), (-1)$ という係数の組み合わせで実験を行い、次に、 $(-1), (3), (-1)$ となるような組を考えてみてください。他にもいろいろ試してみましょう。

4 2 時間目: 側抑制の一般化 — フィルタ計算の概要 —

次に、式(2)で表されるような側抑制モデルの数理的、工学的な側面を考えてみたいと思います。式(2)では、係数の組 $\{-1, 2, -1\}$ が位置をずらしながら重み付きの和を計算していたことになります。これは、小難しい言葉でいえば空間並進対称性があるといいます。このような計算方法は、音声や画像を扱う場合によく現れます。画像の場合は、入力が 2 次元係数の組が 2 次元状の配置になります。すなわち図 3 のような形の計算になります。このような係数を画像を移動させながら施していくことになります。この操作は“畳み込み演算 (convolution)” や、“フィルタ演算 (filtering)” という名前で呼ばれる計算になります。前述の重み係数は、“畳み込み核 (convolution

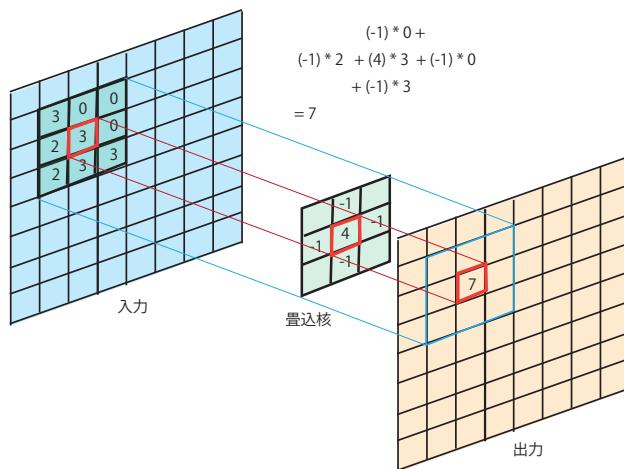


図 3 2 次元のフィルタ演算概要

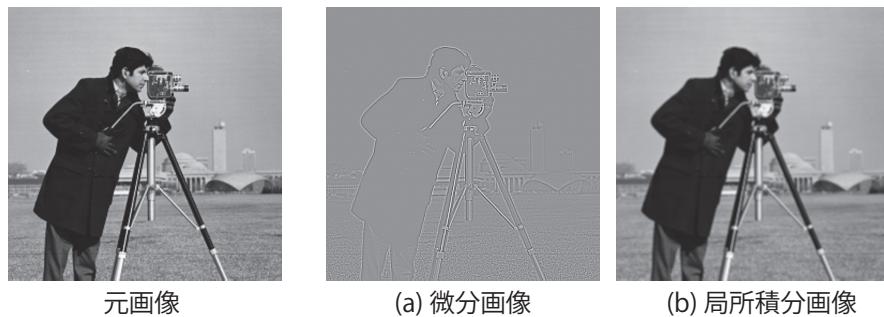


図 4 画像フィルタの適用例: (a) 微分フィルタ, (b) 積分フィルタ

kernel)" や、単に "フィルタ" などといった名前で呼ばれることになります。実際に、この計算は次元が増えると大変になっていくのですが、みなさんが使っている Python などでは数値科学計算用の道具(パッケージライブラリ)が整備されており、"convolve" などといった処理が使えますので、わざわざ陽に計算する必要がありません。パッケージライブラリを用いたほうが、高速で信頼性の高い結果がでますので、むしろ積極的に用いたほうがお得です。

それでは実際に画像にフィルタをかけてみましょう。前述では、 $\begin{array}{|c|c|c|} \hline -1 & +2 & -1 \\ \hline \end{array}$ といった形だったのに対して、

この場合は、 $\begin{array}{|c|c|c|} \hline 0 & -1 & 0 \\ \hline -1 & +4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$ といった形のフィルタになります。すなわち横方向のフィルタと縦方向のフィルタを重ねなわせたような形です。

このような形のフィルタは、側抑制の拡張なので、画像中の“変化”を抽出するようなフィルタになり、“微分フィルタ”などと呼ばれます。このフィルタをかけると図 4(a)のような図が得られることになります。図全体はグレーが主体で、白から黒に変わった部分が画像の変化を表すような場所になります。この結果から解釈すると自然な画像とは、割と多くの領域がグレーで変化の少ない領域と、変化の著しい領域から成り立っていることがわかります。この変化の激しい領域を“エッジ”と呼んだりします。

それでは、フィルタの係数を全部 $1/9$ にして $\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$ といったフィルタを考えてみましょう。このフィルタは、微分形のように周辺との差分を考えるのではなく、周りの意見を容れて考えるようなフィルタになります。

このようなフィルタは“積分形”フィルタと呼ばれます。積分形フィルタの特徴は、画像に含まれる微小な変動をぼかすような効果を持ちます。丁度、図 4(b) に示すような図が積分フィルタの出力となります。

課題 2

微分形フィルタと積分形フィルタを画像に対して適用してみましょう。上ではフィルタを 3×3 の大きさで設定していますが、もっと大きなフィルタをかけた場合や、複数回フィルタをかけた場合に、どのような結果が得られるかを試してみてください。

さらに、フィルタをとったあとに、絶対値をとったり (abs), 0 以下の値を切り捨てたりするような変換 (Rectified Linear) を行うことで、どのような画像が得られるかを確認してください。

5 3 時間目: 現代 AI の中核技術 —畳み込みネットワーク—

最後の時間は画像における現代 AI の中核技術である畳み込みニューラルネットワーク (Convolutional Neural Network: CNN) を作ってみましょう。なにか難しそうな名前ですが、これは複数の特徴抽出フィルタと積分フィルタを組み合わせた構造を単位として持っています。

まず、我々の視覚の概要を説明します。図 5 を参照してください。サルやヒトの視覚野において、モノの形状の

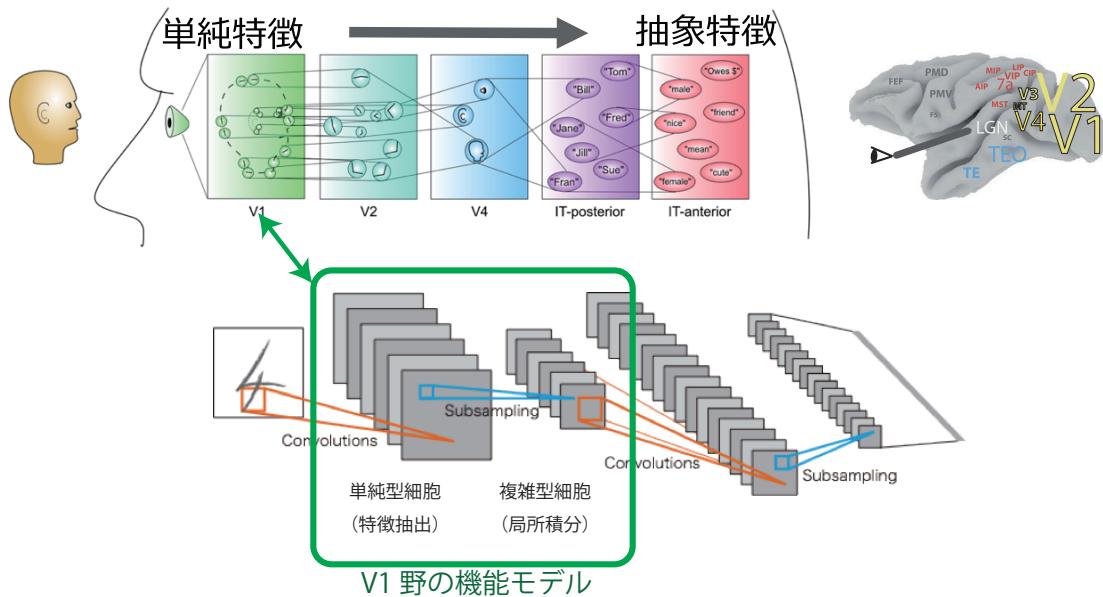


図 5 視覚関連領域の概要: 腹側経路の処理に関して

視覚情報は側頭部に沿って処理されています。眼球からの信号は、まず後頭部に伝達します。この領域は初期視覚(V1)野と呼ばれる領域です。ここで、物体の運動成分と形状成分の情報処理が分岐し、V2野、V4野から、側頭のIT野に到達します。V1野の細胞は画像中のエッジや線分といった構造に強く反応し、IT野の細胞はヒトの顔といった複雑な構造に反応することが知られています。一方ネットワークの構造としては、V1野のネットワークの構造が比較的解明されています。V1野の細胞は、おもに2種類の細胞から構成され、単純型細胞と呼ばれる複雑型細胞と呼ばれる細胞が存在することが知られています。単純型細胞は微分のような特徴抽出を行うような機能を司り、複雑型細胞は抽出された特徴をぼかす(空間プーリングと呼ばれます)ような積分形フィルタで記述できます。

V2野以降のネットワーク構造は、よくわからないのですが、同じ脳の構造が成り立っているのであれば、V1野のような構造を外挿してもよからうというコンセプトで畳み込みネットワークはできています。畳み込みネットワークの構造は、図5の下側が畳み込みネットワークを表しています。まず、畳み込みネットワークの基本構造は、入力から画像特徴を抽出するフィルタを複数枚ならべた構造をもってきます。これらが単純型細胞群にあたります。次に抽出された画像特徴を積分形フィルタでぼかすような操作を行います。これらが複雑型細胞群に対応します。特徴抽出と空間プーリングをフィルタ群と非線形変調を行う、2つの操作を基本構造として、積み重ねたものが畳み込みネットワークと呼ばれるものになります。畳み込みネットワークの基本構造は、サルやヒトの初期視覚野の構造を模倣したモデルです。

前述の特徴抽出層では、たくさんフィルタがあるので設計は面倒なのですが、フィルタの重み係数をデータから決めることがあります。このデータから機械内部のパラメータを定める行為を“学習”と呼びます。学習方法についての詳細は、このテーマの範囲を逸脱することになるので、概略だけ説明します。図5に、学習の概略図を示します。ここでいう学習とは、学習データに正解ラベルがついているようなケースです。たとえば、画像データセット一つ一つに ‘0’ ~ ‘9’ の正解ラベルがついているとしましょう。 p 番目の正解ラベルを t_p として表したとします。一方、学習機械は画像は、なんらかの入力 \mathbf{x} を読み込んで、答え $y(\mathbf{x}; \mathbf{w})$ を返すものとします。 \mathbf{w} は、学習機械内部のパラメータで、フィルタの係数の集まりみたいなものだと考えればよいです。いま、学習機械に期待することは、正解ラベル t_p を先生として、学習機械の答え $y(\mathbf{x}_p; \mathbf{w})$ をできるだけ似せるようにすることです。例えば $\|t_p - y(\mathbf{x}_p; \mathbf{w})\|^2$ が小さくなるような \mathbf{w} を見つけることが課題になります。これが、できるだけ多くのパターンで成り立っていてほしいのです。

$$E(\mathbf{w}) = \sum_{p=1}^P \|t_p - y(\mathbf{x}_p; \mathbf{w})\|^2$$

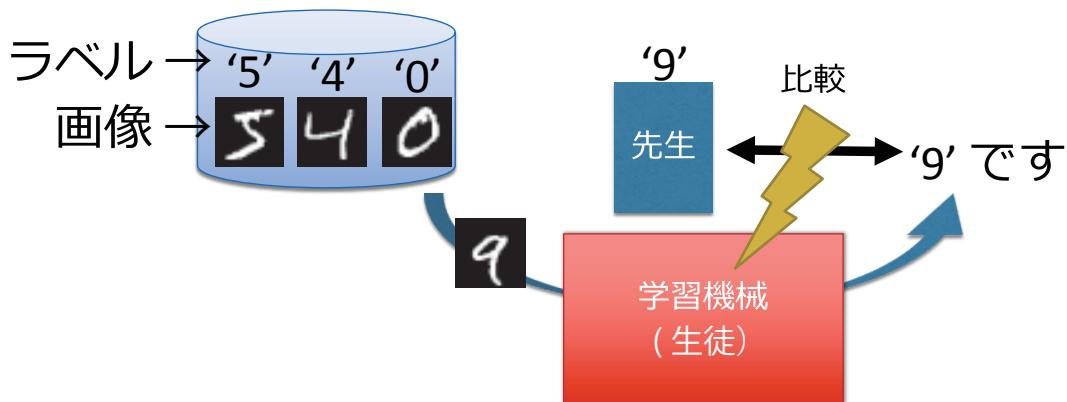


図 6 学習の概要

のような量が小さくなるような w を見つけ出すことができれば良いというのが骨子となります。（具体的には、もうすこし違った関数になりますが、二乗誤差が説明しやすいので、そのように説明しています）

実装には Keras と呼ばれる深層学習用の枠組みを用います。対象は、数字文字データの識別実験になります。数字は '0' ~ '9' までの文字からなるので 10 種類の分類問題を考えることになります。すでに資料の中には、記載していますが、大まかに言えば

- model を連続的なシーケンス (`Sequential()`) として定義し
- 入力側から処理する階層を加える。ここでは `Conv2D()`, `MaxPooling2D()`, `Dense()` など
- 識別を行うために 10 個のカテゴリに分類

という枠組みを最初に記述し、これに対して学習することでフィルタの係数を決めることがあります。