

有名論文から学ぶディープラーニング

講師役：近棟 稔

2016/3/25

ウルシステムズ株式会社

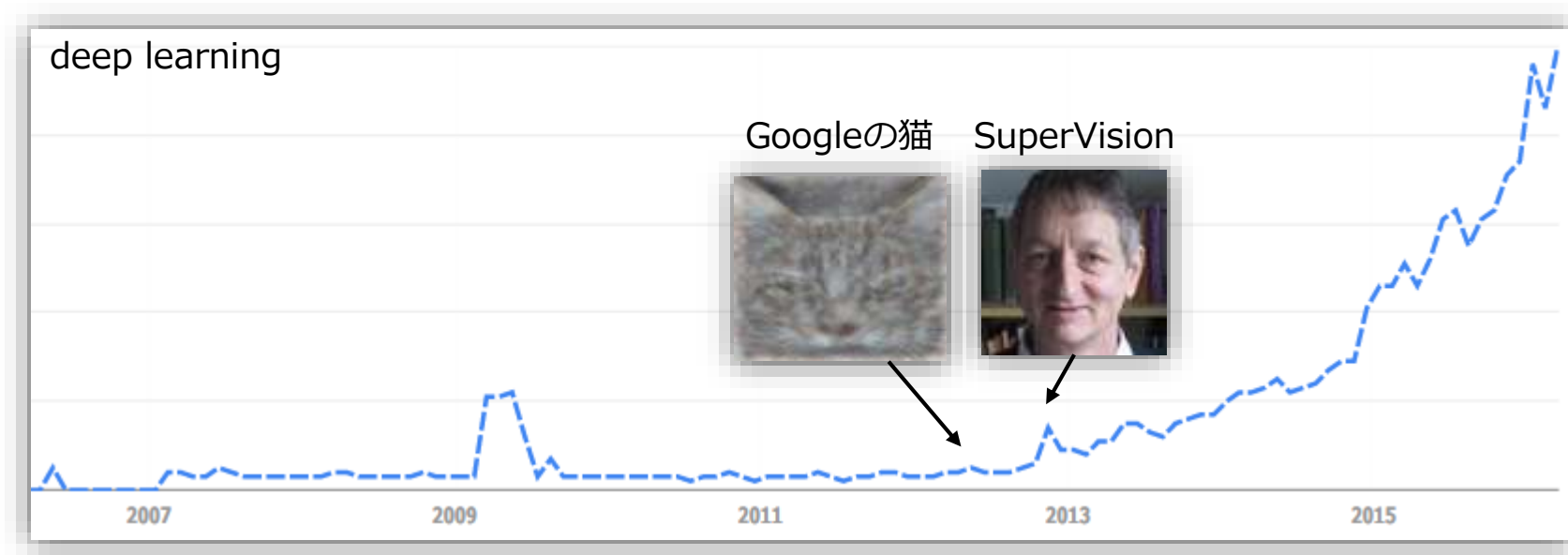
<http://www.ulsystems.co.jp>

<mailto:info@ulsystems.co.jp>

Tel: 03-6220-1420 Fax: 03-6220-1402

はじめに

- 近年、機械学習やAIの最先端技術としてディープラーニング（深層学習）は非常に注目を集めています。ディープラーニングが注目を集め始めたのはGoogle Trendsによると以下のように2012年末になります。世の中のイベント的には2012年に各種機械学習コンテストでディープラーニングを採用したチームが連勝し始めた事がきっかけとなりました。



- 今となっては、Google音声認識で人の声をテキスト化する際に用いられていますし、高精度な物体認識を行いたい場合などに欠かせない技術となっています。
- この勉強会では大きなニュースとして扱われたディープラーニングの有名論文を引用しながら、ディープラーニングの内部を紐解いていきたいと思います。

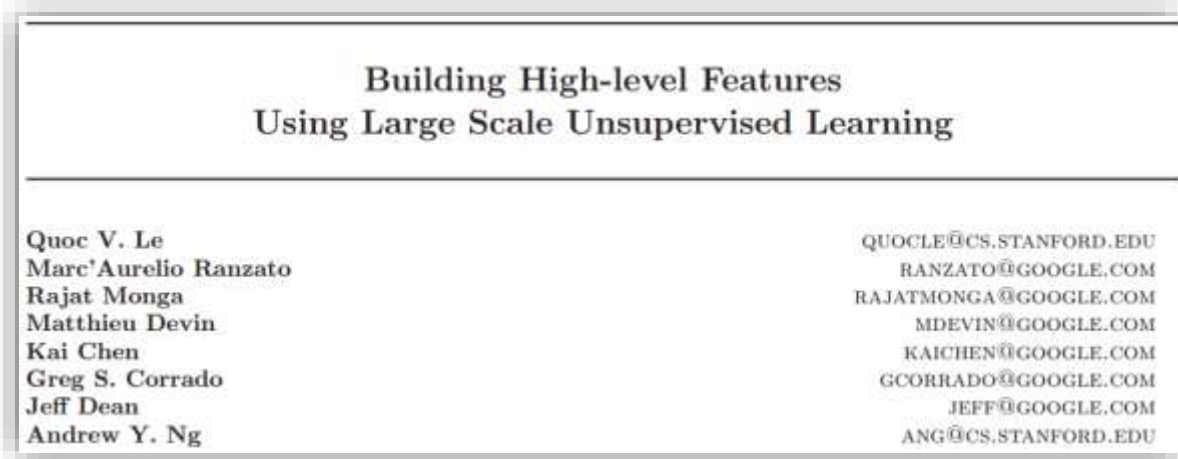
ニュースになったディープラーニング の成果を振り返る

ニュースになった論文：2012年

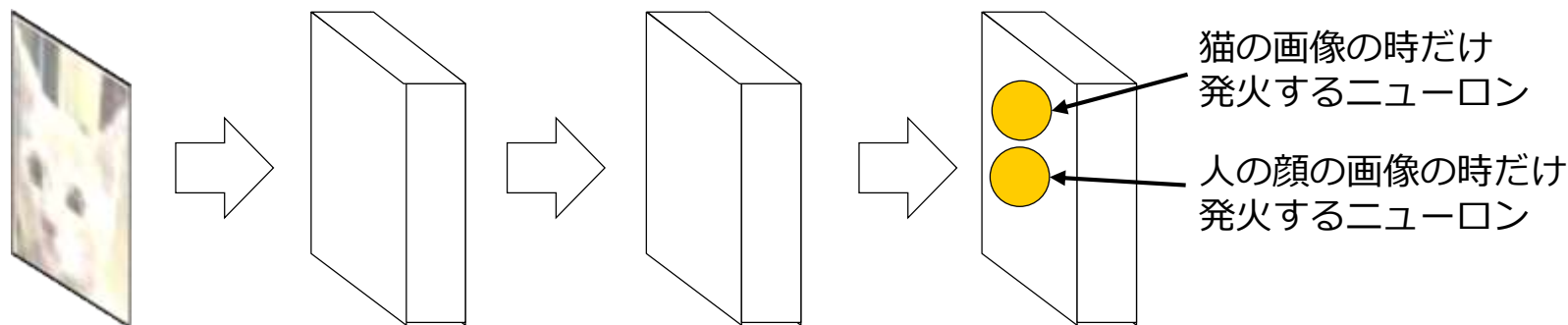
Googleのコンピュータが自動的に猫を認識できるようになった

UL Systems, Inc.

- 「どの写真が猫だよ」と教えなくても、Youtubeからランダムに切り出した大量の画像から「教師なし学習」によって人の顔や猫の顔に反応するニューラルネットワークを自己形成出来たというニュースです。



- 具体的には、1000万枚の200x200ピクセルのカラー画像をYouTubeからランダムに取り出し、トレーニングデータとして利用したようです。トレーニングデータでは、顔の位置や拡大率は全く適当で、顔がフレームから切れていたりします（上記論文に画像が添付されています）。また、どの画像が猫の顔であるかなどの情報は与えていません。このデータでニューラルネットワークをトレーニングすると、猫の画像や人の顔に対して特定のニューロンが発火するように出来たようです。使ったマシンは1,000台(16,000コア)で、3日間学習したそうです。なお、ネットワークの深さは9層です。



- Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) というコンペティションにて、Geoffrey Hintonチーム（カナダのトロント大学）が開発したSuperVisionが2位以下に大差をつけて勝利しました。

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky University of Toronto kriz@cs.utoronto.ca	Ilya Sutskever University of Toronto ilya@cs.utoronto.ca	Geoffrey E. Hinton University of Toronto hinton@cs.utoronto.ca
--	---	---

- この論文の骨子としては、以下のものが含まれています。
 - 使ったのはDeep Convolutional Neural Networkで、画像のRGBピクセル値を直接学習しました。
 - 5つのコンボリューション層とmax-pooling層を持ち、softmaxを用いています。
 - NVIDIAのGPUを2つ使い、1週間学習させています。
 - activation functionとしてrectified linear unitを用いています。
 - overfittingを避けるために、dropoutを用いています。
 - autoencoderなどのpretrainingは、あえて採用しなかったそうです。採用すれば助けになりそうな感触はあったものの、あえて使わず、それでも高い認識能力が得られたようです。

ニュースになった論文：2015年 LSTMで上手な受け答えが可能なチャットボットが作れた

UL Systems, Inc.

- LSTMというニューラルネットワークの一種を用い、既存の人工無能よりもちゃんとした会話を成立可能なチャットボットが作れたという物です。論文にはチャットの例が沢山掲載されています。
- チャットボットの学習のために使ったデータは、ITヘルプデスクのトラブルシューティングデータセットと、映画の字幕データセットの2種類のようなです。データによって受け答えの内容が違います。

A Neural Conversational Model

Oriol Vinyals

Google

VINYALS@GOOGLE.COM

Quoc V. Le

Google

QVL@GOOGLE.COM

Machine: *hi , this is <NAME> from helpdesk connect , what can i help you with today ?*

Human: *hi , i forgot my password*

Machine: *ok great*

Human: *can you tell me how i can recover it ?*

Machine: *i ' ll need to verify who it is at that machine*

Machine: *can we do a hangout*

Machine: *?*

Human: *what is the purpose of living ?*

Machine: *to live forever .*

Human: *what is the purpose of existence ?*

Machine: *to find out what happens when we get to the planet earth .*

Human: *where are you now ?*

Machine: *i 'm in the middle of nowhere .*

Human: *what is the purpose of dying ?*

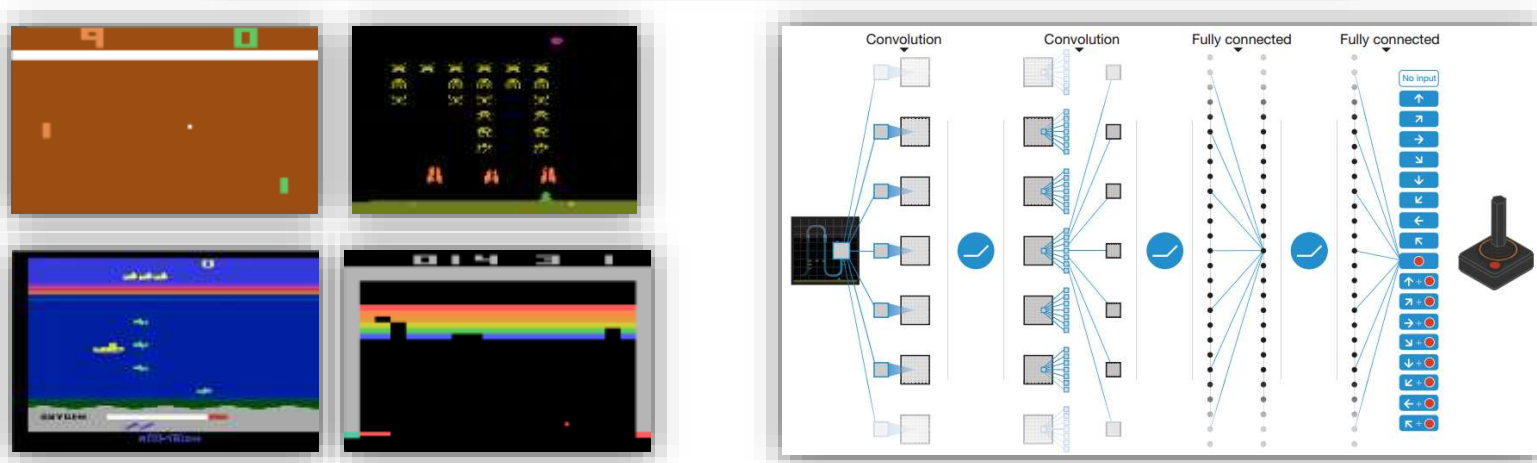
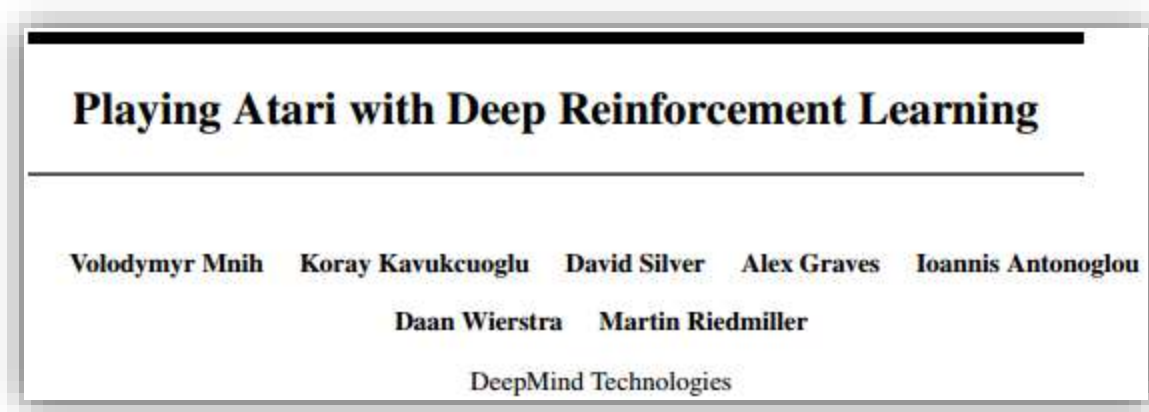
Machine: *to have a life .*

ニュースになった論文：2015年

Atariのゲームを試行錯誤しながら上手にプレイできるように

UL Systems, Inc.

- Atariのさまざまなゲームを、ゲームごとに特別にプログラミングすることなく、強化学習で上手にプレイできるようになったという論文です。
- この論文の場合、ソースコードも公開されています。luaというJavaScriptに似た言語で書かれていて、ライブラリーとしてはtorchが使われていました。ソースコードのトータル行数は1460行しかありません！GPUだけでなく、CPUでも動作するようになっていました。（そのような起動スクリプトが入っています）



(画像は論文からの抜粋)

- AlphaGoが世界棋士レーティングで4位のイ・セドルに勝利しました。今まで囲碁でコンピューターがプロに勝つ事はまだまだ先のことだと言われてきましたが、これをディープラーニングで達成しました。

Mastering the Game of Go with Deep Neural Networks and Tree Search

David Silver^{1*}, Aja Huang^{1*}, Chris J. Maddison¹, Arthur Guez¹, Laurent Sifre¹, George van den Driessche¹, Julian Schrittwieser¹, Ioannis Antonoglou¹, Veda Panneershelvam¹, Marc Lanctot¹, Sander Dieleman¹, Dominik Grewe¹, John Nham², Nal Kalchbrenner¹, Ilya Sutskever², Timothy Lillicrap¹, Madeleine Leach¹, Koray Kavukcuoglu¹, Thore Graepel¹, Demis Hassabis¹.

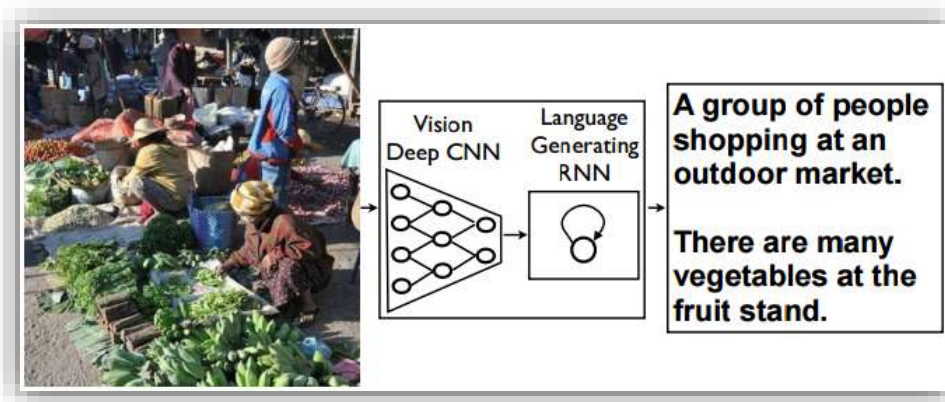
- convolutional neural network (CNN)が碁に最初に適用されたのは1994年だったそうです。ただ、当時のニューラルネットワークは層が浅いものでした。2015年にDeepMindは12のコンボリキュレーション層とsoftmaxを使い、探索処理なしのNNによる直感勝負で当時最強の囲碁プログラムと互角の強さのプログラムを作っています。(論文：Move evaluation in Go using deep convolutional neural networksに掲載)
- 2016年の上記論文はフランスのプロ棋士であるFan Huiに勝利した後に書かれたもので、AlphaGoのアルゴリズムが説明されています。より詳しい内容は後述しますが、以下のような関数をNNで構築し、それを用いてゲーム木の探索（深読み）を行うことで強い囲碁プログラムを作ること成功しています。
 - バリューネットワーク
局面の評価を行うニューラルネットワークです。局面を特殊加工したテンソルを引数に入れると、戻り値として勝率が数字で出てくるようなものです。
 - ポリシーネットワーク
局面を特殊加工したテンソルを引数に入れると、戻り値として次の一手をどこに打つべきか出力してくれます。
 - ロールアウト用ポリシーネットワーク
ロールアウト（プレイアウトとも呼ばれる）に用いるポリシーネットワークです。早打ち用のポリシーネットワークと言ってもいいです。ポリシーネットワークは3ミリ秒かかりますが、これは2マイクロ秒で計算できます。

他にも色々・・・

全部ディープラーニング(ディープニューラルネットワーク)です。

UL Systems, Inc.

- 音声認識が実用段階に。(論文: Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition)
→ 今の時代、声で入力あたりまえ。
- 画像を入力したら、自然言語(たとえば英語)で画像に対して説明文を適切に生成してくれるものが作れた。(論文: Show and Tell: A Neural Image Caption Generator)



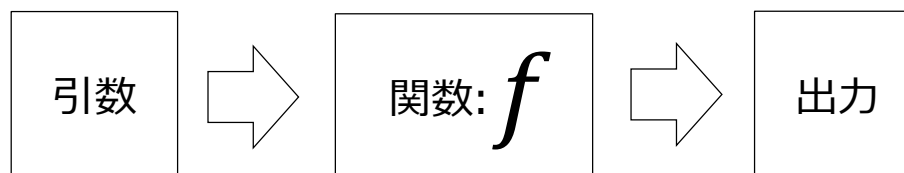
- いろいろな画像を、画家の作風を真似た画像に自動変換するプログラムが作れた。(論文: A Neural Algorithm of Artistic Style)



ニューラルネットワークは 何に使えるの？

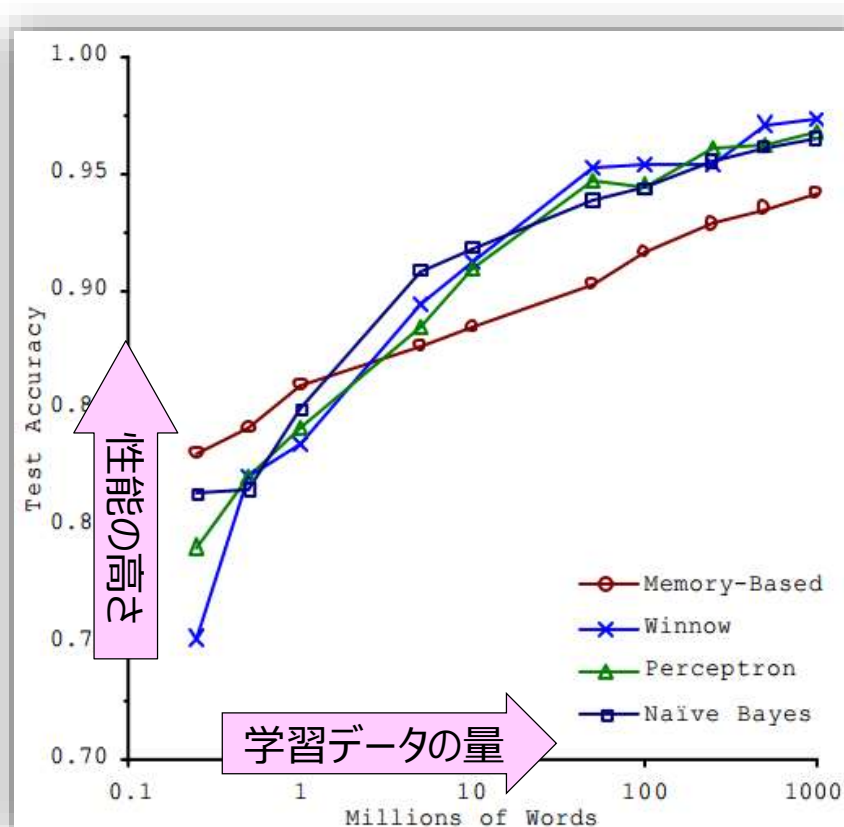
ニューラルネットワークの利用可能範囲

- ニューラルネットワークは、意外にも機械学習で扱われる以下のすべての領域に適用可能です。
 - 教師あり学習
 - 回帰 (regression)
碁の局面を入力すると、出力として勝率（数値）を出してくれるようなものがこれに該当します。
 - 分類 (classification)
写真を入力すると、人の顔が写っているか否かを出力するものや、手書きの数字文字画像を入力すると、0～9の数字のどれに該当しそうかを確率分布として出力してくれるものが該当します。
 - 教師なし学習
 - クラスタリング
自己組織化マップ(SOM)がこれに該当します。
 - 次元圧縮
autoencoderやRestricted Boltzmann Machineが該当します。
 - 強化学習
 - Atariの例や、AlphaGoの例が該当します。
- ニューラルネットワークは任意のチューリングマシンをシミュレートする能力をもつ事が古くから知られています。うまくニューラルネットワークを作れば、かなり利用範囲は広いはずです。



なんでもニューラルネットワークでいいの？

- なんでもニューラルネットワークで済まそう、というのはやりすぎな場合が多いようですが、一方で、2001年にMichele BankoとEric Brillが出した論文によると、学習アルゴリズムの違いはそれほど重要ではなく、使用する学習データの量が性能の決め手になるとも言われています。ここで言う性能は、スパム検出が正確だったり、写真の分類に間違いが少ないといった意味での性能です。



学習アルゴリズムの違いが重要でないならば、なぜDeep Neural Network (DNN) が流行っているのか、と疑問を感じると思いますが、それは今までの機械学習アルゴリズムが出せなかったような性能がDNNでは出せてしまうためです。

一説には、左のグラフを見ると、大量のデータを食わしていても古い機械学習アルゴリズムは、皆、頭打ちになっていますが、この頭打ちがDNNの場合発生しにくいという事が言われています。ただし、最近の研究では少ないデータにおいても、DNNの有効性が分かってきているようですので、大量データが必須というわけでもありません。

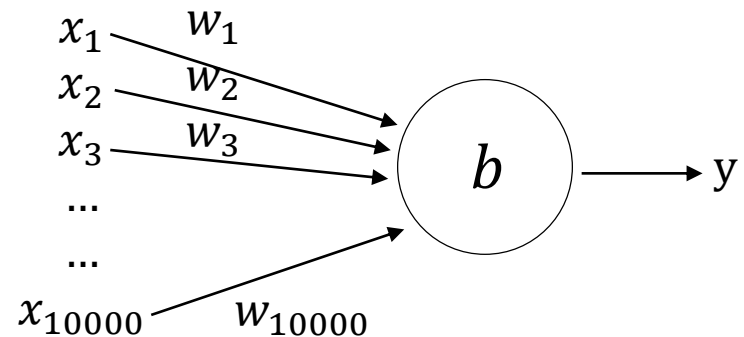
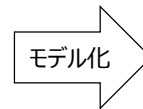
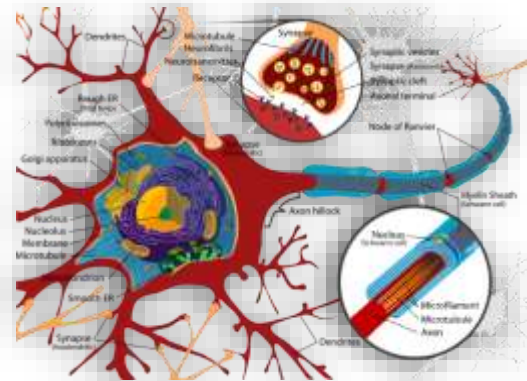
また、研究者にとっては驚きに満ちた成果が出しやすいフロンティアだということも関係していそうです。次々と驚くような発見があり、興味深い論文がハイペースで発表されています。

<http://ucrel.lancs.ac.uk/acl/P/P01/P01-1005.pdf>

ニューラルネットワークの基本

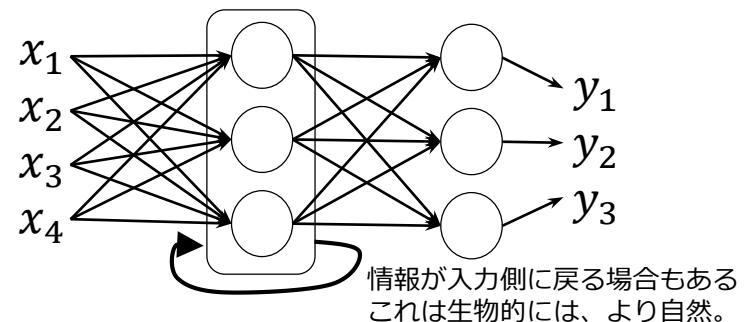
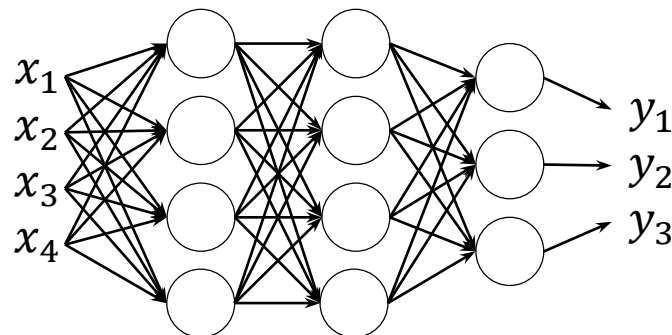
脳とニューロンとニューラルネットワーク

- 「脳は最後のフロンティア」と言われています。脳の仕組みが真の意味で解明されれば、鉄腕アトムやドラえもんのようなヒトに近いロボットが作れそうですが、現状まだそこまで至っていません。とはいえ、fMRIやオプトジェネティクスといった技術によって急速に脳の解明は進んでいますので、いつの日か真の意味で脳の仕組みが分かるかもしれません。
- ヒトの脳を構成する基本的な素子は「ニューロン」と呼ばれる神経細胞です。このニューロンの動作に関しては、かなり昔(1943年)から詳細に分かっています。このモデル化された人工のニューロンをネットワーク状に繋がれたものがニューラルネットワークです。



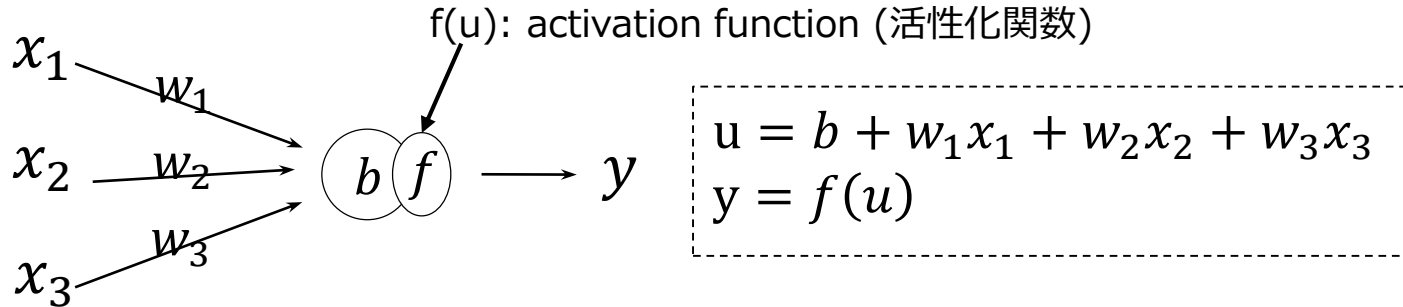
<https://ja.wikipedia.org/wiki/神経細胞>

- 以下のように人工ニューロンをネットワーク状に繋がめます。繋ぎ方は用途によってさまざまです。



ニューラルネットワークの1つのノード(ユニット)について

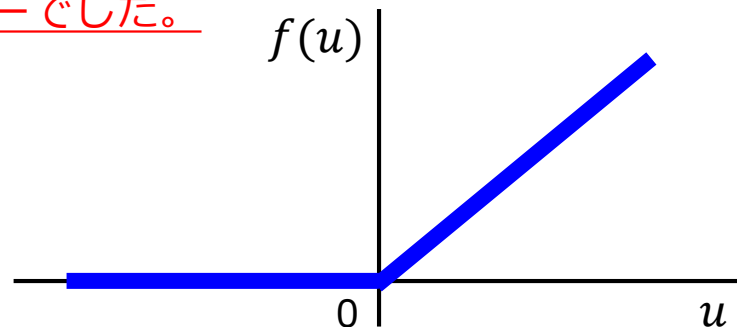
- ニューラルネットワークを構成するノード(人工ニューロン)は以下のようなものです。まず、複数の入力 x_1, x_2, x_3 があります。入力は、他のニューラルネットワークのノードからの出力から来るものもありますし、画像の画素を入力として受け取るものもあります。この入力1つ1つに対し、重み付け w_1, w_2, w_3 があります。また、ノード1つには1つのしきい値 b (バイアス)を数値として持ちます。ノードからの出力結果はここでは y とします。 y を計算する際にはactivation functionである f を介して算出されます。



- activation function $f(u)$ として、古くはシグモイド関数が使われてきましたが、現在は rectified linear (ReLU) という活性化関数がとてもよく使われます。2011年のReLUの発見により、より深いニューラルネットワークの学習(ディープラーニング)が可能になったためです。ReLUの発見は大きなブレイクスルーでした。

ReLU

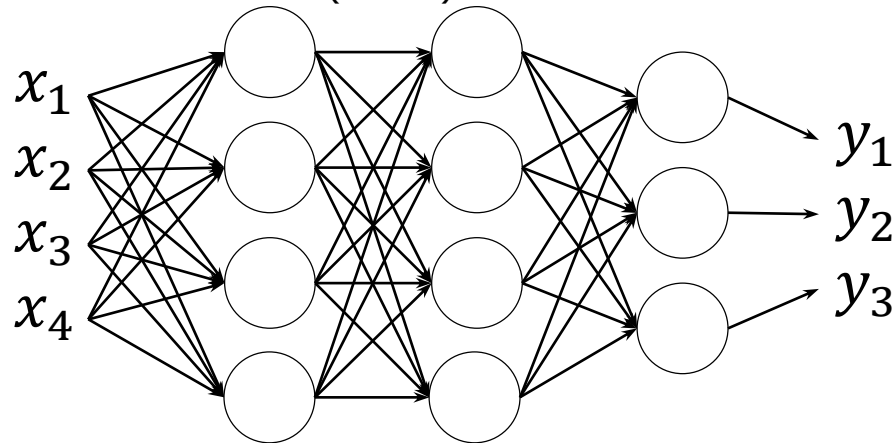
$$f(u) = \max(u, 0)$$



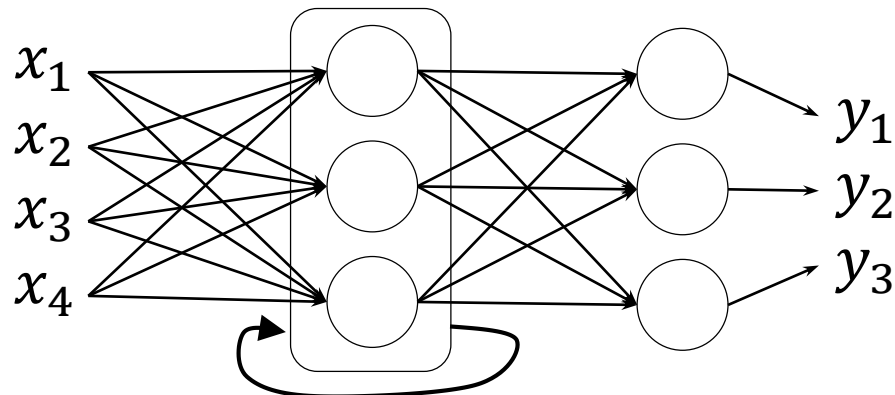
ニューラルネットワーク

- 実用化されているニューラルネットワークには、大きく2つの種類があります。

- feedforwardニューラルネットワーク(FFNN)



- recurrentニューラルネットワーク(RNN)



- 主な違いは、情報の伝達が一方方向であるか、それとも再度出力情報が入力側に戻る可能性があるかです。画像認識などで広く使われているのはFFNNです。RNNは情報のシーケンスを扱う際に用いられます。たとえば、音声情報を扱う際や、文章の自動生成などで使われます。
- FFNNとRNNは組み合わせて使われる事もあります。

ニューラルネットワーク(NN)へ与える入力データのイメージ

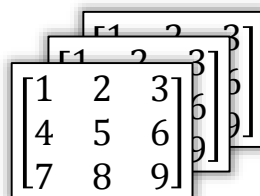
- ニューラルネットワークに対する入力データは以下の様な形をした、固定の大きさの多次元配列（数学的にはテンソル）です。

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

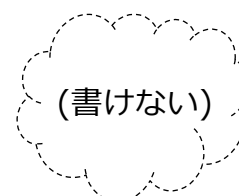
1次元配列
(ベクトル)

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

2次元配列
(行列)



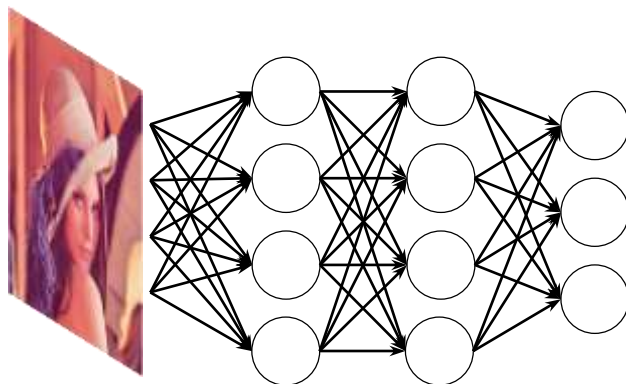
3次元配列
(テンソル)



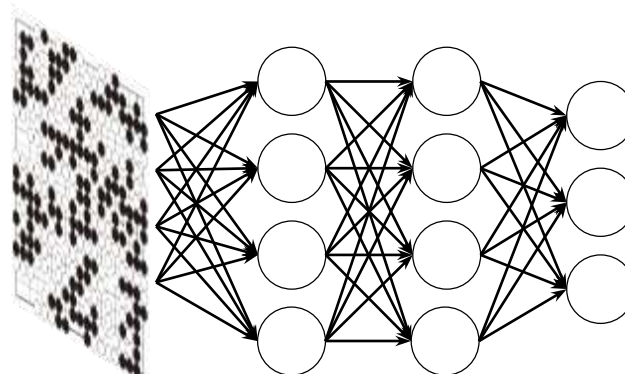
4次元配列
(テンソル)

...

- たとえば、モノクロ画像をラスタスキャンしたものは1次元配列になり、NNへの入力に出来ます。モノクロ画像そのままであれば、2次元配列となりますから、これもNNへの入力にそのまま使えます。ちなみに碁の局面も19x19の2次元配列になりますから、そのままNNへの入力として使えます。カラー画像は1つのピクセルにつきRGBの3つのデータが入りますから、3次元配列となります。これも、NNへの入力として使えます。



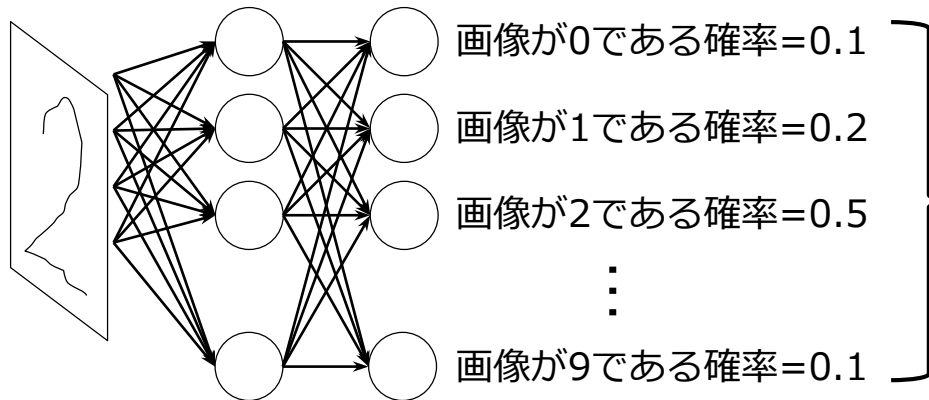
写真のなまのRGBピクセルデータを入力



碁の局面のデータを行列として入力

ニューラルネットワーク(NN)からの出力データのイメージ

- ニューラルネットワークからの出力データに関しては、主に2つのデータ表現を用います。1つは単なる1つの数値です。これは、回帰用にニューラルネットワークを使う場合に使用します。もう一つは、確率分布を出力するタイプです。
- 出力として確率分布を出力するというイメージは分かりにくいですが、具体的には以下の様なものです。



画像が2である確率が一番く出力されたから、「2」と判定しているみたいだ。

- 上記のような確率分布でデータを入出力する際、ニューラルネットワークでは「one hot vector」という表現もよく用いられます。これは、該当ノードだけ確率を1とし、他を0とした分布です。以下の例は数字の表現方法です。アルファベットやASCIIコードを扱う場合も同じ要領になりますが、もっとベクトルの長さが長くなります。

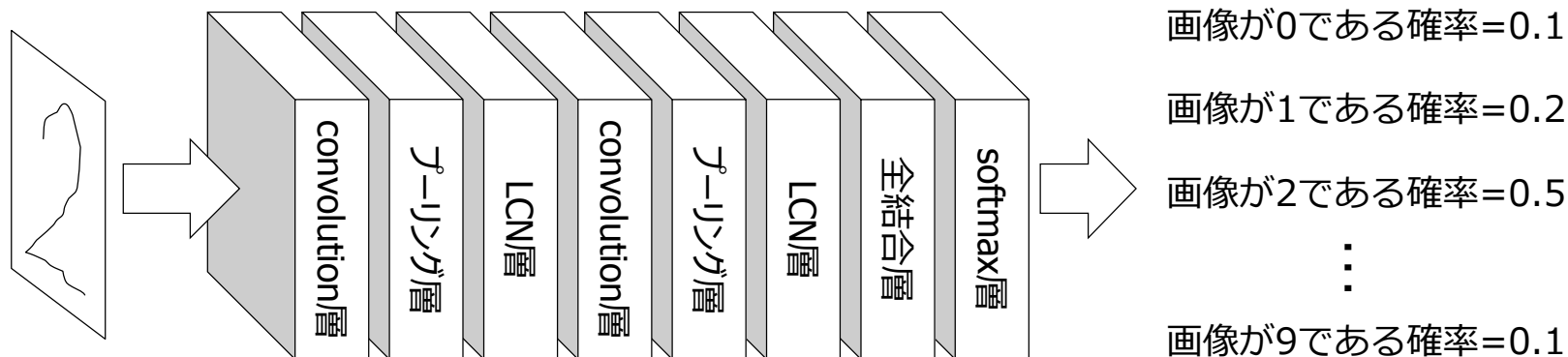
$$0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, 1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, 2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, 3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, 4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, 5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, 6 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, 7 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, 8 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, 9 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

ニューラルネットワークにおける さまざまな種類の層（レイヤー）

ニューラルネットワークを構成する層（レイヤー）の意味を理解することが、ディープラーニングを理解する近道

- ニューラルネットワークでは、さまざまな層を組み合わせることで1つのニューラルネットワークを構成し、全体として何かの機能を構築します。層にはさまざまな種類があるため、層の種類を理解することがニューラルネットワークを理解する近道です。

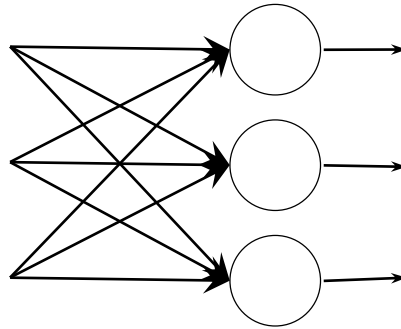
＜典型的な画像認識用 convolutional neural network (CNN)＞



- つまり、上記で言う「convolution層」「プーリング層」「LCN層」「全結合層」「softmax層」などの内容が理解できれば、各種論文が使用しているディープラーニングのニューラルネットワークの構造が理解出来るようになります。

全結合層：汎用の層

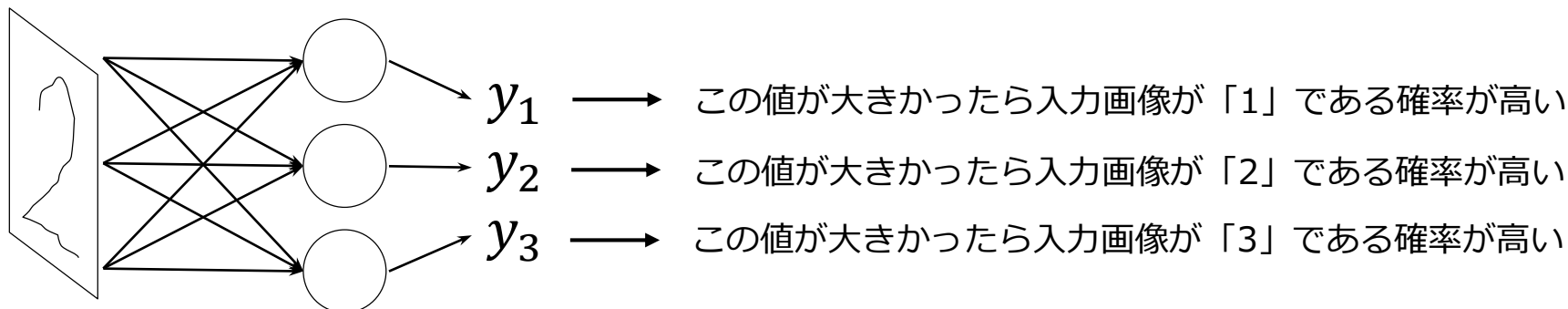
- 全結合層は古くからニューラルネットワークで使われてきた層です。用途は特に限定されておらず、汎用の層として利用可能です。現在でもニューラルネットワークの表現力を底上げするためによく利用されます。



- ネットワークの形としては、1つ1つのユニット（人工ニューロン）が前の層のすべてのユニットと結合している事が特徴です。このため、最も汎用性の高い層となっています。
- 昔はこの全結合層を2～3層重ねた簡素なニューラルネットワークがよく使われていました。現在でも素朴なニューラルネットワークを思い浮かべる際には、この全結合層が重なっているものと考えておけば、おおよそ間違いありません。
- 活性化関数はLeRUが使われる事がほとんどです。

softmax層：ニューラルネットワークにデータを分類させたい時に、最終層として使うもの

- たとえば郵便番号のような手書き文字画像を0～9までの10個の数字に分類したい場合や、囲碁で現在の局面から次の一手としてどこが良さそうかを確率分布で出力したい場合などに、ニューラルネットワークの最終層としてよく用いられます。



- softmax層はネットワーク構造としては全結合層と同じです。異なるのは活性化関数で、softmaxという特殊なものを使用します。
- softmaxでは、まず、各ノードでactivation functionにかける前の値 u_i を $\exp(u)$ で変換します。通常のactivation functionであればこの計算値をそのまま出力するところですが、softmaxの場合は、 $y_1 + y_2 + \dots + y_n$ が1になるように全体の値を調整します。これは、確率分布として扱えるようにするためには、全体の確率の和が1である必用があるためです。数式で記述すると、以下のようになります。

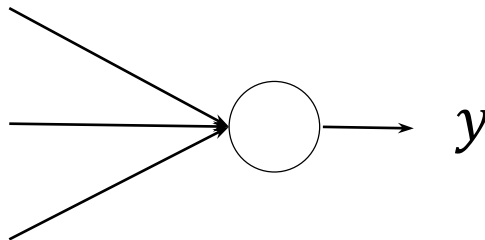
$$u = b + w_1x_1 + w_2x_2 + w_3x_3 \text{ で計算できる各 } u_i \text{ に対して、}$$
$$y_i = \frac{\exp(u_i)}{\sum_{i=1}^n \exp(u_i)}$$

regression層(回帰層) :

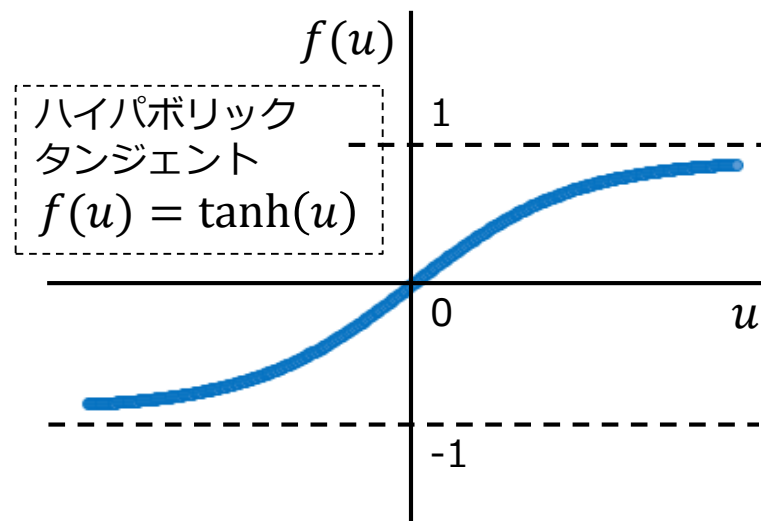
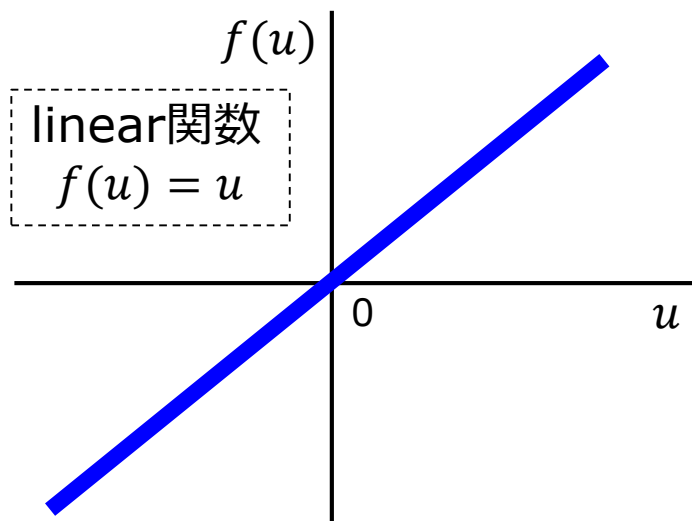
ニューラルネットワークから数値を出力したい場合に使用します

UL Systems, Inc.

- regression層はニューラルネットワークから数値を出力したい場合に使用します。たとえば、囲碁の局面を数値として評価するためのニューラルネットワークの最終層にregression層を使用することで、数値を得るといった目的で使います。



- activation function $f(u)$ としては、 y の値として欲しい範囲で使うものが変わります。最もベーシックなものはlinear関数です。なお、AlphaGoでは $\tanh(u)$ が使われていました。



convolution層の説明の前に・・・

画像処理におけるconvolution(たたみ込み)演算

UL Systems, Inc.

- 画像処理では、convolutionという処理が一般的に広く使われています。たとえば、スムージングやエッジ検出などで使います。このような概念はJava 2D APIのConvolveOpクラスなどで導入されていますし、GIMPなどのツールでも、convolution行列を直接入力して変換を行う処理が提供されています。（GIMPのメニューから、フィルター>汎用>コンボリューション行列で実行できます）



元画像

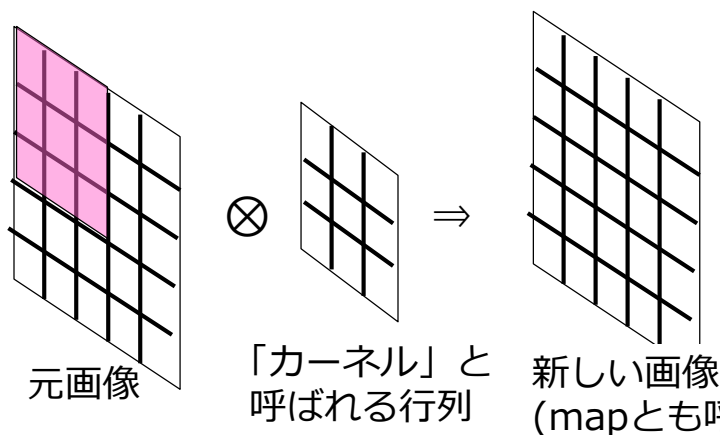


垂直方向エッジ検出



水平方向エッジ検出

- convolutionの考え方は以下の通りです。



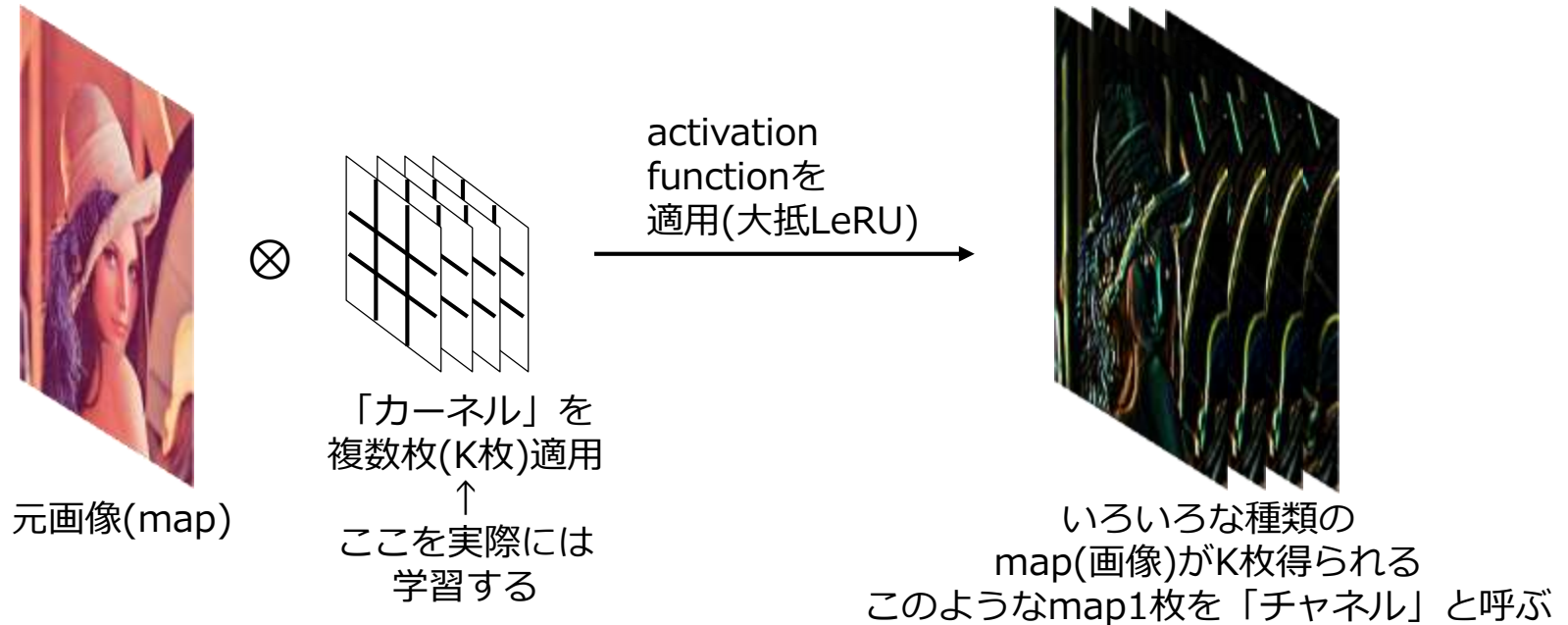
元画像に対し、カーネルと同じサイズのウィンドウをずらしながら計算を行います。新しい画像のピクセル値は、以下のように計算します。

$$\begin{array}{ccc} x_1 & \otimes & h_1 \\ x_2 & & h_2 \end{array} \Rightarrow x_1 h_1 + x_2 h_2$$

元画像 カーネル

convolution層(たたみ込み層) : データを色々な見方(複数種類のフィルター)で見るための層

- convolution層は、前述のconvolution演算をニューラルネットワークにおける層として扱います。(なお、本来はバイアスも加算されます。)



- 通常の画像変換の際には「スムージング用のカーネル」や「エッジ検出用のカーネル」などの手で作られたカーネルを用いますが、convolution層ではカーネルに用いる行列をデータから自動生成します。自動生成したカーネルは、手で作ったカーネルに比べて、よりデータの特徴をうまくとらえたカーネルとなります。

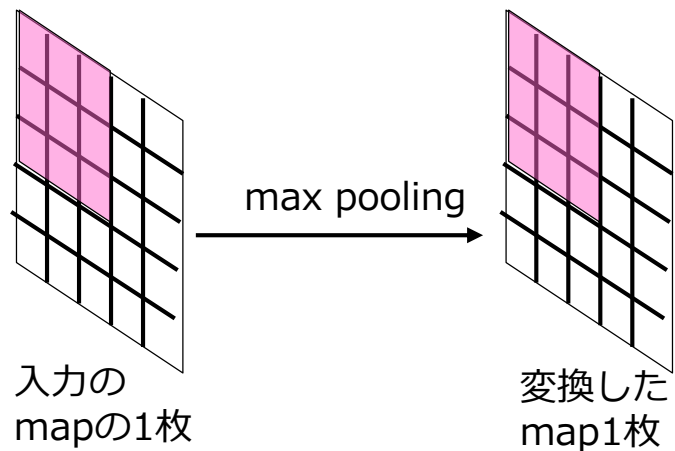
参考 : http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

pooling層:

画像などのデータ中にある「特徴」の位置依存性を緩和するための層

UL Systems, Inc.

- convolution層の後にpooling層という層を置くことが多いです。pooling層は、画像などのデータ中にある「特徴」の位置をぼやかすための層として使用します。
- pooling層の典型的なものとして、max poolingというものがあります。max poolingは、ウィンドウ内の最大ピクセル値を計算するという、とても簡単な変換をしています。



元mapに対し、ウィンドウをずらしながら計算を行います。新しい画像のピクセル値は、単純に、ウィンドウ内の最大ピクセル値となります。

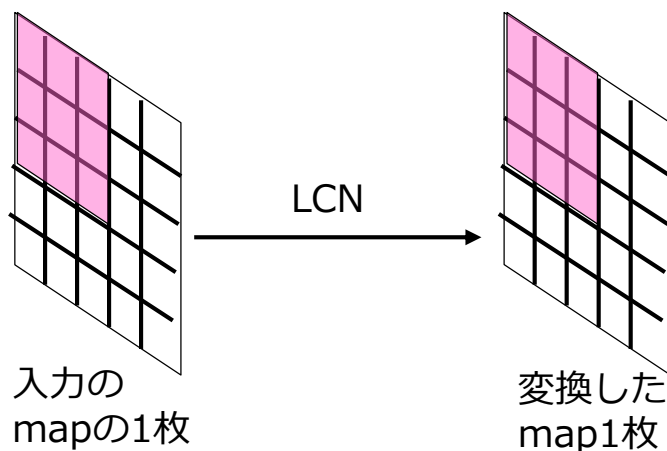
$$\begin{array}{c} x_1 \\ x_2 \end{array} \xrightarrow{\text{max pooling}} \max(x_1, x_2)$$

元mapのウィンドウ

- なお、pooling層は学習の対象となるようなパラメーターはありません。決まった計算をするだけの層です。

LCN層 (局所コントラスト正規化層): 画像などのコントラスト補正のための層

- pooling層の後にLCN(local contrast normalization)と呼ばれる局所コントラスト正規化の処理を挟み込むことがあります。
- 通常、画像はカメラの露出や光源の状況次第で画像全体の明るさやコントラストが大きく変化します。このコントラストの変化を無視したい場合、LCN層を用います。

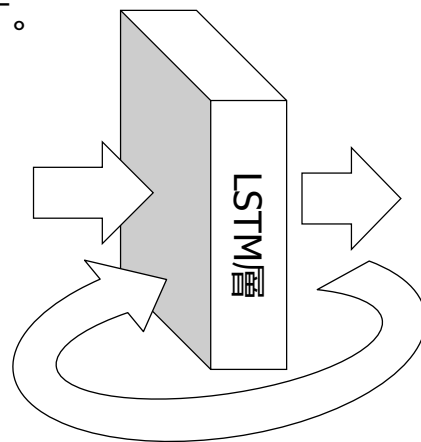


元画像に対し、ウィンドウをずらしながら計算を行います。新しい画像のピクセル値は、単純に、ウィンドウ内の平均を差し引くという計算を行います。目的は、コントラスト補正です。

- なお、Pooling層と同様に、LCN層でも学習の対象となるようなパラメーターはありません。

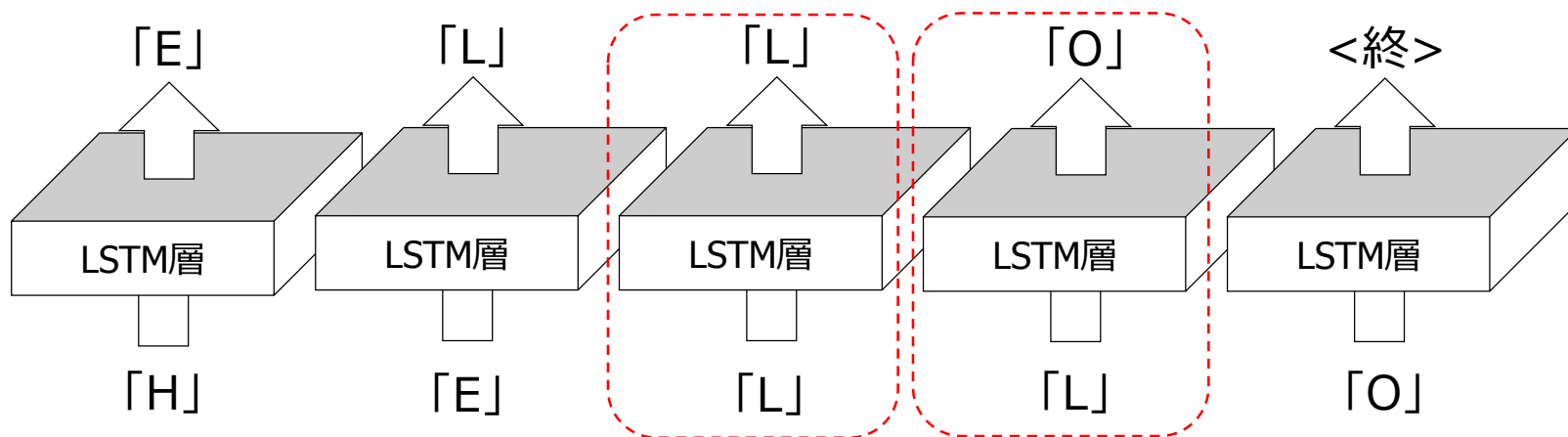
LSTM層 (Long short-term memory層): シーケンスを扱うための層

- LSTM層は他の種類の層とはずいぶん異なり、層の中に状態を持っています。層の中に状態を持つことにより、層をデータが通過するたびに内部状態が変化し、それによってそれまで流されたデータに応じた出力が可能となっています。



出力された情報が再度入力に戻る。
また、1回データが通る
たびに、内部状態が更新される。

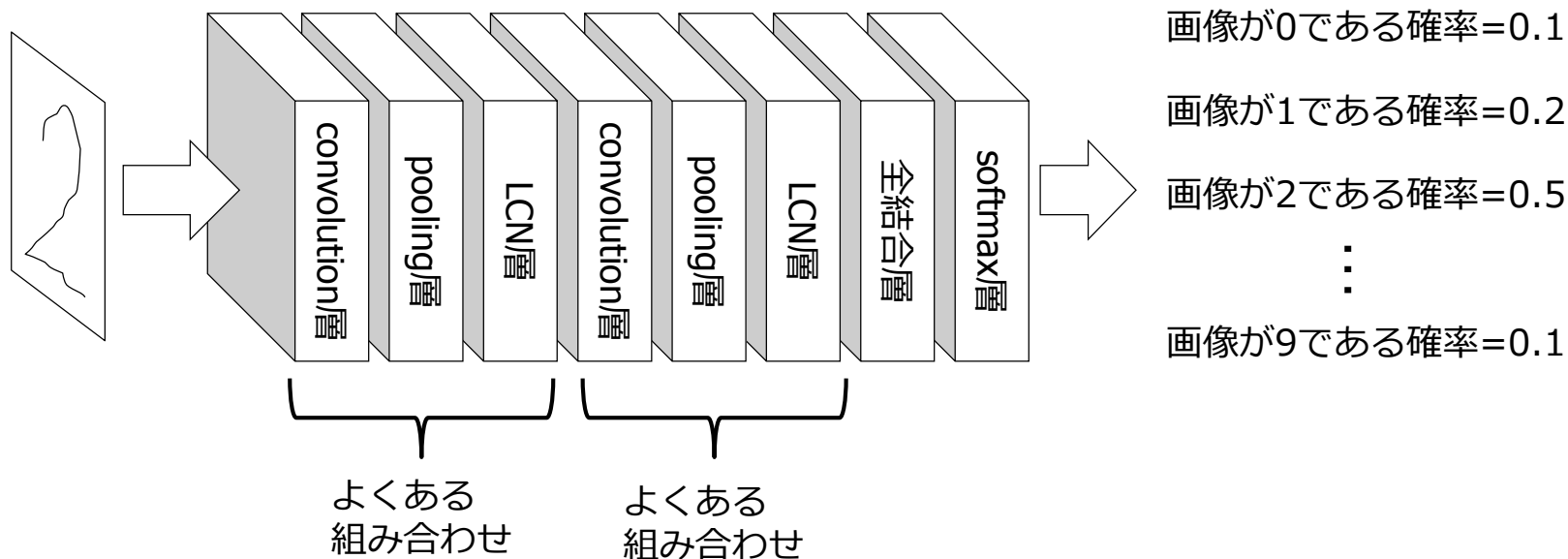
- LSTM層には、たとえば単語を覚えさせる事が出来ます。この場合、入力としてASCIIキャラクターの「one hot vector」を用い、出力としてASCIIキャラクターの確率分布を出すようにすると、「HELLO」は以下のように覚えさせられます。「L」の次の文字が固定されていないのがポイントです。



同じ「L」を入力しても、文脈によって結果は異なる

convolutional neural network (CNN) の例

- convolutional neural network (CNN) の例を以下に再度示します。



- よく見かけるCNNは、convolution層, pooling層, LCN層の3層を1セットとし、それを何度か繰り返し適用した後に、全結合層をいくつか適用し、最後の層にsoftmax層を利用してデータの分類を行います。
- CNNは、結局のところ、何通りもの画像変換を介して、適切にsoftmaxによるデータの分類が行えるようにデータを咀嚼した後、softmax層でデータを分類するという構造となっています。
- このCNNを訓練するためには、教師データを用いて教師あり学習を行います。学習にはバックプロパゲーションを使用します。（昔はautoencoderなどの事前学習を使っていたましたが、最近は使われず、単純にバックプロパゲーションを用いて学習を行います）

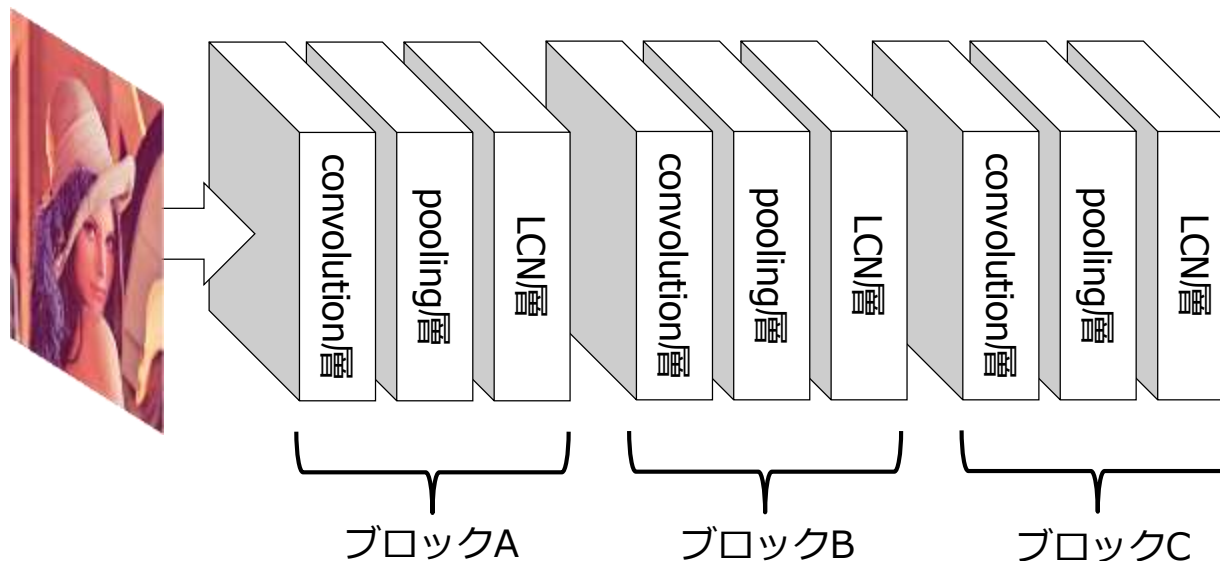
ニュースになった論文を振り返る

ニュースになった論文：2012年

Googleのコンピュータが自動的に猫を認識できるようになった

UL Systems, Inc.

- この論文では以下の様にconvolution層, pooling層, LCN層の3層を1セットとし、それを3回繰り返したNNを用いています。ただし、訓練の仕方が特殊で、autoencoderという訓練方法を用いています。



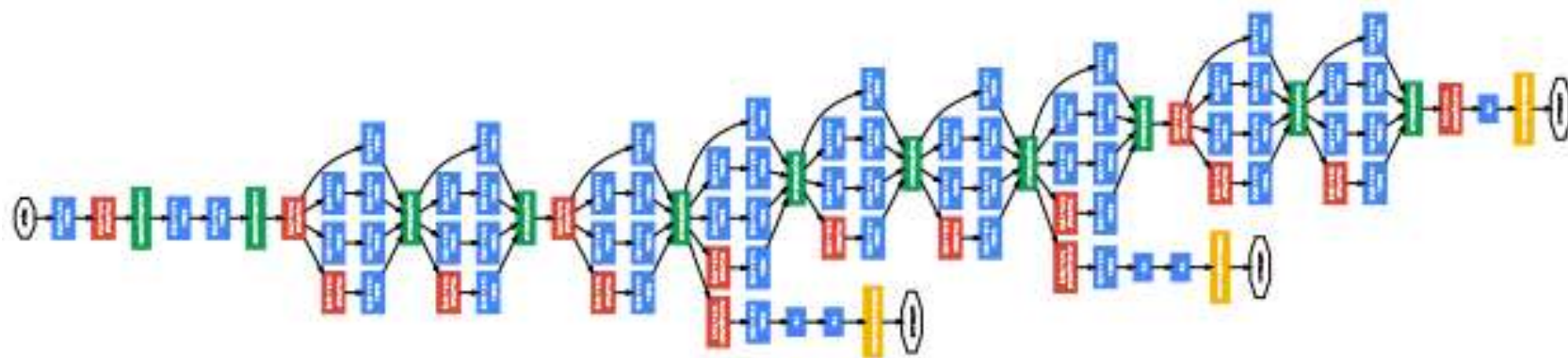
- autoencoderでは、まずブロックAを訓練し、次にブロックB、最後にブロックCを訓練するといった訓練方法を行います。（autoencoderの具体的な内容説明はスキップしてしまいます）
- このように訓練したところ、各LCN層のユニットに、以下のような性質を持つユニットが現れた、というのがこの論文の内容となります。
 - ブロックA：ナナメの線や縦の線など、概念的に単純な画像に反応するユニットが登場
 - ブロックB：目や鼻といった、単純なオブジェクトに反応するユニットが登場
 - ブロックC：人の顔や猫の顔といった、より上位の概念を持つ画像に反応するユニットが登場

ニュースになった論文：2012年 画像認識コンペティションでディープラーニングが圧勝

- コンペティションで勝利したSuperVisionの構造は以下の通りです。



- なお、2014年にGoogleはGoogLeNetという画像認識用ニューラルネットワークを構築していますが、以下のように非常に深いものになっています。SuperVisionが打ち立てた16.4%の誤認識率は更に下がり、6.67%まで誤認識率が下がっています。これは人間の認識能力に匹敵するものです。



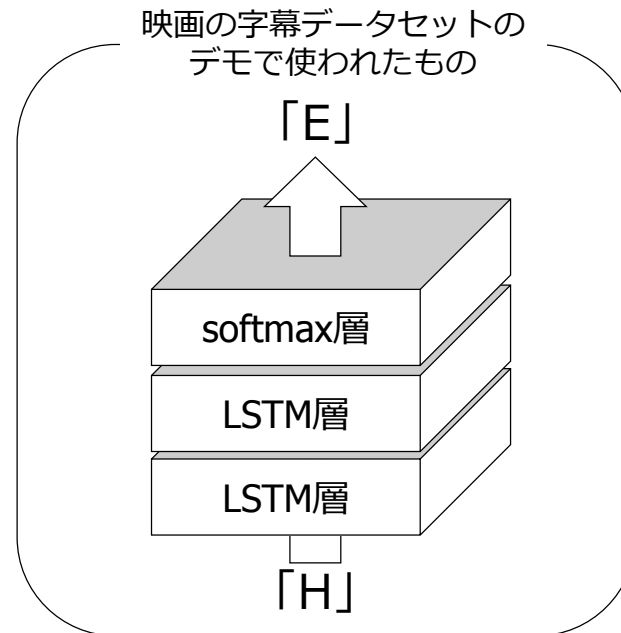
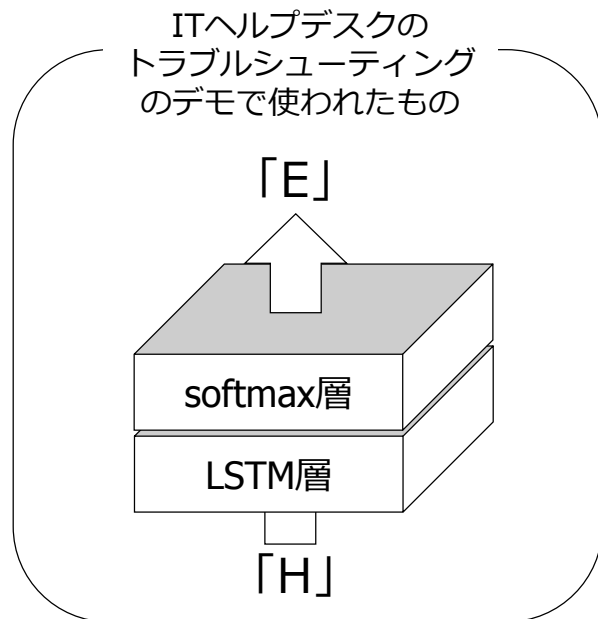
論文：「Going Deeper with Convolutions」より引用

ニュースになった論文：2015年

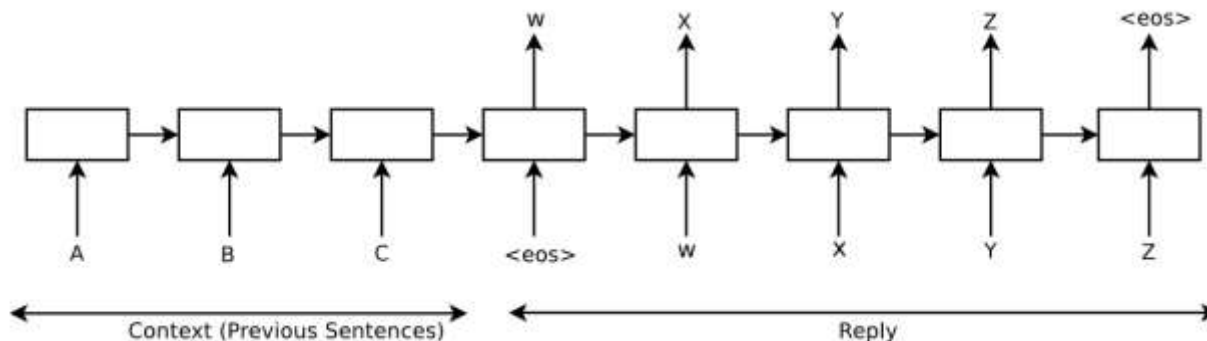
LSTMで上手な受け答えが可能なチャットボットが作れた

UL Systems, Inc.

- この論文で使われたニューラルネットワークは以下の通りです。



- これらのLSTMベースのNNを以下のように使うことによってチャットシステムを作成しています。
まず、人の入力（A）はNNへ順に入力しますが、NNからの出力は無視します。入力（A）の終わり（eos）に来到、NNからの出力を入力側に単に返しはじめ、NNがeos(end of statement)を返すまで繰り返します。



論文：
「A Neural
Conversational
Model」より引用

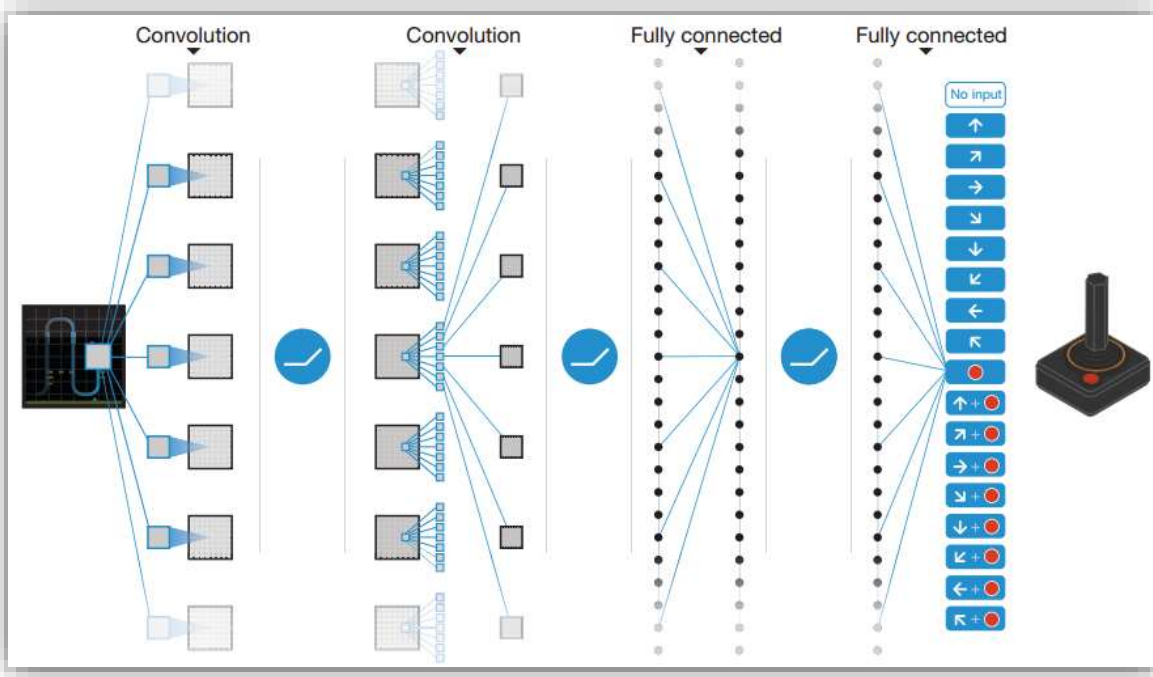
ニュースになった論文：2015年 Atariのゲームを試行錯誤しながら上手にプレイできるように

UL Systems, Inc.

- 以下がこの論文で用いられたニューラルネットワークです。



最後の出力である「操作」は、以下の論文の図にあるような各種操作パターンを意味します。たとえば「ボタンを押しながら斜め上にスティックを倒す」操作などです。操作の種類としては「入力なし」を含めると18通り存在します。



このNNを訓練する際には、以下の3つの情報を与えています。

- ・ ゲームの画面(毎秒60フレーム)
- ・ 現在のゲームのスコア
- ・ ゲームオーバーか否か
- ・ 可能な操作の種類

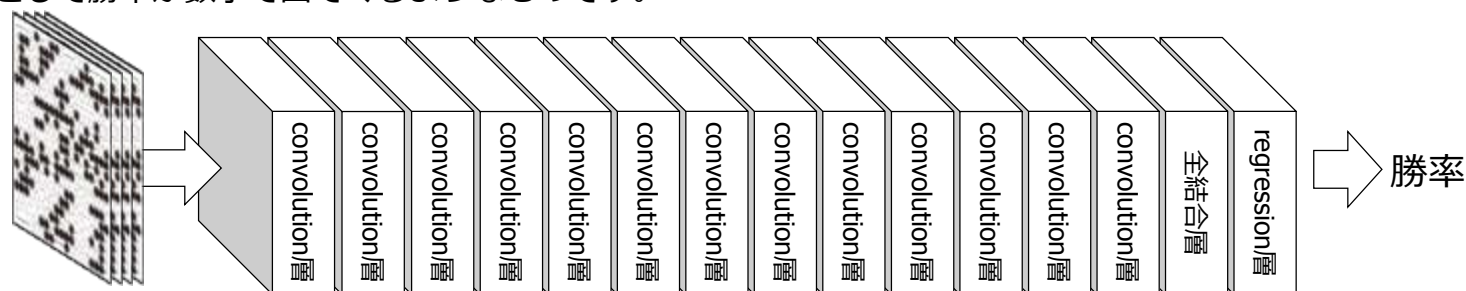
ニュースになった論文：2016年

AlphaGoが世界棋士レーティングで4位のイ・セドルに勝利

- AlphaGoは以下の3つのニューラルネットワークを用いています。

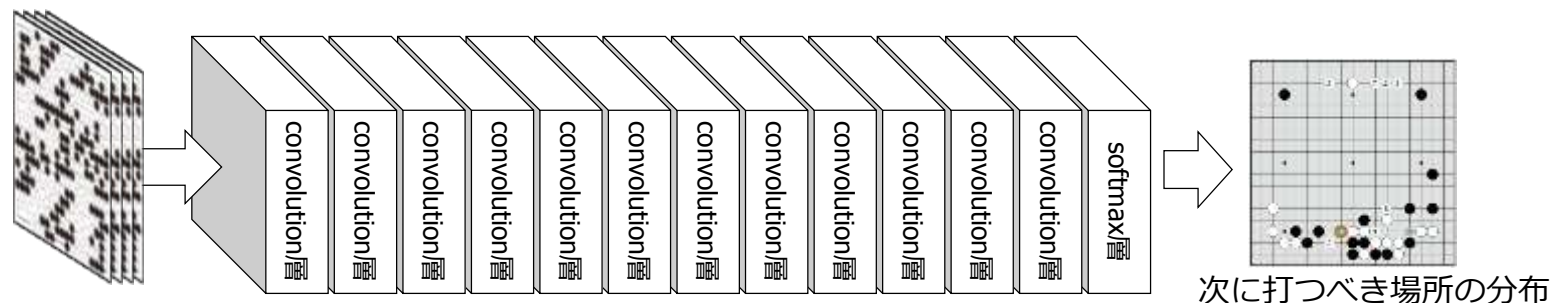
- バリューネットワーク

局面の評価を行うニューラルネットワークです。局面を特殊加工(碁のルールにもとづく)したテンソルを引数に入れると、戻り値として勝率が数字で出てくるようなものです。



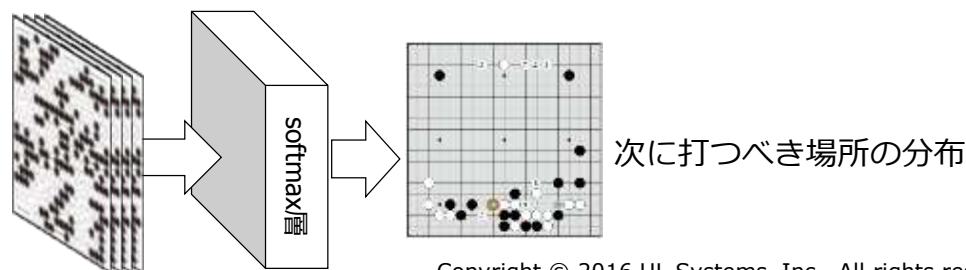
- ポリシーネットワーク

局面を特殊加工したテンソルを引数に入れると、戻り値として、ぱっと見、次の一手をどこに打つべきかを出力してくれます。



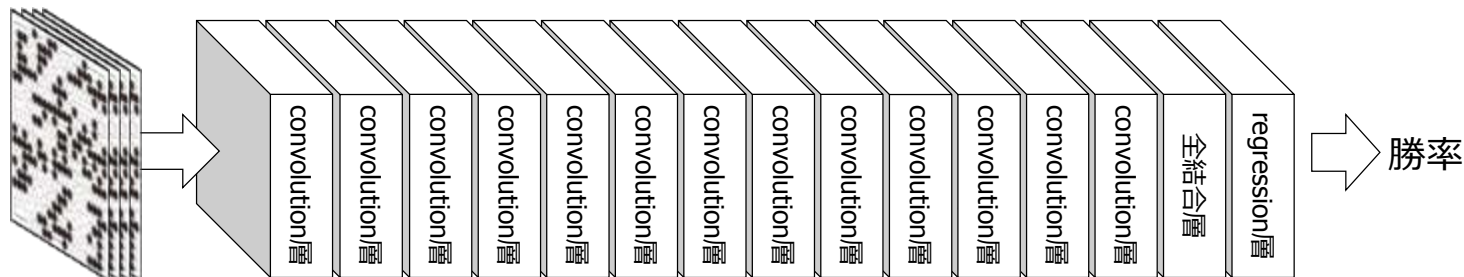
- ロールアウト用ポリシーネットワーク

ロールアウト (プレイアウトとも呼ばれる) に用いるポリシーネットワークです。早打ち用のポリシーネットワークと言ってもいいです。ポリシーネットワークは3ミリ秒ほど計算にかかりますが、これは2マイクロ秒で計算できます。

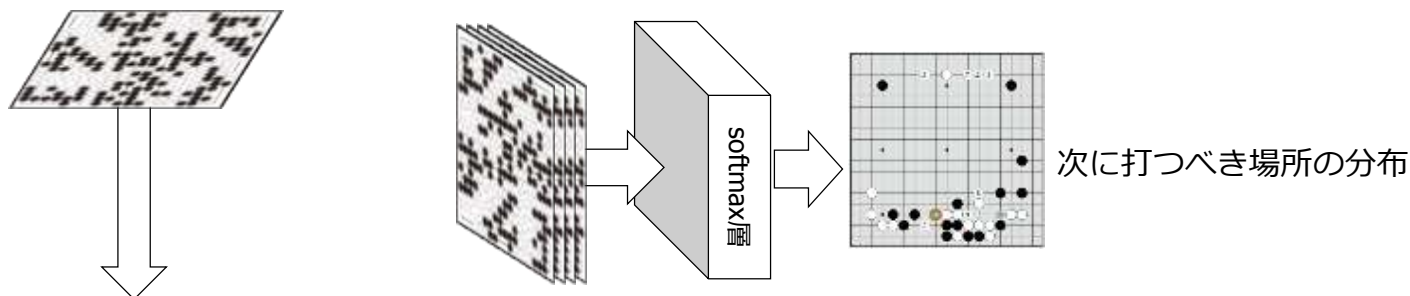


AlphaGoにおける局面の評価方法

- AlphaGoでは、2種類の局面評価を行い、これらの局面評価結果をマージして、実際の局面評価結果としています。
 - バリューネットワークによる局面評価
バリューネットワークは局面の評価を行うニューラルネットワークです。つまり、バリューネットワークの引数に局面を入力すれば、その局面の勝率が計算されます。これが1つ目の局面評価方法です。



- ロールアウトによる局面評価
ロールアウト（もしくはプレイアウト）という方法を用いた局面評価が2つ目の局面評価方法です。ロールアウトとは、具体的にはロールアウト用ポリシーネットワークを用いて、現在の局面から終局まで打ち続け、どちらが結果的に勝ったかを見る方法です。碁盤は打つ所が361箇所しかなく、ロールアウト用ポリシーネットワークは高速なので、単純計算で722マイクロ秒で終局の陣地がどうなりそうかを見ることが出来ます。終局の陣地の状態を見れば、どちらがどれだけ勝ったのか見ることが出来ます。



終局まで高速に打ってみて、終局の状態を知り、それをもって、現在の局面の判断を行う。

- 上記2種類の局面評価は互いに補完しあう関係にあるようで、どちらの局面評価も重要です。よって、AlphaGoではこの2種類の局面評価を「足して2で割る」ことで最終的な局面評価としています。なお、前ページのポリシーネットワークに関しては、どんな手を深読みするべきかを判定するために使用しています。

- 今回の勉強会ではニューラルネットワークをどう訓練するかをスキップして説明することで、ディープラーニングにおける難しい話をスキップしています。
- ただし、本当に深い話まで分からないとディープラーニングに手がつけられないかという、そうでもありません。世の中には便利なライブラリーがあり、今回説明した程度の内容の理解で色々試行錯誤することは出来る状況にあります。
- 今回説明した有名論文も、たとえばAtariのゲームをプレイするニューラルネットワークはtorchで実装されていたりと、大抵何らかの優秀なライブラリーを使うことが多いのです。
- 最終的には深い理解が必要ですが、そこにたどり着くまで何も動かせないわけではなく、ライブラリーを利用する方向でいろいろな経験をしていく事が、ディープラーニングの勉強を効率よく進めていく1つの方法ではないでしょうか。