

畳み込みニューラルネットワークからのルール抽出

月本 洋[†] 佐藤 優也[‡]

東京電機大学工学部情報通信工学科 〒120-8551 東京都足立区千住旭町 5 番

E-mail: [†] tsukimoto@mail.dendai.ac.jp, [‡] 15ec058@ms.dendai.ac.jp

あらまし 畳み込みニューラルネットワークの内部構造の理解に関しては、可視化の手法があるが、特徴量を抽出する手法はほとんどない。本稿では、畳み込みニューラルネットワークの全結合層からルールを抽出することで、特徴量を抽出する方法を提示する。ルール抽出方法は、以前、筆者の一人が開発した方法(近似法)を用いた。ルールへの入力、畳み込み層の出力なので、ルールを理解する際には、この出力を最大化する画像を求める必要がある。畳み込み層出力最大化画像は、SmoothGrad を用いて求めた。本稿では、このルール抽出方法を MNIST データに適用した結果を報告する。

キーワード 畳み込みニューラルネットワーク、特徴抽出、ルール抽出、近似法、SmoothGrad、MNIST

Extracting rules from convolutional neural networks

Hiroshi TSUKIMOTO[†] and Yuya SATO[‡]

School of Engineering, Tokyo Denki University 5 Senju-Asahi-cho, Adachi-ku, Tokyo, 120-8551, Japan

E-mail: [†] tsukimoto@mail.dendai.ac.jp, [‡] 15ec058@ms.dendai.ac.jp

Abstract To understand the inner structures of convolutional neural networks, several techniques of visualization have been developed. However, few techniques have been developed to extract features from convolutional neural networks. This paper presents a method of extracting features from convolutional neural networks by extracting rules from fully connected layers. The rule extraction method developed by one of the authors was adopted. The inputs of the rules are the outputs of convolutional layers. To understand the rules, the images maximizing the outputs of convolutional layers are needed, which are calculated by Smoothgrad. The method was applied to MNIST data.

Keywords convolutional neural networks, feature extraction, rule extraction, Approximation method, SmoothGrad, MNIST

1. はじめに

畳み込みニューラルネットワークの内部構造の理解に関しては、可視化等の技術[1]がある。しかし、特徴(量)を抽出するような技術はほとんどない。

ニューラルネットワークの内部構造を理解する方法であるルール抽出は、かなり前から研究されて来た[2]。深層ニューラルネットワークに関しても、ルール抽出の研究はいくつかある[3]。しかし、畳み込みニューラルネットワークからのルール抽出は、ほとんど開発されていない。画像が入力であるため、ルール抽出が難しいからである。

本稿では、畳み込みニューラルネットワークの全結合層からルールを抽出することで、特徴(量)を抽出する方法を提示する。ルール抽出方法は、以前、筆者の一人が開発した方法(近似法)[4][5][6]を用いた。

ルールへの入力変数は、畳み込み層の出力なので、ル

ールを理解する際には、この出力を最大にする画像を求める。その出力を最大にする画像を求める方法は、いくつか存在するが、本稿では、SmoothGrad[7]を用いた。SmoothGrad を用いる理由は、手法が簡単で、それなりに良い結果が得られるからである。

本稿では、上記の方法を MNIST データに適用した。なお、本稿では、全結合層は、中間層なし、すなわち出力層のみの場合を報告する。

2 節で基本的な考えを述べる。3 節でルール抽出について説明し、4 節では SmoothGrad を説明する。5 節では、最小構成(畳み込み層 1 層 + 出力層)での実験結果を報告し、6 節では、畳み込み層 3 層 + 出力層の実験結果を報告する。

2. 基本的な考え

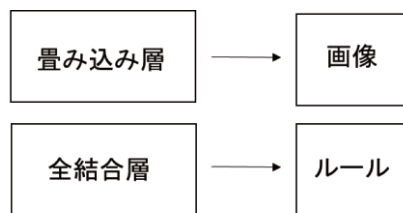
畳み込みニューラルネットワークの全結合層から

近似法で、ルールを抽出する。そのルールは、全結合層の入力で記述されるが、全結合層の入力は畳み込み層の出力なので、ルールは畳み込み層の出力で記述されることになる。「畳み込み層の出力」とは、詳細に書くと、畳み込み層の最終層のプーリング出力である。この出力でルールを記述できるように、本稿では、畳み込み層の最終層の活性化関数は、出力が[0,1]のシグモイド関数を用いる。なお、活性化関数は、出力が[0,1]でない関数でも可能である。その場合は、正規化をすればよい。

普通のニューラルネットワークの場合に、例えば、 $x \wedge y \rightarrow z$ というルールが抽出されたとしよう。 x と y と z はともに[0,1]である。今、 x と y と z を、それぞれ圧力と温度と品質としよう。そうすると、 $x=1$ は、圧力が(最も)高いを意味し、 $y=1$ は温度が(最も)高いを意味し、 $z=1$ は品質が(最も)良いを意味することになる。そして、 $x \wedge y \rightarrow z$ は、圧力が高く、かつ、温度が高ければ、品質が良くなる、と解釈できる。

これと同様に考えると、畳み込み層の最終層の或るプーリング出力の値=1 は、そのプーリング出力を最大化するような画像のことである。したがって、全結合層から得られたルールを解釈するために、畳み込み層の最終層のプーリング出力ごとに、最大化画像を求め、それを入力としてルールを解釈することになる。最大化画像は、誤差逆伝搬法等の方法を試したが、比較的良好な結果が得られた SmoothGrad を用いる。なお、以降、畳み込み層の最終層のプーリング出力を最大化する画像を「最大化画像」と略記する。

本稿では、MNIST データを用いるので、全結合層の出力は 10 個あり、0 から 9 の数字に対応する。したがって、全結合層で抽出されるルールは、例えば、 $x \wedge y \rightarrow z$ (x と y は最大化画像、 z は 0-9 の数字)のような形をしている。図 1 を参照。



ルールの例：画像A \wedge 画像B \Rightarrow 3

図 1 基本的な考え

3. ルール抽出

ルール抽出の方法を大別すると、分解法(decompositional method)と教示法(pedagogical method)の二つになる。分解法は、ニューラルネットワークの

個々の素子を、各々ルール(ブール関数)に近似し、それを合成することで、ニューラルネットワーク全体のルール(ブール関数)を得る方法である。教示法は、ニューラルネットワークに、入力を割り付けて、その出力を計算することで、入力-出力表を獲得し、それから、ルール(ブール関数)を得る方法である。

本稿で用いる近似法は、分解法である。以下で、簡単に近似法を説明する。ニューラルネットワークの各素子をブール関数で近似することが基本である。この場合、ニューラルネットワークの活性化関数は、単調増加であり、[0,1]で正規化されているものとする。

ニューラルネットワークの或る素子の出力を $y = f(x_i)(i = 1, \dots, n)$ とする。 x_i に 0 か 1 を割り付け(代入)すると、 2^n 通りの出力(y)を得る。出力は、0 以上 1 以下なので、0.5 で四捨五入を行い、0 か 1 に近似する。これによって得られた真理値表からブール関数(ルール)を求める。これが近似法の原理であるが、この方法の計算量は(変数の数 n)の指数オーダーなので、時間が非常にかかる。そこで、多項式オーダーの方法を開発した。

或る素子を以下とする。

$$S(p_1x_1 + \dots + p_nx_n + p_{n+1})$$

近似後のブール関数に

$$x_{i1} \dots x_{ik} \overline{x_{i(k+1)}} \dots \overline{x_{il}}$$

が存在するかどうかを、

$$S(p_{n+1} + \sum_{i_1}^{i_k} p_j + \sum_{1 \leq j \leq n, j \neq i_1 \dots i_l} p_j) \geq 0.5$$

で判定する。この方法だと、或る次数(k)までの項を、 n^k のオーダーの時間で生成できる。ただし n は変数の数である。詳細は参考文献[4][5][6]を参照。

4. SmoothGrad

最大化画像を SmoothGrad で求めるので、きわめて簡単に SmoothGrad について説明する。SmoothGrad は、感度解析的な方法である。その概略手順は以下の通りである。

- ① 入力画像の各画素に、正規分布を加える。
- ② プーリング出力の各画素による偏微分を求める。
- ③ ①と②を複数の画像で繰り返し、その平均を求める。

上記の手順により、最大化画像を求める。詳細は参考文献[7]を参照。なお、本稿でのパラメータは以下の通りである。

正規分布の平均：0.0

正規分布の標準偏差：0.1

上記③の画像の数：100 枚

5. 最小構成（畳み込み層 1 層＋出力層）の場合

まず，最小構成（畳み込み層 1 層＋出力層）での実験を行なった．畳み込み層を 1 層にすることにより，フィルターと，SmoothGrad で得られた最大化画像を比べることができ，その正確さを調べることができる．構成と学習条件は以下の通りである．図 2 を参照．

- ・入力：28×28
- ・畳み込み層
 - フィルター数:10,
 - フィルターサイズ:27,
 - パディング:0
 - ストライド:1
 - 活性化関数：シグモイド関数
 - プーリング（高さ:2，幅：2，ストライド:2）
- ・出力層
 - 出力素子数:10
 - 活性化関数:ソフトマックス関数
- ・学習
 - 学習データ:50000
 - テストデータ：10000
 - ミニバッチ：100
 - 誤差逆伝搬反復:10000 回
 - テスト精度： 0.939

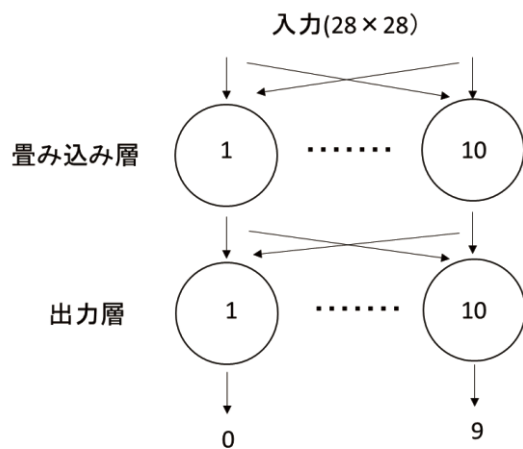
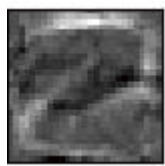



図 2 ニューラルネットワークの構成 1

NO.	フィルター	SmoothGrad
1		

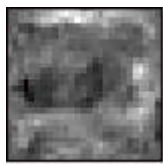

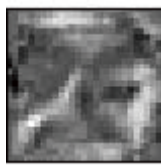



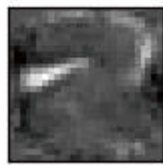

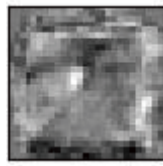

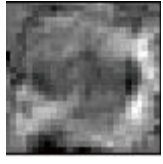

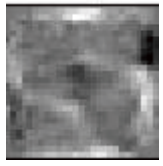

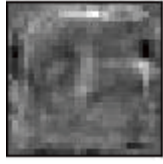

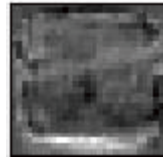

2		
3		
4		
5		
6		
7		
8		
9		
10		

図 3 フィルターと SmoothGrad の結果

図3のフィルターと SmoothGrad を比較すると、良く似ていることがわかる。これから、SmoothGrad による最大化画像が、それなりに正しいことがわかる。

次に、ルール（ブール関数）を記述する。ブール関数は、DNF(積和標準形)で出力されるが、それだと煩雑になり、理解が難しくなるので、精度が良くてリテラル数の少ない積項（論理積だけからなる項）を自動的に選んで、表1に示した。なお、出力素子をこのブール関数に置き換えた場合の精度は 0.959 であり、ニューラルネットワークそのものの精度(0.939)より良い。

表1 ブール関数(畳み込み層1層)

数字	ブール関数
0	$x_9 \wedge \bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_5 \wedge \bar{x}_6 \wedge \bar{x}_7 \wedge \bar{x}_8$
1	$x_2 \wedge x_3 \wedge x_4 \wedge x_6 \wedge \bar{x}_1 \wedge \bar{x}_7 \wedge \bar{x}_8$
2	$x_1 \wedge \bar{x}_2 \wedge \bar{x}_4 \wedge \bar{x}_6 \wedge \bar{x}_7 \wedge \bar{x}_8 \wedge \bar{x}_9$
3	$x_6 \wedge x_8 \wedge \bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_7 \wedge \bar{x}_{10}$
4	$x_2 \wedge x_7 \wedge x_8 \wedge x_9 \wedge \bar{x}_4 \wedge \bar{x}_5 \wedge \bar{x}_{10}$
5	$x_4 \wedge x_9 \wedge \bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_6 \wedge \bar{x}_7$
6	$x_{10} \wedge \bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_4 \wedge \bar{x}_6 \wedge \bar{x}_8 \wedge \bar{x}_9$
7	$x_1 \wedge x_5 \wedge x_6 \wedge x_9 \wedge \bar{x}_2 \wedge \bar{x}_4 \wedge \bar{x}_8$
8	$x_4 \wedge x_8 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_6 \wedge \bar{x}_7 \wedge \bar{x}_9 \wedge \bar{x}_{10}$
9	$x_1 \wedge x_6 \wedge x_7 \wedge x_8 \wedge x_9 \wedge \bar{x}_3 \wedge \bar{x}_4 \wedge \bar{x}_5$

ただし、 x_i は、 i 番目のプーリング出力である。このブール関数と図3の画像で、内部処理を調べてみよう。一例として、数字の3を見てみよう。数字の3のブール関数に肯定で入っている部分は、 $x_6 \wedge x_8$ である。No.6とNo.8の画像を見ると、数字の3に類似している。否定の部分は、 $\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_7 \wedge \bar{x}_{10}$ であるが、肯定の部分ほど直感的な理解はできない。

次に、数字の5を見てみよう。数字の5のブール関数に肯定で入っている部分は、 $x_4 \wedge x_9$ である。図3のNo.4とNo.9の画像を見ると、数字の5に類似している。否定の部分は、 $\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_6 \wedge \bar{x}_7$ であるが、やはり、肯定の部分ほど直感的な理解はできない。

他の数字のブール関数も、同様に解釈できる。このようにして、0-9の数字の分類の特徴(量)らしきものを、或る程度まで理解できる。

6. 畳み込み層が3層の場合

次に、畳み込み層を3層に増やして実験を行った。構成と学習条件は以下の通りである。図4を参照。

- ・入力：28×28
- ・畳み込み層1
 - フィルター数:10,
 - フィルターサイズ:8,
 - パディング:0,

ストライド:1

活性化関数：シグモイド関数

・畳み込み層2

フィルター数:10,

フィルターサイズ:7,

パディング:0

ストライド:1

活性化関数：シグモイド関数

・畳み込み層3

フィルター数:10,

フィルターサイズ:14,

パディング:0

ストライド:1

活性化関数：シグモイド関数

プーリング（高さ:2, 幅:2, ストライド:2）

・出力層

出力素子数:10

活性化関数:ソフトマックス関数

・学習

学習データ:50000

テストデータ:10000

ミニバッチ:100

誤差逆伝搬反復:15000回

テスト精度:0.986

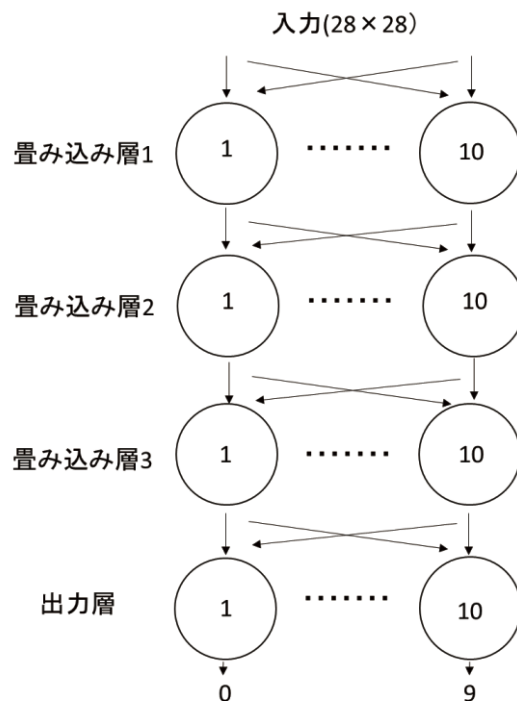


図4 ニューラルネットワークの構成2










NO.	SmoothGrad
1	
2	
3	
4	
5	
6	
7	
8	
9	



図 5 SmoothGrad の結果

図 5 に SmoothGrad の結果を示す．畳み込み層を 3 層にすると，1 層の場合と違い，フィルタの可視化で最大化画像が正しいかどうかは判定できないが，1 層の結果(5 節)の結果を踏まえれば，それなりに正しい結果であろうと思われる．

次に，ルール(ブール関数)を記述するが，これについても畳み込み層 1 層の場合と同様に DNF で出力され，その出力の中から，精度が良くてリテラル数の少ない積項を選ぶ．表 2 にそのブール関数を示す．

なお，出力素子をこのブール関数に置き換えたときの精度は 0.899 であり，もとのニューラルネットワークの精度(0.986)よりも低かった．1 層の場合と逆の結果になった．

表 2 ブール関数(畳み込み層 3 層)

数字	ブール関数
0	$x_3 \wedge x_7 \wedge x_{10} \wedge \overline{x_2} \wedge \overline{x_4} \wedge \overline{x_5} \wedge \overline{x_6} \wedge \overline{x_9}$
1	$x_1 \wedge x_6 \wedge x_8 \wedge \overline{x_7} \wedge \overline{x_9} \wedge \overline{x_{10}}$
2	$x_2 \wedge x_4 \wedge x_5 \wedge \overline{x_1} \wedge \overline{x_7} \wedge \overline{x_8} \wedge \overline{x_{10}}$
3	$x_3 \wedge x_5 \wedge \overline{x_2} \wedge \overline{x_4} \wedge \overline{x_6} \wedge \overline{x_8}$
4	$x_4 \wedge x_8 \wedge x_9 \wedge \overline{x_3} \wedge \overline{x_5}$
5	$x_7 \wedge x_{10} \wedge \overline{x_4} \wedge \overline{x_5} \wedge \overline{x_6} \wedge \overline{x_8} \wedge \overline{x_9}$
6	$x_4 \wedge x_{10} \wedge \overline{x_1} \wedge \overline{x_2} \wedge \overline{x_5} \wedge \overline{x_6} \wedge \overline{x_8} \wedge \overline{x_9}$
7	$x_2 \wedge x_3 \wedge x_8 \wedge \overline{x_4} \wedge \overline{x_5} \wedge \overline{x_7}$
8	$x_2 \wedge x_7 \wedge \overline{x_1} \wedge \overline{x_3} \wedge \overline{x_4} \wedge \overline{x_6} \wedge \overline{x_8}$
9	$x_5 \wedge x_7 \wedge x_8 \wedge \overline{x_3} \wedge \overline{x_4} \wedge \overline{x_6}$

畳み込み層 1 層のときと同様にブール関数と図 5 の画像で内部処理を調べてみよう．同じく数字の 3 を見てみよう．数字の 3 のブール関数に肯定で入っている部分は $x_3 \wedge x_5$ である．No.3 と No.5 の画像を見ると，数字の 3 には見えないが類似性はあるように見える．否定の部分は， $\overline{x_2} \wedge \overline{x_4} \wedge \overline{x_6} \wedge \overline{x_8}$ であるが，あまり直感的な理解はできない．

次に，数字の 5 を見てみよう．数字の 5 のブール関数に肯定で入っている部分は， $x_7 \wedge x_{10}$ である．No.7 と No.10 の画像を見ると，数字の 5 には見えないが類似性はあるように見える．否定の部分は $\overline{x_4} \wedge \overline{x_5} \wedge \overline{x_6} \wedge \overline{x_8} \wedge \overline{x_9}$ であるが，あまり直感的な理解はできない．

他の数字に関しても同様であり，畳み込み層が 3 層の場合は，1 層とは違い，特徴(量)はあまり理解するこ

とができない。これは、SmoothGrad が良好に動作していないからであろうと思われる。

7. 終わりに

本稿では、畳み込みニューラルネットワークの内部構造の理解のための一方法を提示した。その方法は、全結合層から抽出したルールと、畳み込み層の出力最大化画像を組み合わせるものであった。畳み込み層が1層の最小構成では、特徴(量)らしきものが確認でき、結果が良好であったが、畳み込み層が3層の場合には、結果はあまり良好ではなかった。今後、改良してゆきたい。なお、全結合層に中間層がある場合は、現在、作業を行なっているので、近いうちに報告したい。

謝辞：プログラミング等で協力していただいた荒木郁哉氏に感謝します。

文 献

- [1]J.Yosinski,etal., Understanding Neural Networks Through Deep Visualization, ICML DL Workshop, 2015.
- [2]R. Andrews, J. Diederich and A. Tickle, Survey and critique of techniques for extracting rules from trained artificial neural networks, Knowledge-Based Systems, Vol.8, No.6, pp. 373-389, 1995.
- [3]T. Hailesilassie, Rule Extraction Algorithm for Deep Neural Networks: A Review, International Journal of Computer Science and Information Security, Vol. 14, No.7,pp.376-391, 2016.
- [4]H. Tsukimoto, Extracting Rules from Trained Neural Networks, IEEE Transactions on Neural Networks, Vol.11, No.2, pp.377-389, 2000.
- [5]月本洋, 下郡信宏, 高島文次郎, 多重線形関数を用いたニューラルネットワークの構造分析, 電子情報通信学会論文誌, Vol.J79-D-II No.7, pp.1271-1279, 1996.
- [6]月本洋, 松本一教, 実戦データマイニング, オーク社, 2018.
- [7]D.Smilkov et al., SmoothGrad: removing noise by adding noise, arXiv:1706.03825, 2017.