CNN 機械の眼はどこまで生物に近づけるか?

工学部計数工学科 4 年 石井悦子

1. キーワード

機械学習, 深層学習, 畳み込み, 畳み込みニューラルネットワーク (CNN), bio-inspired algorithm, VGG, 画像分類, neural style transfer

2. 理論解説

今日の我々の生活に機械学習は欠かせないものになっている. その一種である畳み込みニューラルネットワーク (CNN) はとりわけ画像処理に強い力を発揮する. 今回の展示では, CNN を用いた VGG16(図 1) という有名なネットワークに焦点をあて, CNN の基礎から VGG の性質を活かした応用例まで紹介する.

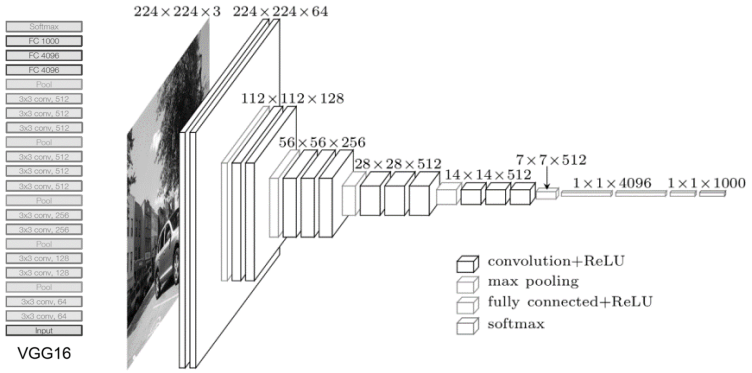


図 1: VGG16 の基本構造と画像認識のイメージ図. VGG16 に入力サイズ $224 \times 224 \times 3$ の画像を与えると, 1000 クラスの内そのクラスに分類される確率の行列 (サイズ $1 \times 1 \times 1000$) を出力する [1].

2.1. 畳み込み (convolution)

そもそも畳み込みとはどのような演算であろうか? 関数 f, g に対する畳み込みは, 「関数 g を平行移動しながら関数 f に重ね足し合わせること」と定義される. Fourier 変換や Laplace 変換と相性が良い演算で, 信号処理や制御工学で広く使用されている. また, 画像処理の文脈での畳み込みとは, 図 2 のような演算を指す. すなわち, 小さい「フィルタ」を画像の上をスライドさせていくことで, 特徴を抽出した新たな画像を得る操作のことである.

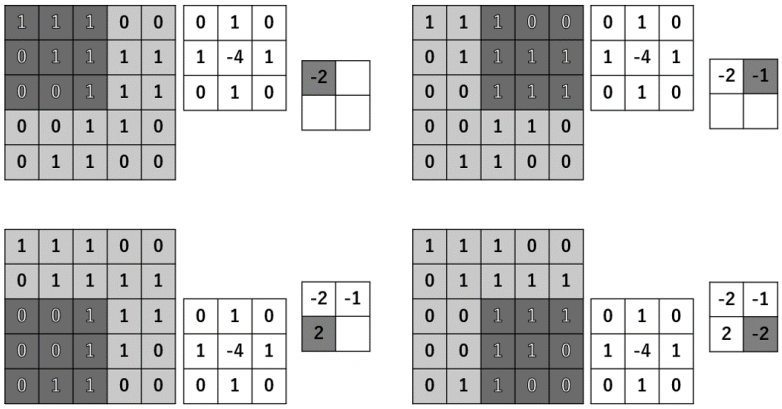
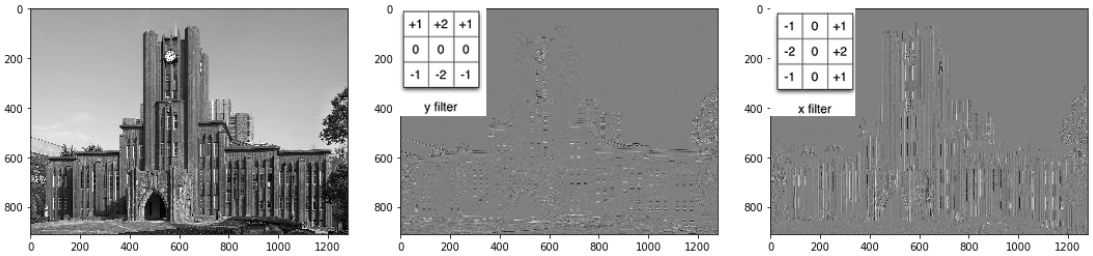


図 2: 画像上での畳み込み演算. 5×5 の画像上に 3×3 のフィルタ (kernel) をかけている. 新たに生成された画像 (ここでは 2×2 の画像) のことを特徴マップ (feature map) と呼ぶ.

フィルタの種類を変えることで, 同じ画像から様々な特徴を抽出できることが知られている. 例えば, Sobel フィルタというフィルタを用いることで, 図 3 のように画像内の輪郭線を抽出することができる (エッジ検出).



(a) 元画像　(b) 横方向検出　(c) 縦方向検出

図 3: Sobel フィルタを用いた画像のエッジ検出.

2.2. CNN の歴史

さて, どうしてこの畳み込み演算を用いたニューラルネットワーク, CNN が考案されたのだろうか? CNN の歴史は 1959 年, ノーベル賞受賞者 Hubel, Wiesel によって, 猫の大脳のニューロン中に「線の傾き」に反応する細胞が発見されたこと [2] に始まる. これは, 生物の脳の領域である第一視覚野で, 前述の「エッジ検出」が行われている, ということを示している. 脳の電気信号の更なる解析により, 人間が第一視覚野において, Gabor フィルタというフィルタを用いて「エッジ検出」を行うことが確かめられた [3]. このように, 生物の物体認識の研究が進むにつれ, 生物の視覚認知の階層的構造が明らかになっていった. CNN は, その階層的構造を再現しようという試み [4], [5] の中で生まれたアルゴリズムである. なお, 現在最も広く使われているものは, 1998 年の LeCun et al.[6] がベースとなっている. 「画像処理で最も強力なアルゴリズム」として紹介されることが多い CNN だが, そもそもの開発の経緯から, 「最も成功した生物模倣アルゴリズム (bio-inspired algorithm)」とも呼べるだろう [7]. 現在では画像処理に留まらず, 自然言語処理など幅広い分野での応用例がみられる.

2.3. 物体認識と VGG16

生物の視覚認知の模倣から始まった CNN は, 画像認識において良い精度を出してきている. Stanford 大学の ImageNet グループは毎年, “Large Scale Visual Recognition Challenge (ILSVRC)” という画像認識の大規模なコンペティションを開催している. 大量の写真にいかに正確なラベル付けができるかを競うもので, 1000 種類のクラスが用意されている. VGG16[8] は ILSVRC2014 で 2 位になったアルゴリズムで, 図 1 に示されるように多層の CNN だ. その他にも AlexNet, ResNet, GoogLeNet といった多層の CNN が開発され, ILSVRC で好成績を収めてきている. 我々人間にとって, 写真に写っているのが自動車であるのか, 馬であるのか区別する, という ILSVRC の課題はたやすい. しかし機械が同じ精度を持つためには様々な困難がある. 同じ物体であっても, 異なる角度から見れば全く違う形状をしていることも多い. 時には, ソファーに挟まれたしっぽだけを見て猫と, 馬の絵の描かれたトラックを自動車の仲間と判定することが要求される. ここで VGG 等が威力を発揮したのは, 畳み込み層を重ねることで, 何種類もの「フィルタ」を画像にかけることが可能になり, それぞれのクラスに属する画像に共通する特徴を特定しやすくなったからである. 実際, どのような特徴が学習されているのか見てみると, 図 4 のように序盤の層では「エッジ検出」等基本的な画像認識の処理が行われていることが確認された. しかし層が進むにつれて, 学習された重みの可視化だけでは解釈が難しくなっている [9].

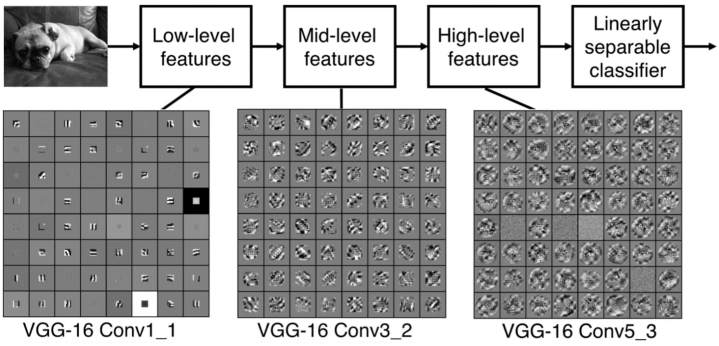


図 4: VGG16 の各層において学習された重みの可視化 [1]. Conv1_1 で学習された重みは, Gabor フィルタを可視化したものと類似している. しかし, Conv3_2, Conv5_3 で学習された重みの解釈は難しい.

2.4. VGG の各層では何が認識されているのか?

多層の CNN の中〜高レベル層で行われた学習を理解するために, 様々なアプローチがなされてきた [9], [10]. 中でも Gatys et al. が 2015 年に発表した neural style transfer というアルゴリズム [11] は高い注目を集めている. VGG はそもそも ILSVRC 用に開発されたアルゴリズムであるから, 層が進むにつれ物体認識により重要な, コンテンツ (content) に関する情報が抽出され, 画風 (style) の情報は失われていく. この特性を定式化し, 最小化問題に書き直すことを考える. すると, 図 5 のように, 入力画像のコンテンツ情報を保持したまま, 画風を変換することができる. 生成画像を $\boldsymbol{x}$, スタイル画像を $\boldsymbol{s}$, コンテンツ画像を $\boldsymbol{c}$, 低レベル層を $\mathcal{S}$, 高レベル層を $\mathcal{C}$ とし, スタイルとコンテンツ間の重みを λ_S とおき, 全体の損失関数 $\mathcal{L}(\boldsymbol{x}, \boldsymbol{c}, \boldsymbol{s})$ をそれぞれの損失関数 $\mathcal{L}_{content}(\boldsymbol{x}, \boldsymbol{c})$, $\mathcal{L}_{style}(\boldsymbol{x}, \boldsymbol{s})$ の和とすれば, 最小化問題は

$$\min_{\boldsymbol{x}} \quad \mathcal{L}(\boldsymbol{x}, \boldsymbol{c}, \boldsymbol{s}) = \mathcal{L}_{content}(\boldsymbol{x}, \boldsymbol{c}) + \lambda_S \mathcal{L}_{style}(\boldsymbol{x}, \boldsymbol{s})$$

と表せる．$\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}}$ を計算することで, 生成画像 $\boldsymbol{x}$ が更新されていく．ただし,

$$\mathcal{L}_{content}(\boldsymbol{x}, \boldsymbol{c}) = \sum_{j \in \mathcal{C}} \frac{1}{n_j} \| f_j(\boldsymbol{x}) - f_j(\boldsymbol{c}) \|_2^2$$

$$\mathcal{L}_{style}(\boldsymbol{x}, \boldsymbol{s}) = \sum_{i \in \mathcal{S}} \frac{1}{n_i} \| \mathcal{G}[f_i(\boldsymbol{x})] - \mathcal{G}[f_i(\boldsymbol{s})] \|_F^2$$

である．ここで, f_l は l 層での活性化関数, $\mathcal{G}$ は Gram 行列という, 特徴マップ間の相関を表す行列である．詳細は [11] を, 実装は後述の GitHub を参照されたい.

図 5: neural style transfer で赤門の画風を変換したもの.

このアルゴリズムを応用し, 白黒画像を着色したり [12], 画像の解像度を上げたり [13] と様々な画像加工技術が提案されている.

3. 実験・実装について

Python3, PyTorch で実装した．GitHub(https://github.com/ettttte/mayfest2018) で公開している.

参考文献

[1] Li, F., Johnson, J., Yeung, S. CS231n: Convolutional Neural Networks for Visual Recognition (2017). [lecture notes and slides]. Retrieved from http://cs231n.stanford.edu/

[2] Hubel, D. H., Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3), 574-591.

[3] Quiroga, R. Q., Reddy, L., Kreiman, G., Koch, C., Fried, I. (2005). Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045), 1102-1107.

[4] Marr, D., Poggio, T. (1976). Cooperative Computation of Stereo Disparity. *Science*, 194(4262), 283-287.

[5] Fukushima, K. (1988). Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2), 119-130.

[6] LeCun, Y., Bottou, L., Bengio, Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

[7] Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org/

[8] Simonyan, K., Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ICLR conference paper*.

[9] Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., Lipson, H. (2015). Understanding Neural Networks Through Deep Visualization. https://arxiv.org/abs/1506.06579

[10] Simonyan, K., Vedaldi, A., Zisserman, A. (2014). Deep Inseide Convolutional Networks: Visualising Image Classification Models and Saliency Maps. https://arxiv.org/pdf/1312.6034.pdf

[11] Gatys, L., Ecker, A., Bethge, M. (2015). A Neural Algorithm of Artistic Style. https://arxiv.org/abs/1508.06576

[12] Zhang, R., Isola, P., Efros, A. (2016). Colorful Image Colorization. https://arxiv.org/abs/1603.08511

[13] Johnson, J., Alahi, A., Li, F. (2016). Perceptual Losses for Real-Time Style Transfer and Super-Resolution. https://arxiv.org/abs/1603.08155