1　勾配消失問題

・DNN の学習過程における各ネットワーク要素 (Node) における誤差消失
→ 誤差逆伝搬法における微分連鎖律を利用したパラメータ学習における勾配消失を招く
→ 結果として DNN の学習が進まなくなる (誤差が消失するので)
→ 勾配消失を対策する方法として以下 3 点の措置をとる

1. 活性化関数の選定
　活性化関数については day1 にて説明されているので詳細割愛
　DNN の中間層に ReLU を用いる事で勾配消失を抑える (事が多い)

2. パラメータ w の初期値設定
　初期値が 0 だと勾配消失問題を招きやすい (活性化関数の出力も 0 に初期化なので)
　→ とりあえず初期値を正規分布乱数で初期化 → 不十分
　→Xavier の初期化 (正規分布乱数を手前 Node 数の平方根で割る ($=\frac{1}{\sqrt{n}}$ を掛け算))
　→He の初期化 (Xavier の式に 2 の平方根をかける ($=\sqrt{\frac{2}{n}}$ を掛け算))

　手前 Node 数の平方根で割る意味は正規分布を中央に適当に寄せるため (と理解する)

　Xavier の初期化は活性化関数が sigmoid の場合によく用いる
　He の初期化は活性化関数が ReLU の場合によく用いる

　それぞれ活性化関数の初期出力分布がザックリと中間値中心に均された分布へ誘導の効果
　なので学習開始早々に誤差勾配をつかみやすく学習の進み具合に優れる (と理解しておく)

3. バッチ正規化

　学習用データをランダムに等分割して学習する方法 → ミニバッチ勾配降下法
　ミニバッチ単位で誤差を正規化するのが (ミニ) バッチ正規化
　学習が速くなるとも言われている

2 学習率最適化手法 (Optimizer)

・ものすごくザックリといえば「残差の大きさに対して学習率を動的に調整する」様な方法
・学習が高速化、局所解に嵌まりにくくなる、等のメリット
・以下の様な手法がある

1. モメンタム

現在のパラメータより一つ前のパラメータを減算した値に慣性係数 μ を掛け算した値を従来の勾配降下法における誤差とともに加える方法

$V_t = \mu V_{t-1} - \epsilon \nabla E$

$w^{(t+1)} = w^{(t)} + V_t$

物理法則でいうところの慣性より命名

慣性係数 μ はハイパーパラメータとなる (適切に調整しないと学習が収束し難い)

2.AdaGrad

誤差の微分値をハイパーパラメータ h に加算し続ける様な方法

結果として学習率が徐々に小さくなる効果

$h_0 = \theta$ (θ はハイパーパラメータ)

$h_t = h_{t-1} + (\nabla E)^2$

$w^{(t+1)} = w^{(t)} - \epsilon \frac{1}{\sqrt{h_t + \theta}} \nabla E$

欠点は鞍点問題を引き起こしやすい事 (=ある極点で停滞しやすくなる)

3.RMSProp

前回までの誤差と今回の誤差を一定割合で加算して学習率を調整する方法

$h_t = \alpha h_{t-1} + (1 - \alpha)(\nabla E)^2$

$w^{(t+1)} = w^{(t)} - \epsilon \frac{1}{\sqrt{h_t + \theta}} \nabla E$

α と θ はハイパーパラメータ

AdaGrad よりは鞍点問題を起こしにくい (と言っているが局所解に嵌まる可能性は残る)

4.Adam

モメンタムと RMSProp の良いところを取り入れ欠点を解消したと説明

単純に以下式のように組み合わせされている

$w^{(t+1)} = w^{(t)} - \epsilon \frac{V_t}{\sqrt{h_t + \theta}} \nabla E$

3　過学習

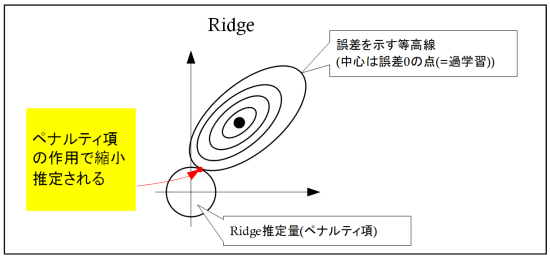
・モデルが訓練データに過剰に fit しすぎて未知データへの対応ができない状態
　(機械学習でも同じ問題を含むのは言うまでもない)

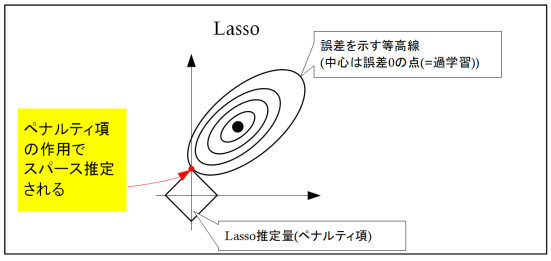
・機械学習同様に正則化項を導入して対策する
　DNN におけるパラメータ w の値が、学習が進むにつれて更新される
　パラメータ w が過剰に大きくなることが過学習につながると言われている
　→ 誤差関数に対して正則化項を導入して過学習しない様に抑制する

　　正則化項はパラメータ w の大きさ (距離)
　　$\|W^{(1)}\|_p = (|W_1^{(1)}|^p + \ldots + |W_n^{(1)}|^p)^{\frac{1}{p}}$
　　$\|W^{(2)}\|_p = (|W_1^{(2)}|^p + \ldots + |W_n^{(2)}|^p)^{\frac{1}{p}}$
　　$\|x\|_p = \|W^{(1)}\|_p + \|W^{(2)}\|_p$
　　誤差関数 + 正則化項は
　　$E_n(w) + \frac{1}{p}\lambda\|x\|_p$
　　p は 1 または 2(以下となる)

　1.L1 正則化 (p=1)
　　Lasso ともいう、正則化項が 1 次 (の絶対値) となる、マンハッタン距離
　　スパース性 (無関係なパラメータを 0 にする) に寄与

　2.L2 正則化 (p=2)
　　Ridge ともいう、正則化項が 2 次となる、ユークリッド距離
　　縮小性 (全パラメータを生かしつつペナルティを課す)





・ドロップアウト

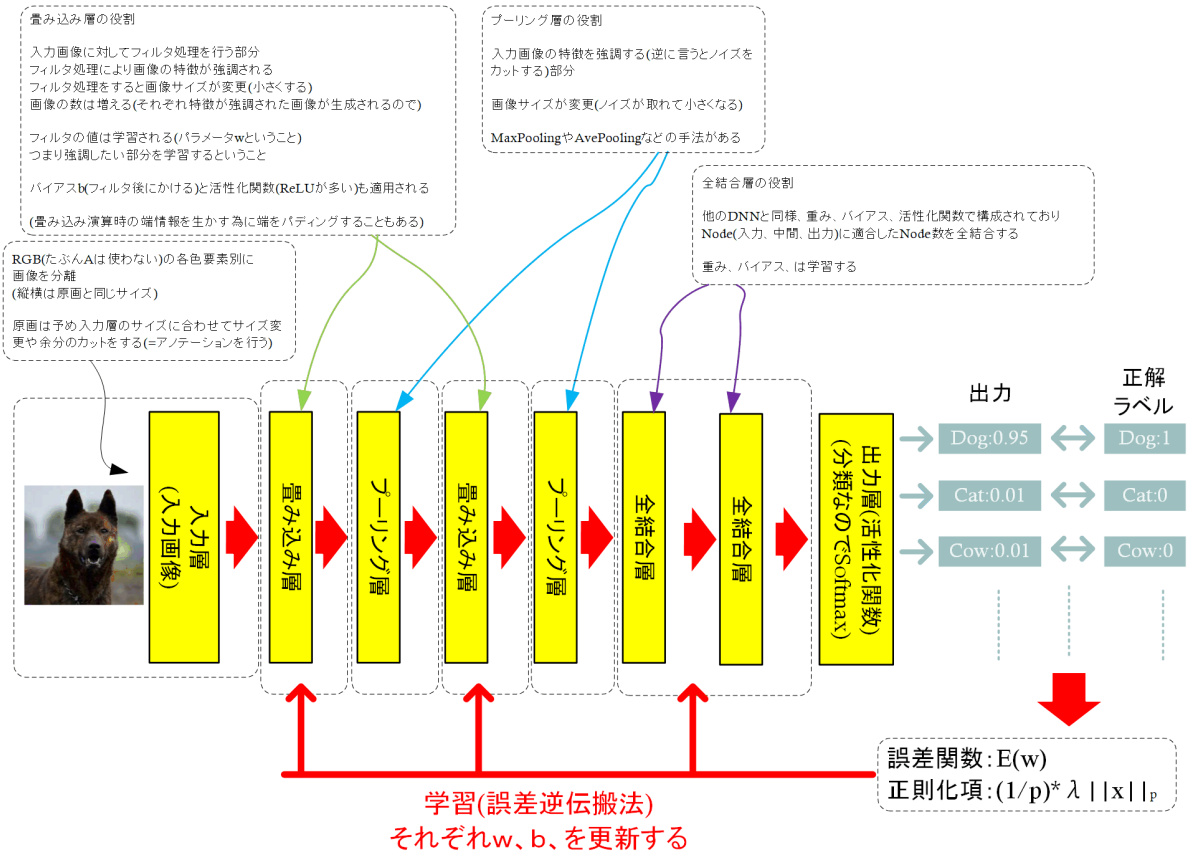
学習の過程にてランダムに Node を遮断する方法

結果的に訓練データの量を増やす様な効果が生まれ、過学習対策に寄与する

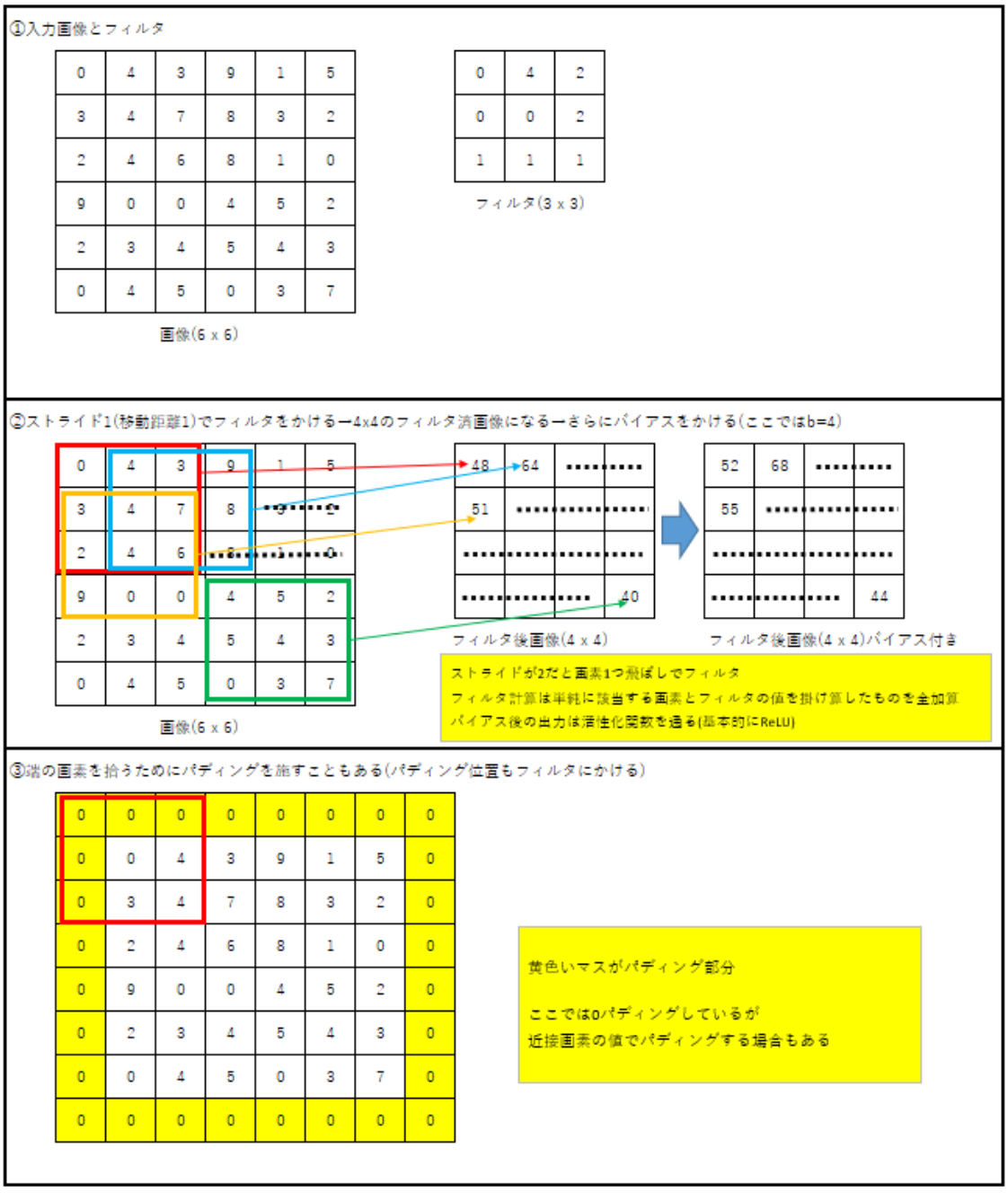
4　CNN(畳み込みニューラルネットワーク)

・画像データの分類に良く利用されるネットワーク
・一次元 (音声)、二次元 (静止画)、三次元 (動画)、の次元間関連を持つデータの処理に強い
・畳み込み層、プーリング層、で構成される中間層を持つ
・終段は全結合層

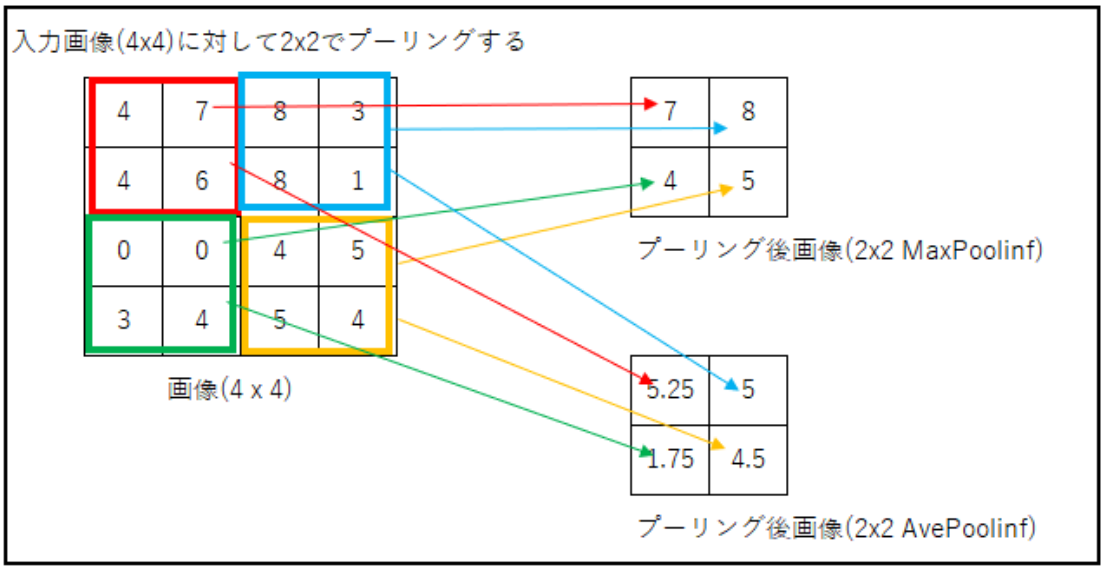
CNN の全体像



畳み込み処理説明


①入力画像とフィルタ
0 4 3 9 1 5
3 4 7 8 3 2
2 4 6 8 1 0
9 0 0 4 5 2
2 3 4 5 4 3
0 4 5 0 3 7
画像(6 x 6)
0 4 2
0 0 2
1 1 1
フィルタ(3 x 3)
②ストライド1(移動距離1)でフィルタをかける→4x4のフィルタ済画像になる→さらにバイアスをかける(ここではb=4)
48 64
51
40
フィルタ後画像(4 x 4)
52 68
55
44
フィルタ後画像(4 x 4)バイアス付き
画像(6 x 6)
ストライドが2だと画素1つ飛ばしでフィルタ
フィルタ計算は単純に該当する画素とフィルタの値を掛け算したものを全加算
バイアス後の出力は活性化関数を通る(基本的にReLU)
③端の画素を拾うためにパディングを施すこともある(パディング位置もフィルタにかける)
0 0 0 0 0 0 0 0
0 0 4 3 9 1 5 0
0 3 4 7 8 3 2 0
0 2 4 6 8 1 0 0
0 9 0 0 4 5 2 0
0 2 3 4 5 4 3 0
0 0 4 5 0 3 7 0
0 0 0 0 0 0 0 0
黄色いマスがパディング部分
ここではoパディングしているが
近接画素の値でパディングする場合もある

プーリング説明



・MaxPooling は対象の全画素中から最も大きい値を採用する
・AvgPooling は対象の全画素の平均値を採用する

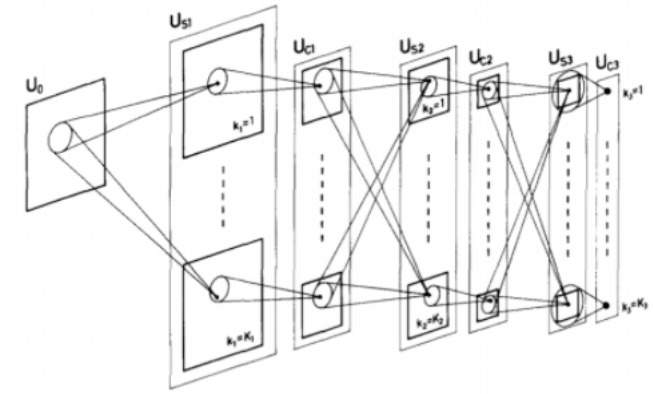
5　CNN 界隈トレンド

講義では AlexNet の紹介がされているが、どちらかというとその中の
GlobalMaxPooling や GlobalAvgPooling の紹介の方が肝と言いたいように感じた
という事で本レポートにて CNN 界隈のトレンド変遷を探してみた (主に ILSVRC をトラッキング)

参考文献 (図面引用も)

https://qiita.com/yu4u/items/7e93c454c9410c4b5427
https://keno-edu-lasalle.hatenablog.com/entry/2017/11/15/175512
https://aizine.ai/glossary-vgg/
https://www.kumilog.net/entry/resnet-paper/

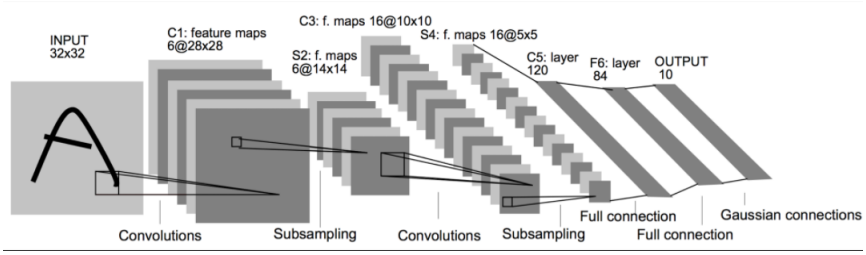
・Neocognitron(1982)
　日本の福島氏が提案したニューラルネットワーク
　単純細胞 (畳み込み層) と複雑細胞 (プーリング層) を組み合わせ、
　のちの CNN の着想につながる。



・LeNet(1989)

Yann LeCun らによって提案された。

誤差逆伝搬法を初めて導入したニューラルネットワーク (初期の CNN と言われる)



・AlexNet(2012)

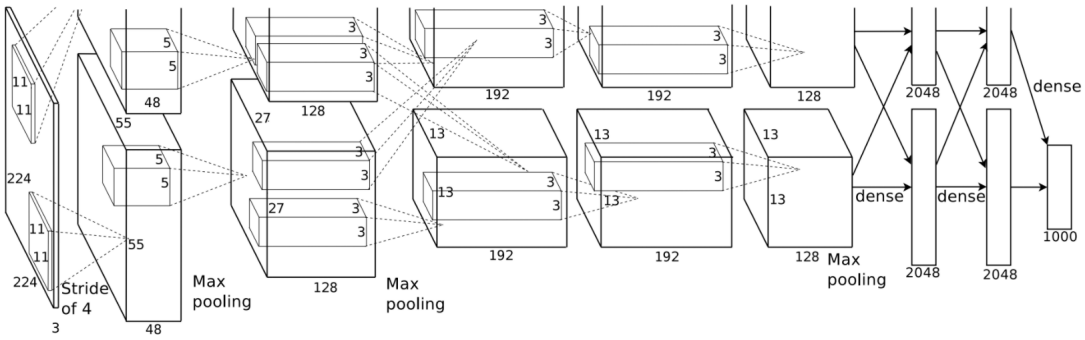
Alex Krizhevsky、 I. Sutskever、 G. E. Hinton により設計されたモデル

ILSVRC 2012 で優勝し、深層学習の有効性を示したモデルでもある (8 層)

ReLU やドロップアウトを採用しているのも特徴、

また 2 台の GUP を用いての独立した学習をする

(2 台の GPU を結合する層 (出力層を含む) を有する)。



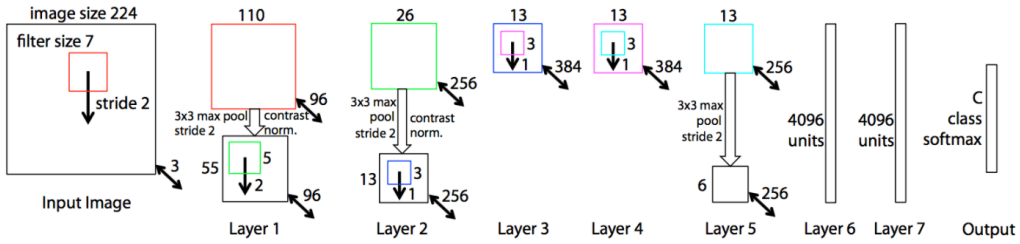
・ZFNet(2013)

M. D. Zeiler と R. Fergus が提唱

ILSVRC 2013 の優勝モデル (8 層)

AlexNet の 2 つの欠点改良を行ったもので、基本構成は AlexNet と同じ

こちらは GPU 1 台で学習 (モデルも AlexNet の 2ch を 1 つにまとめたような構成)



・GoogLeNet(2014)

名前の由来は Google 社開発のネットワークで LeNet に敬意を評してとの事

ILSVRC 2014 で優勝したモデル (22 層)

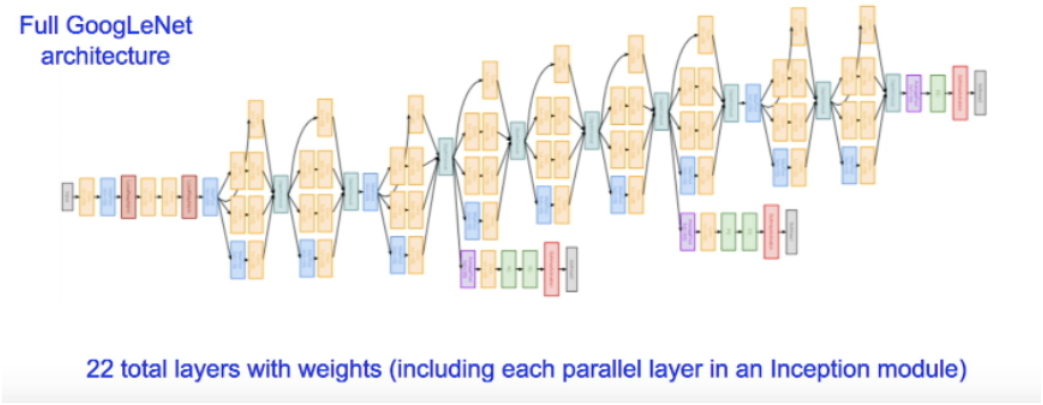
Inception モジュールの導入による次元数やパラメータの削減 (スパース化)

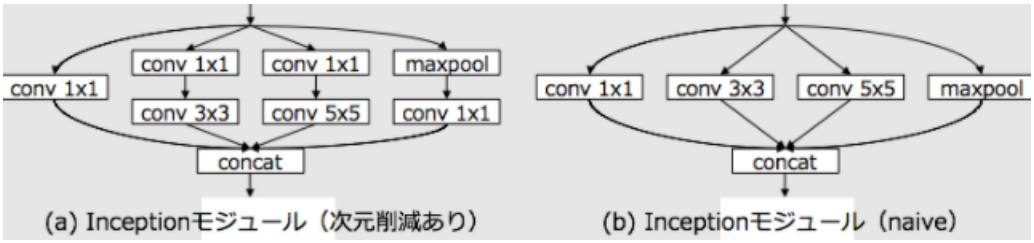
GlobalAveragePooling の導入によるパラメータの削減や過学習抑制効果

(またそれによる全結合層の削除)

Auxiliary Loss を導入してサブネットワークでもクラス分類を行う

(これによるネットワーク中間層の勾配消失対策と正則化も実現する)



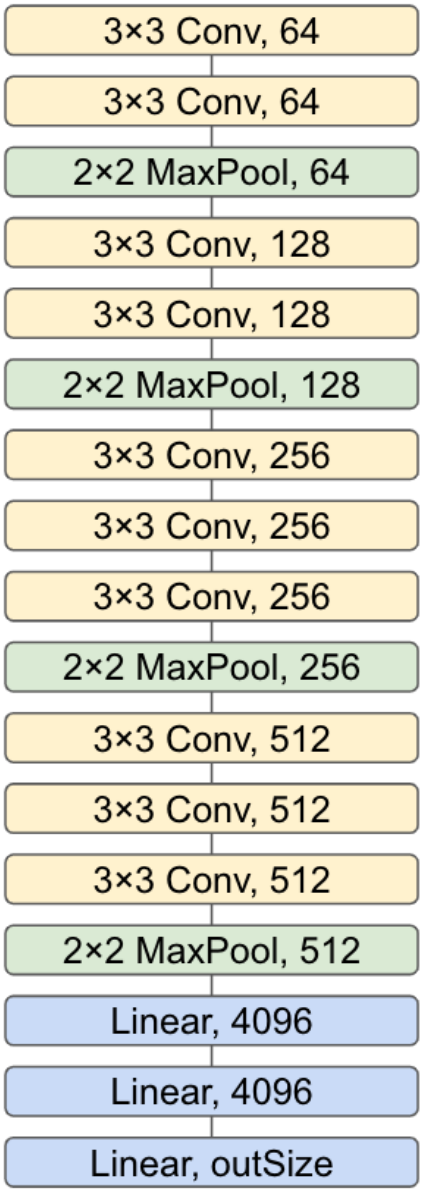


・VGGNet(2014)

ILSVRC 2014 で二位のモデル (16 層と 19 層)

3x3 の畳み込み層を多用してチャネル数を増やしているのが特徴

GoogLeNet とともに後の ResNet の着想の元となったとのこと



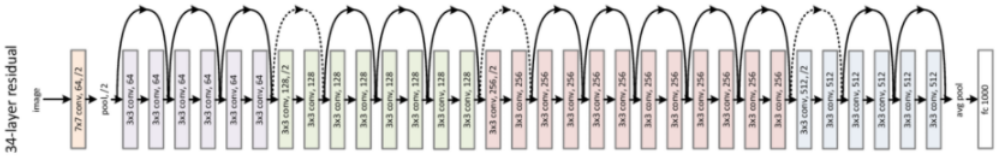
・ResNet(2015)

ILSVRC 2015 で優勝のモデル (34 層)

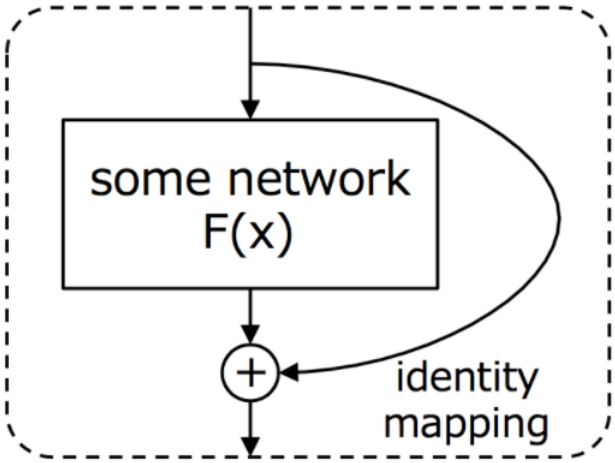
多層 DNN に起こる勾配消失問題を Residual モジュールの導入で解決

(これは誤差逆伝搬をモジュール入出力ショートカットする事で学習を行う)

バッチ正則化や He の初期化を導入したモデルでもある



Residualモジュール



・SENet(2017)

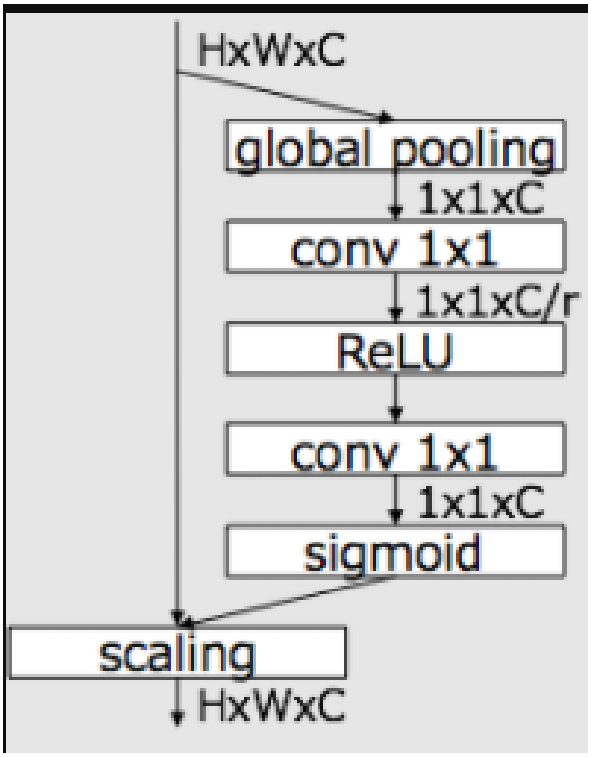
ILSVRC 2017 で優勝のモデル

Squeeze-and-Excitation Block (SE Block) を導入して Attention を実現したとある

Attention は Google の BERT(自然言語処理モデル) 等でも使われている機構

(本教育の Stage4 コンテンツで取り上げられるはずなのでここでは調査しない)

Attention によりチャネル毎に適応的重み付けする特徴マップを実現との事



6 実習

・本レポートと同じレポジトリにそれぞれ課題を実装したコードを置く

2_1_network_modified.ipynb
2_2_1_vanishing_gradient.ipynb
　実行のみ (結果は提出コードの Log を参照のこと)

2_2_2_vanishing_gradient_modified.ipynb
2_3_batch_normalization.ipynb
2_4_optimizer.ipynb
2_5_overfiting.ipynb
2_6_simple_convolution_network.ipynb
2_7_double_comvolution_network.ipynb
　”Try”に対応変更し実行結果を残した (結果は提出コードの Log を参照のこと)

2_8_deep_convolution_net.ipynb
　実行のみ (結果は提出コードの Log を参照のこと)