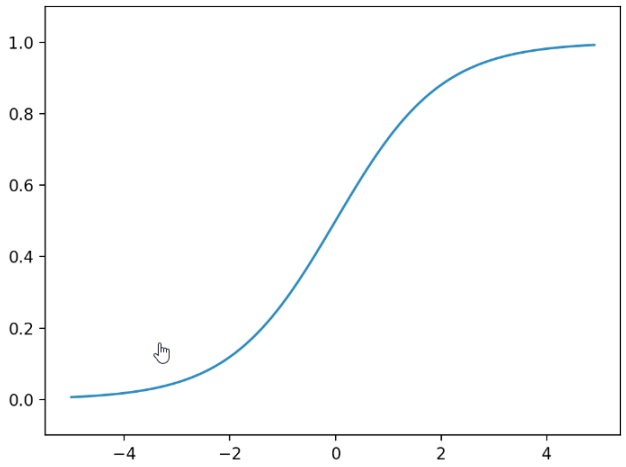
1　Section1_勾配消失問題

勾配消失問題とは、逆伝播の過程で誤差の勾配値が消失してしまい、入力層付近での学習が進まなくなってしまう、深層学習において生じやすい問題のこと。

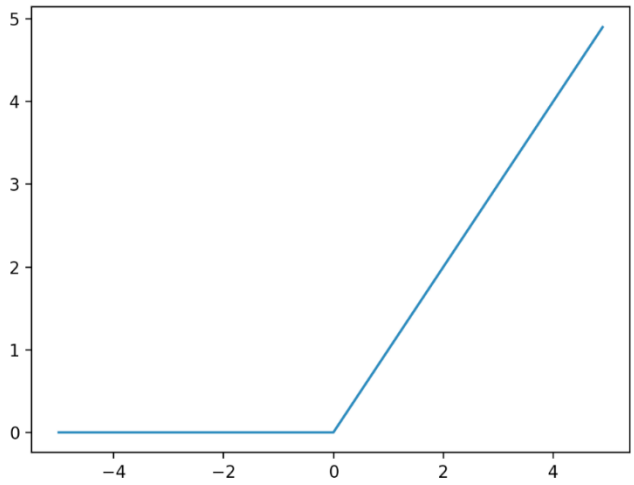
層が深いほど起こりやすい。
勾配消失問題は、活性化関数が何度も作用することにより勾配が小さくなりすぎてしまうことが原因とされているので、ReLU 関数のような「正規化機能を持たない」活性化関数を中間層で用いるなどにより、ある程度回避することが期待できる。

（活性化関数例）

シグモイド関数



ReLU 関数



2　Section2_学習率最適化手法

過学習とは訓練誤差が小さいにもかかわらず、汎化誤差が小さくならない状態のことを指す。
過学習を抑制するための手法として、モメンタムなどの正規化手法が用いられる。
また、ドロップアウトという重みを更新する枝を一定の割合でランダムに無効化する手法もある。

（1）ドロップアウト

訓練時ユニットをランダムに消去しながらパラメータの最適化を行う。
ユニットを消去する割合は、定数として与える。

（2）正則化

正則化はパラメータに何らかの制約を加える方法の総称であり、過剰適合を緩和させる目的でモデルに取り入れられる。

①正則化の一つの方法として、損失関数にペナルティ項を加えることがある。ペナルティ項には L2 ノルムがよく用いられるが、L1 ノルムも用いられることがある。
②ニューラルネットワークにおいては、一般に各層の重みにペナルティを課し、各層のバイアスにはペナルティを課さ荷ことが多い。
③ニューラルネットワークにおいて、畳み込み層のパラメータは、入力された特徴マップの複数個所と結合される。
これをパラメータ共有という。パラメータ共有は、パラメータの自由度を制限しているという点で、正則化の一種であると解釈できる。

（3）その他の学習率最適化手法

①データ拡張
②早期終了
③バッチ正規化
④レイヤー正規化
⑤インスタンス正規化

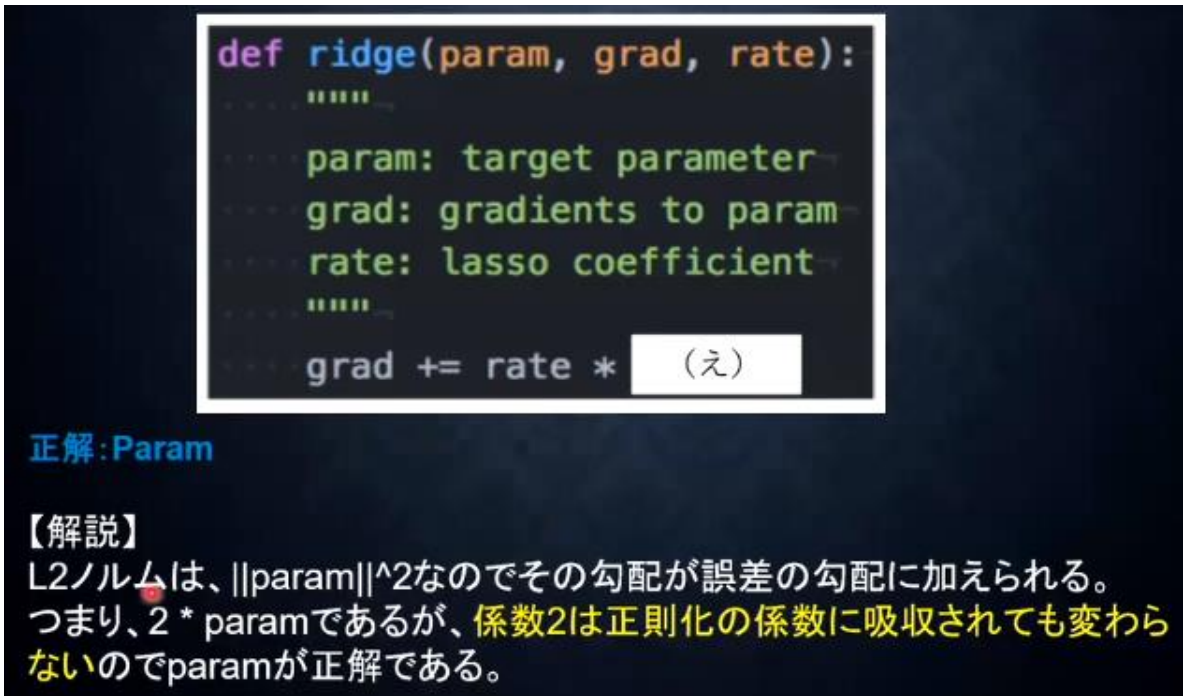
3　Section3_過学習

問題 1

深層学習において、過学習の抑制・汎化性能の向上のために正則化が用いられる。そのひとつに、L2 ノルム正則化（Ridge, Weigh Decay）がある。以下は L2 正則化を適用した場合に、パラメータの更新を行うプログラムである。あるパラメータ param と正則化がないときにそのパラメータに伝播される誤差の勾配 grad が与えられたとする。
最終的な勾配を計算する（え）にあはてはまるのはどれか。ただし rate は L2 正則化の係数を表すとする。

```
def ridge(param, grad, rate):
    """
    param: target parameter
    grad: gradients to param
    rate: ridge coefficient
    """
    grad += rate * （え）
```

(1) np.sum(param**2)
(2) np.sum(param)
(3) param**2
(4) param

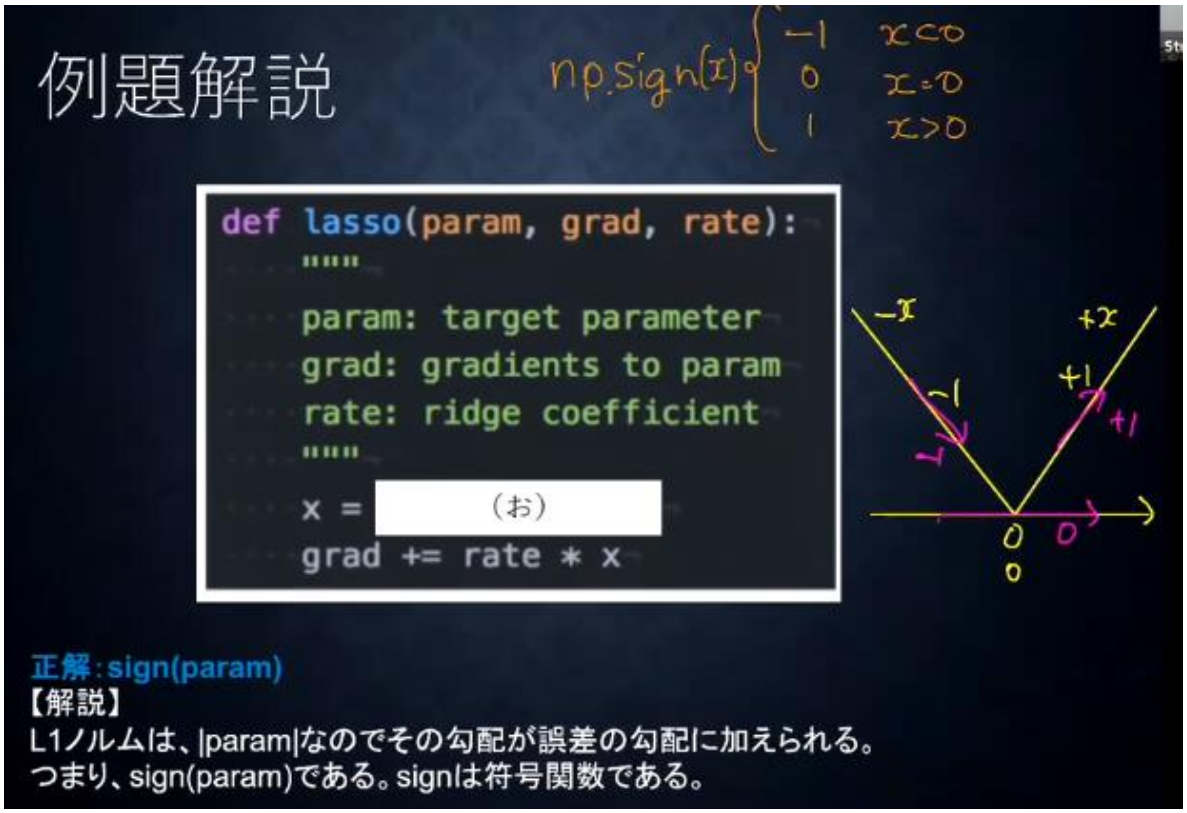


問題２

以下はL1ノルム正則化（Lasso）を適用した場合に、パラメータの更新を行うプログラムである。あるパラメータparamと正則化がないときにそのパラメータに伝播される誤差の勾配gradが与えられたとする。
最終的な勾配を計算する（お）にあはてはまるのはどれか。ただしrateはL1正則化の係数を表すとする。

```
def lasso(param, grad, rate):
    """
    param: target parameter
    grad: gradients to param
    rate: lasso coefficient
    """
    x = （お）
    grad += rate * x
```

(1) np.maximum(param, 0)
(2) np.minimum(param, 0)
(3) np.sign(param)
(4) np.abs(param)



問題３

画像認識などにおいて、精度向上や汎化性能の向上のためにデータ拡張が行われることが多い。データ拡張には、画像を回転・反転させるなど様々な種類がある。以下は画像をランダムに切り取る処理を行うプログラムである。これは画像中の物体の位置を移動させるなどの意味がある。
（か）にはてはまるのはどれか。

```
def random_crop(image, crop_size):
    """
    image: (height, width, channel)
    crop_size: (crop_height, crop_width)

    height >= crop_height, width >= crop_width
    """
    h, w, _ = image.shape
    crop_h, crop_w = crop_size

    # 切り取る位置をランダムに決める
    top = np.random.randint(0, h - crop_h)
    left = np.random.randint(0, w - crop_w)
    bottom = top + crop_h
    right = left + crop_w

    image = （か）
    return image
```

(1) image[:, top:bottom, left:right]
(2) image[:, bottom:top, right:left]
(3) image[bottom:top, right:left, :]
(4) image[top:bottom, left:right, :]

4　Section4_畳み込みニューラルネットワーク

（1）畳み込み処理

```
# レイヤの生成
self.layers = OrderedDict()
self.layers['Conv1'] = layers.Convolution(self.params['W1'], self.params['b1'], conv_param['stride'],
                                          conv_param['pad'])
self.layers['Relu1'] = layers.Relu()
self.layers['Pool1'] = layers.Pooling(pool_h=2, pool_w=2, stride=2)
self.layers['Affine1'] = layers.Affine(self.params['W2'], self.params['b2'])
self.layers['Relu2'] = layers.Relu()
self.layers['Affine2'] = layers.Affine(self.params['W3'], self.params['b3'])
```

```
========== input_data ===========
 [[[[ 4. 61. 35. 29.]
   [63. 38. 23. 53.]
   [32. 39. 12. 94.]
   [78. 47. 38. 29.]]]


 [[[31. 60. 57. 74.]
   [29. 84. 16.  9.]
   [57. 77. 10.  8.]
   [11. 49. 67. 60.]]]]
============================
============ col =============
 [[ 4. 61. 35. 63. 38. 23. 32. 39. 12.]
 [61. 35. 29. 38. 23. 53. 39. 12. 94.]
 [63. 38. 23. 32. 39. 12. 78. 47. 38.]
 [38. 23. 53. 39. 12. 94. 47. 38. 29.]
 [31. 60. 57. 29. 84. 16. 57. 77. 10.]
 [60. 57. 74. 84. 16.  9. 77. 10.  8.]
 [29. 84. 16. 57. 77. 10. 11. 49. 67.]
 [84. 16.  9. 77. 10.  8. 49. 67. 60.]]
============================
```

im2col()

5　Section5_最新の CNN

（1）MobileNet

近年さまざまな畳み込みが提案されている。MobileNet ではスマートフォンのような計算資源が限られた環境下で使用するために、通常の畳み込みの代わりにデプスワイズ畳み込みとポイントワイズ畳み込みの２種類の畳み込みを採用している。

畳み込みの計算量はパラメータ数に依存するため、それぞれの畳み込み層が持つ学習可能パラメータ数を計算することで計算量の改善量を見積もることが可能になる。フィルタサイズが縦・横どちらも f 、入力チャネル数が Cin、出力チャネル数が Cout の時に

　CinCoutf**2

となる。

デプスワイズ畳み込みはチャネル毎に独立のフィルタを用いて畳み込みを行うため

　Cinf**2 となり通常の畳み込み

　1/Cout 倍となる。

ポイントワイズ畳み込みはフィルタサイズを１とした畳み込み層のことで、1/f**2 倍となる。

セマンティックセグメンテーションのように広い領域を参照しなければならないタスクの場合計算量が倍増してしまう。

そのためダイレイト畳み込みを使用する。また、生成モデルなどで小さな画像から大きな画像を生成する際に使う転置畳み込みなどがある。

（2）R-CNN

物体検出の代表的なモデル

R-CNN では、まず物体が存在するであろう候補画像領域を画像中から求める。その際選択的探索を用いる。

まず約２０００個の候補領域を選出し、その１つひとつを CNN に入力することで特徴マップを取得し、その特徴マップと教師を用いて SVM により分類した後、バウンディングボックスの回帰を行う。

候補領域ごとに CNN の順伝播計算を行う必要があり、検出速度の低下が有った。これを改造したのが Fast R-CNN である。

候補領域の提案をニューラルネットワークにより実現し、選択的探索を排除したモデルが Faster R-CNN である。