知能プログラミング演習 I 演習課題

1 準備

1.1 自前の環境の場合

- Moodle から課題を各自任意のフォルダにダウンロードし, 展開したフォルダの中に以下のものがすべて入っていることを確認
  - CNN1.py
  - P64.npy
  - L256.npy
  - task.pdf

1.2 CSE の場合

- まだ演習用のフォルダを作っていない人は DLL のフォルダを作成
  - ホームディレクトリに演習用のディレクトリを作成

    step1: `mkdir -p DLL`
- 作業ディレクトリ DLL に移動

  step1: `cd ./DLL`
- 今日の課題を DLL にダウンロードして展開

  step1: `wget http://www-als.ics.nitech.ac.jp/~karasuyama/DLL20/Lec6.zip`

  step2: `unzip Lec6.zip`
  - 展開したフォルダの中に, 以下のものがすべて入っていることを確認
    - CNN1.py
    - P64.npy
    - L256.npy
    - task.pdf
- Lec6 へ移動

  step1: `cd ./Lec6`

2 課題

畳み込みニューラルネットワークの順伝播を実装する．ただし, CNN1.py にコードを保存すること．実行用の例として読み出されている画像はグレースケール画像（チャネル数 $K = 1$）である．

1. 畳み込み層の順伝播計算を行う関数 Convolution を完成させよ[*1]．呼び出し元はすでに記載されているので，まず与えられている引数が何を表現しているか確認すること（例えば，X や V0）．活性化関数は ReLU とし，定数項はなしとする．ヒント: 3 次元以上の配列もこれまで同様の方法で部分配列にアクセスできる

```
In [1]: A = np.zeros((4,4,3))

In [2]: A  # 4x4x3 の配列
Out[2]:
array([[[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]]])

In [3]: B = np.ones((2,2,3))  # 2x2x3 の配列を作成

In [4]: A[1:3,1:3,:] = B  # A の部分配列に B を代入
                          # "1:3"の指定で添字 1 から 2 が指定される
In [5]: A[:,:,0]  # 結果を確認．3 次元目に 0 を指定
Out[5]:
array([[0., 0., 0., 0.],
       [0., 1., 1., 0.],
       [0., 1., 1., 0.],
       [0., 0., 0., 0.]])

In [6]: A[:,:,1]   # 3 次元目に 1 を指定
Out[6]:
array([[0., 0., 0., 0.],
       [0., 1., 1., 0.],
       [0., 1., 1., 0.],
       [0., 0., 0., 0.]])
```

[*1] CNN1.py では畳み込みのスライド移動を単純に for 文で実現するが，python では for 文は行列演算に比べて非常に遅い．そのため，実際には畳み込み演算を行列の掛け算で書けるように変換して実行することが多い．

```
In [7]: A[:,:,2]   # 3次元目に2を指定
Out[7]:
array([[0., 0., 0., 0.],
       [0., 1., 1., 0.],
       [0., 1., 1., 0.],
       [0., 0., 0., 0.]])
```

2. Max プーリングの順伝播計算を行う関数 `MaxPooling` を完成させよ．
3. 以下の六つのフィルタを組み合わせて，Convolution→MaxPooling→Convolution→MaxPooling のように畳み込み演算を 2 回繰り返して適用することを考える．

$$F_{\text{sobel\_x}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, F_{\text{sobel\_y}} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, F_{\text{laplacian}} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix},$$
$$F_{\text{smooth}} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, F_{\text{sharpen}} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}, F_{\text{zero}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

単なる零行列である F_{zero} 以外は，それぞれ画像処理でよく利用されるフィルタであり，$F_{\text{sobel\_x}}$ は 1 次微分によるエッジ検出 (横方向)，$F_{\text{sobel\_y}}$ は 1 次微分によるエッジ検出 (縦方向)，$F_{\text{laplacian}}$ は 2 次微分によるエッジ検出 (8 方向)，F_{smooth} は平滑化 (ぼかし)，F_{sharpen} はエッジ強調の効果があるとされる．これらを使ってまず，最初の Convolution 層で用いる $V^{(0)}$ を以下のように $3 \times 3 \times 1 \times 3$ 配列として定義する．

$$V^{(0)}_{:,:,1,1} = F_{\text{sobel\_y}}, \quad V^{(0)}_{:,:,1,2} = F_{\text{sobel\_x}}, \quad V^{(0)}_{:,:,1,3} = F_{\text{smooth}}$$

つまり，3×3 でチャネル数 $K = 1$ のフィルタが $M = 3$ 存在する状態である．ただし，ここでは $V^{(0)}_{:,:,k,m}$ の表記は $V^{(0)}_{pqkm}$ の k と m を固定して，添字 p と q（それぞれ 1 から H）の行と列に対応する行列とする．次に，2 回目の Convolution 層で用いる $V^{(1)}$ を以下のように $3 \times 3 \times 3 \times 3$ 配列として定義する．

1 つ目の $3 \times 3 \times 3$ フィルタ: $V^{(1)}_{:,:,1,1} = F_{\text{sharpen}}, \quad V^{(1)}_{:,:,2,1} = F_{\text{sharpen}}, \quad V^{(1)}_{:,:,3,1} = F_{\text{zero}}$
2 つ目の $3 \times 3 \times 3$ フィルタ: $V^{(1)}_{:,:,1,2} = F_{\text{smooth}}, \quad V^{(1)}_{:,:,2,2} = F_{\text{smooth}}, \quad V^{(1)}_{:,:,3,2} = F_{\text{zero}}$
3 つ目の $3 \times 3 \times 3$ フィルタ: $V^{(1)}_{:,:,1,3} = F_{\text{zero}}, \quad V^{(1)}_{:,:,2,3} = F_{\text{zero}}, \quad V^{(1)}_{:,:,3,3} = F_{\text{laplacian}}$

こちらでは，1 層目の結果として，入力が 3 チャンネルになるため，フィルタも $3 \times 3 \times 3$ で定義する．また，ここでもフィルタ数は $M = 3$ としている．パディングやストライドなどの設定は表 1 の通りとする．また，図 1 にこのネットワークの模式図を示す．数式内の下付き添字は 1 から始まるが，python の array の添字は 0 からなので，混同しないよう注意すること． 1 回目の Convolution→MaxPlooing までは記載済みなので，これを参考に 2 回目の Convolution→MaxPlooing を作成せよ．作成したら，`plot_result` 変数を `True` にして実行し，cnn.pdf に保存される特徴マップの変化過程とそれぞれのフィルタの意味の対応を確認すること（注 1: ここでは可視化のために，輝度値を関数 `scale` で正規化をしている）（注 2: 今回は padding は全て 0 で埋めていて，これは縁を暗くすることに相当するが，これが不自然な結果をもたらすこともあるため周辺の値から適応的に決めることも多い）．

表 1　各層におけるパラメータの設定.

	フィルタサイズ H	フィルタ数 M	チャネル数 K	パディング数	ストライド数
1 回目の畳み込み層	3	3	1	1	1
1 回目のプーリング層	2	-	-	1	2
2 回目の畳み込み層	3	3	3	1	1
2 回目のプーリング層	2	-	-	1	2

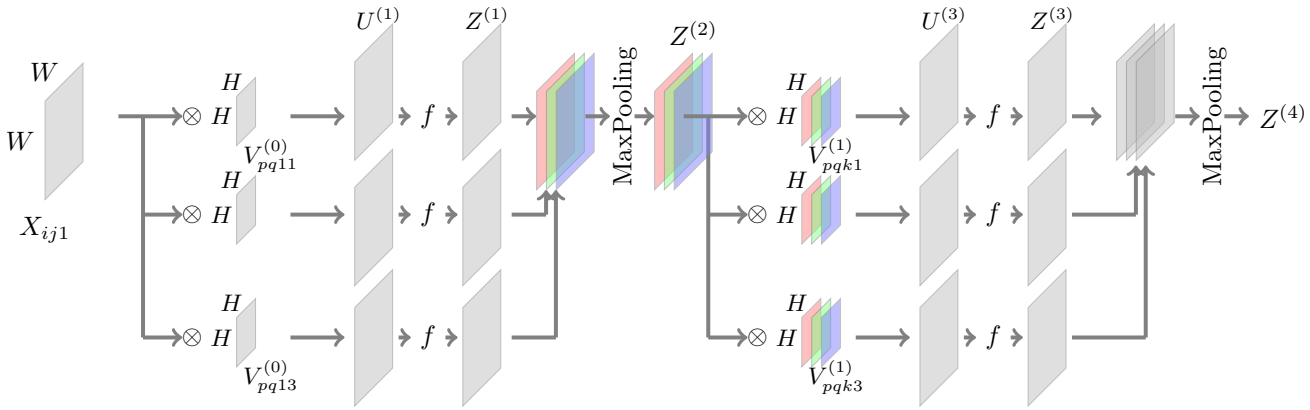

図 1　3. で作成する CNN の構造．入力 X_{ij1} はグレースケールの 1 チャネルのみ (i, j は 1 から W)．ここでは，Convolution(1 回目) , MaxPlooing(1 回目), Convolution(2 回目) , MaxPlooing(2 回目) の適用の順に $Z^{(1)}$, $Z^{(2)}$, $Z^{(3)}$, $Z^{(4)}$ と出力が作られていくとする．畳み込み層の出力 $Z^{(1)}$ と $Z^{(3)}$ において，活性化関数適用前の値を $U^{(1)}$ と $U^{(3)}$ として表現している．1 回目の畳み込みで，三つのフィルタを適用したことにより，2 回目の畳み込みでは入力は 3 チャネル存在することになる．3 チャネルの入力に対して 3 チャネルのフィルタ $V^{(1)}$ を三つ適用していることに注意せよ.

3 課題の提出

Moodle を使ってファイルを提出してください. 提出方法は以下の通りです.

- Moodle にログインし, 知能プログラミング演習のページへ移動.
- Lec6 の項目に, CNN1.py をアップロードする.

9/4(金) の 17:00 を提出期限とします.