Εμρηκα

神經網路的理論架構分析及程式實作

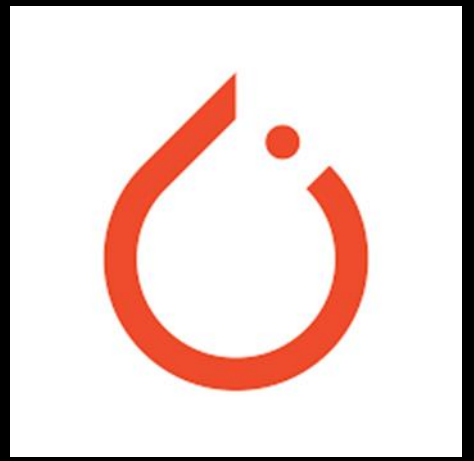
-自製Python神經網路套件

葉適穎
廖士樘
李昶毅
指導老師：彭天健 老師

目錄

- 研究動機及目的
- 架構及原理介紹
- 活化函數
- 機器學習
- 套件成果

研究動機



研究目的

- 研究神經網路的基礎架構(如FCN及CNN)並轉換成簡單的運算邏輯
- 在不使用任何神經網路相關套件(如上面這些)的情況下寫出可以使用的神經網路
- 撰寫可以使用的Python神經網路套件

神經網路的結構及原理

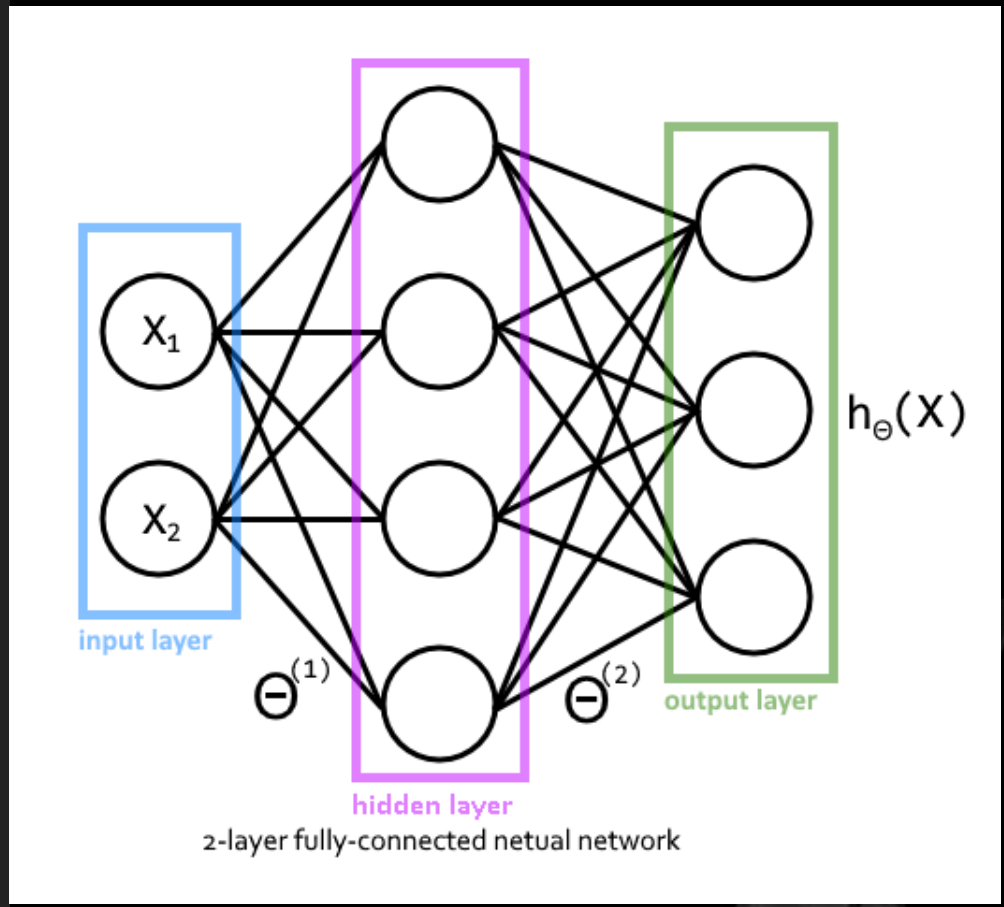
FCN

- 右側圖片的第一層(藍到紫)為:

$$x_1 w_1 + x_2 w_2 + b_1 = y_1$$
$$x_1 w_3 + x_2 w_4 + b_2 = y_2$$
$$x_1 w_5 + x_2 w_6 + b_3 = y_3$$
$$x_1 w_7 + x_2 w_8 + b_4 = y_4$$

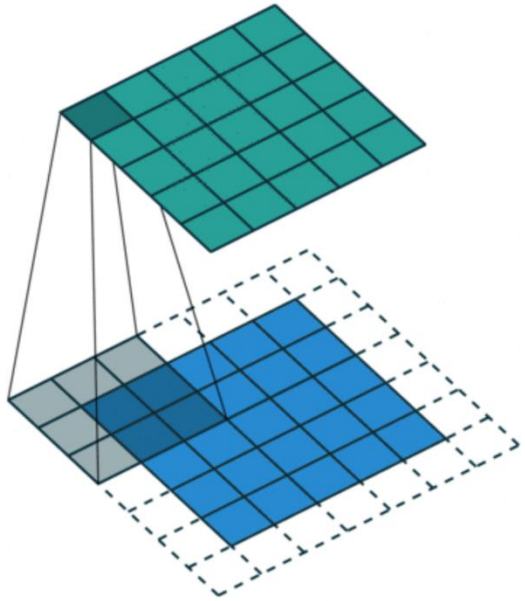
上述算式可簡化成:

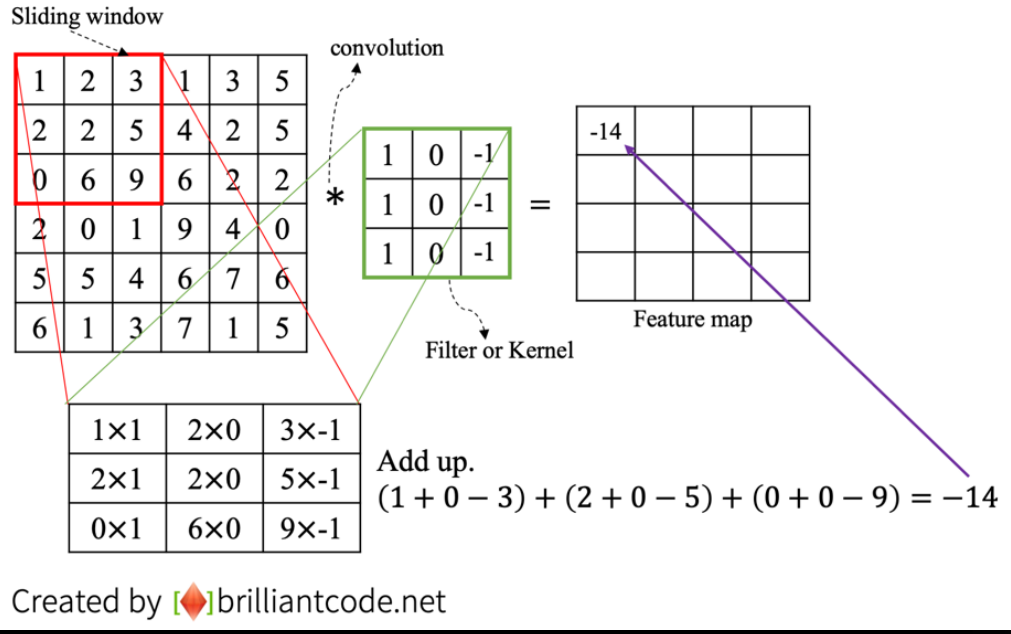
$$[x_1 \quad x_2] \times \begin{bmatrix} w_1 & w_3 & w_5 & w_7 \\ w_2 & w_4 & w_6 & w_8 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$



CNN

convolutional neural network





CNN的運算簡化

X1	X2	X3	X4
X5	X6	X7	X8
X9	X10	X11	X12
X13	X14	X15	X16

*

W1	W2	W3
W4	W5	W6
W7	W8	W9

=

Y1	Y2
Y3	Y4

X1W1+X2W2+X3W3+X5W4+...+X10W8+X11W9 = Y1

X2W1+X3W2+X4W3+X6W4+...+X11W8+X12W9 = Y2

X5W1+X6W2+X7W3+X9W4+...+X14W8+X15W9 = Y3

X8W1+X7W2+X8W3+X9W4+...+X15W8+X16W9 = Y4

Im2col →

CNN的運算簡化

X1	X2	X3	X5	X6	X7	X9	X10	X11
X2	X3	X4	X6	X7	X8	X10	X11	X12
X5	X6	X7	X9	X10	X11	X13	X14	X15
X6	X7	X8	X10	X11	X12	X14	X15	X16

*

W1
W2
W3
W4
W5
W6
W7
W8
W9

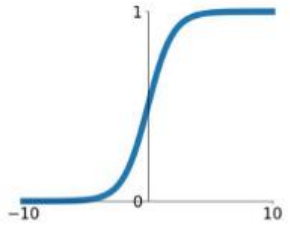
=

Y1
Y2
Y3
Y4

活化函數

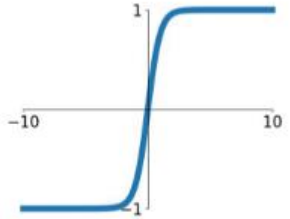
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



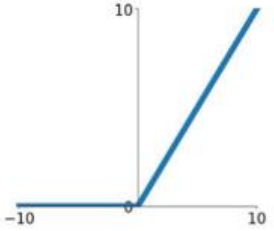
tanh

$$\tanh(x)$$



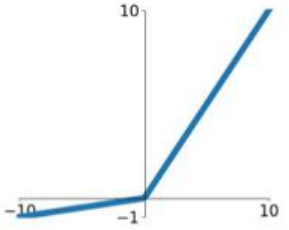
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

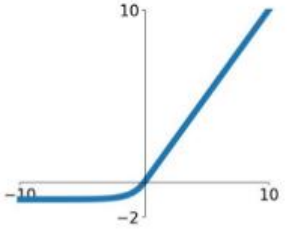


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

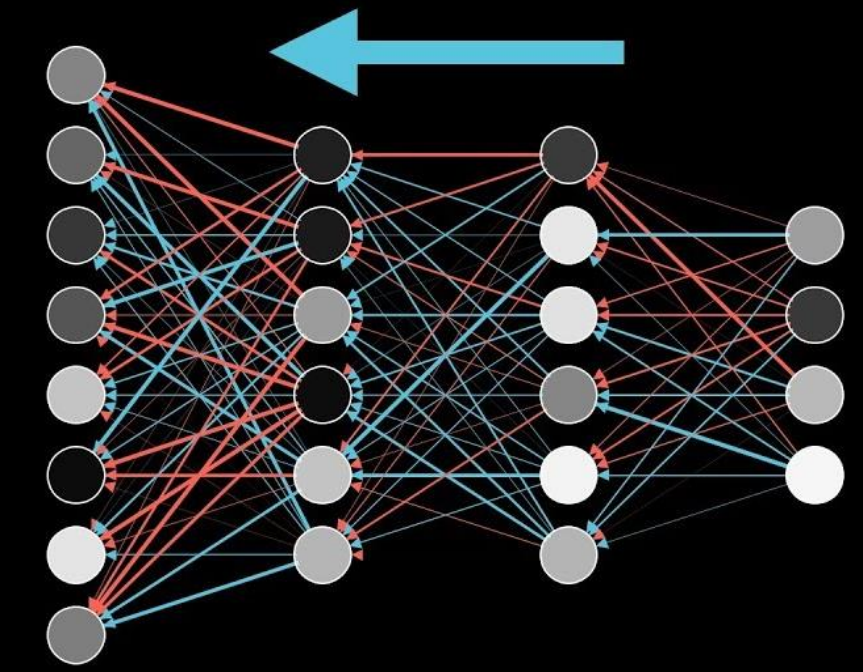
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



- 常使用的活化函數有:
  - Sigmoid/ReLU/Leaky ReLU/Elu/Tanh/Max out

神經網路如何學習

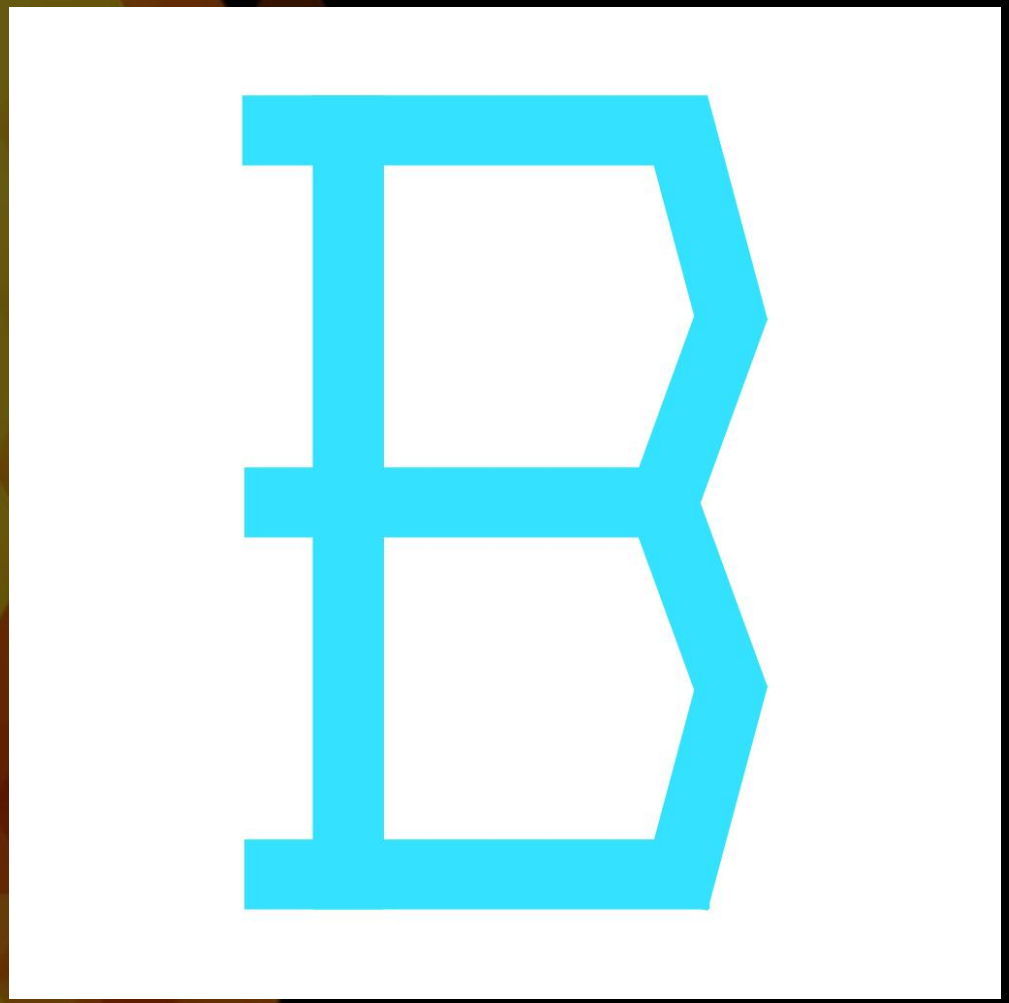
- 神經網路的學習簡單來說就是找到最佳的權重或濾鏡
- 最暴力的做法就是對每個參數做偏微分，並且對將權重往梯度方向做改變
- 目前最常使用的方法是:
  誤差反向傳播法
  (BackPropagation)



套件製作

- 想要達成的效果:
  - 方便易用，適合程式/神經網路的初學者
  - 擴展性高，適合專業用戶外加功能等等
  - 大部分的網路結構都必須支援
  - 效率不能比現有的套件(如keras tensorflow)差太多
  - 程式易讀，運算結構簡單，適合用來教學

套件成果



套件成果—實作

- 使用MNIST資料集來做實作測試
  使用LeNet架構來測試:
- 訓練10個Epoch之後的準確率為99%

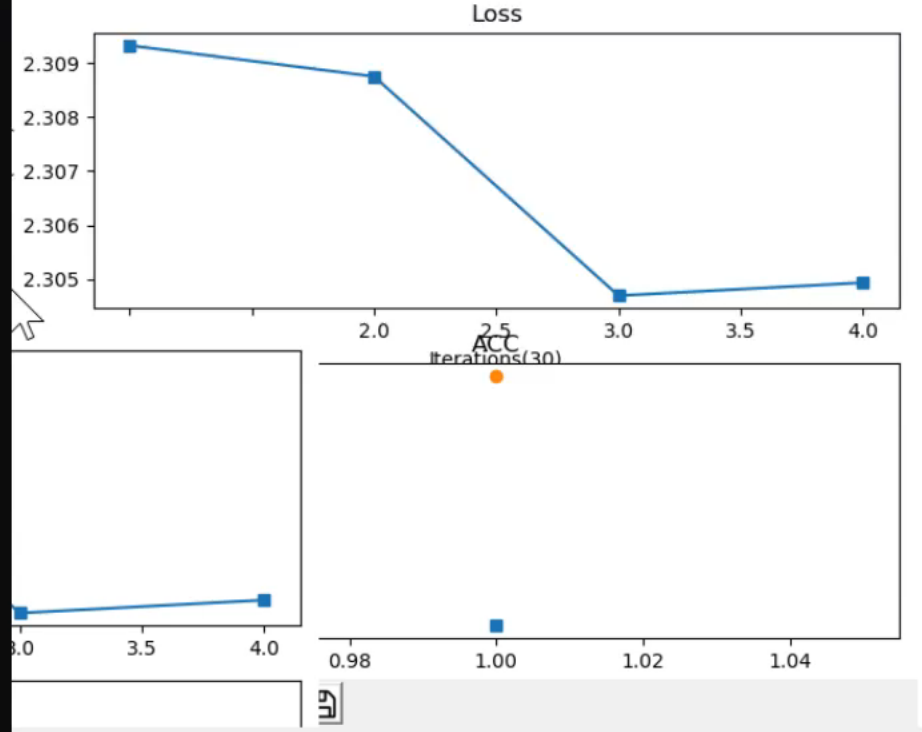
Layer	GFLOPs	Params	Shape(In)	Shape(Out)
Conv	0.000	156	(1, 28, 28)	(6, 28, 28)
Max-Pool	0.000	0	(6, 28, 28)	(6, 14, 14)
Conv	0.000	2416	(6, 14, 14)	(16, 10, 10)
Max-Pool	0.000	0	(16, 10, 10)	(16, 5, 5)
Conv	0.000	48120	(16, 5, 5)	(120, 1, 1)
Flatten	0.000	0	(120, 1, 1)	(120,)
Dense	0.000	10164	(120,)	(84,)
Dense	0.000	850	(84,)	(10,)
Softmax	0.000	0	(10,)	(10,)
Total	0.00	61706		

```
KeyboardInterrupt

C:\Users\apoll\OneDrive\桌面\code\Net>python train.py
Start Accuracy:
Test: 10.28% Train: 10.44%  Time: 2.16sec

Epoch:10

 Epoch  1  Loss:2.304697
```



套件成果—效率

- 根據Keras的官方數據，1080ti運行ResNet一個Epoch要35秒

Model	n	200-epoch accuracy	Original paper accuracy	sec/epoch GTX1080Ti
ResNet20 v1	3	92.16 %	91.25 %	35
ResNet32 v1	5	92.46 %	92.49 %	50
ResNet44 v1	7	92.50 %	92.83 %	70
ResNet56 v1	9	92.71 %	93.03 %	90
ResNet110 v1	18	92.65 %	93.39+-.16 %	165
ResNet164 v1	27	- %	94.07 %	-
ResNet1001 v1	N/A	- %	92.39 %	-

- 我的套件用2080ti運行一個Epoch要約莫45秒
  經過效能換算後可以得出我的套件的效率大約是60~70%
  算是不錯的成果

參考資料

- 斎藤康毅(2016)。*ゼロから作るDeep Learning──Pythonで学ぶディープラーニングの理論と実装*。日本東京都:O'Reilly Japan。
- 斎藤康毅(2018)。*ゼロから作るDeep Learning──自然言語処理編*。日本東京都:O'Reilly Japan。
- CASPER HANSEN(2019).Activation Functions Explained - GELU, SELU, ELU, ReLU and more.
  Retrieved from https://mlfromscratch.com/activation-functions-explained/#/
- Anonymous authors(2018).
  IMPROVING DEEP LEARNING BY INVERSE SQUARE ROOT LINEAR UNITS (ISRLUS).
  Retrieved from https://openreview.net/pdf?id=HkMCybx0-
- Agnan Kessy, Alex Lewin, Korbinian Strimmer(2018).
  Gaussian Error Linear Units (GELUs)
  Retrieved from https://arxiv.org/abs/1606.08415
- 黑暗星球(2018)。ResNet v2論文筆記。
  檢自 https://blog.csdn.net/u014061630/article/details/80558661

參考資料

- iFADA(2018)。機器學習L1和L2範式和歸一化。
  檢自 https://blog.csdn.net/qq_41044525/article/details/80705888
- Christopher Olah(2015). Understanding LSTM Networks.
  Retrieved from https://colah.github.io/posts/2015-08-Understanding-LSTMs/
- Simeon Kostadinov(2017). Understanding GRU Networks.
  Retrieved from https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be
- dokelung(2017). Python-QA.
  Retrieved from https://github.com/dokelung/Python-QA
- DeepAge, Residual Networkの理解とチューニングのベストプラクティス(2016-11-30)
  檢自 https://deepage.net/deep_learning/2016/11/30/resnet.html
- AI-SCHOLAR, 精度を維持したままパラメータ数を大幅に削減「GhostNet」(2020-03-16)
  檢自 http://test2.ai-scholar.tech/image-recognition/ghostnet-ai-383/

感謝

- 父母
- 老師
- 同學

特別感謝

- 高煥堂 教授