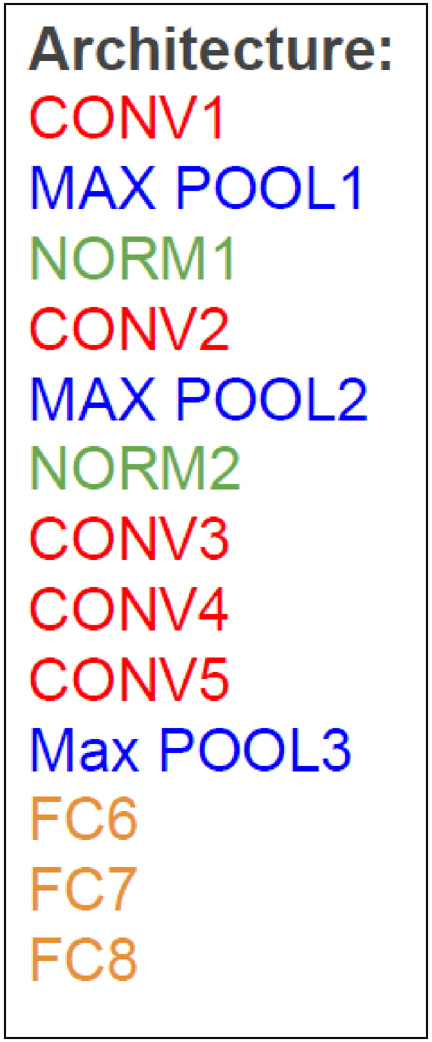
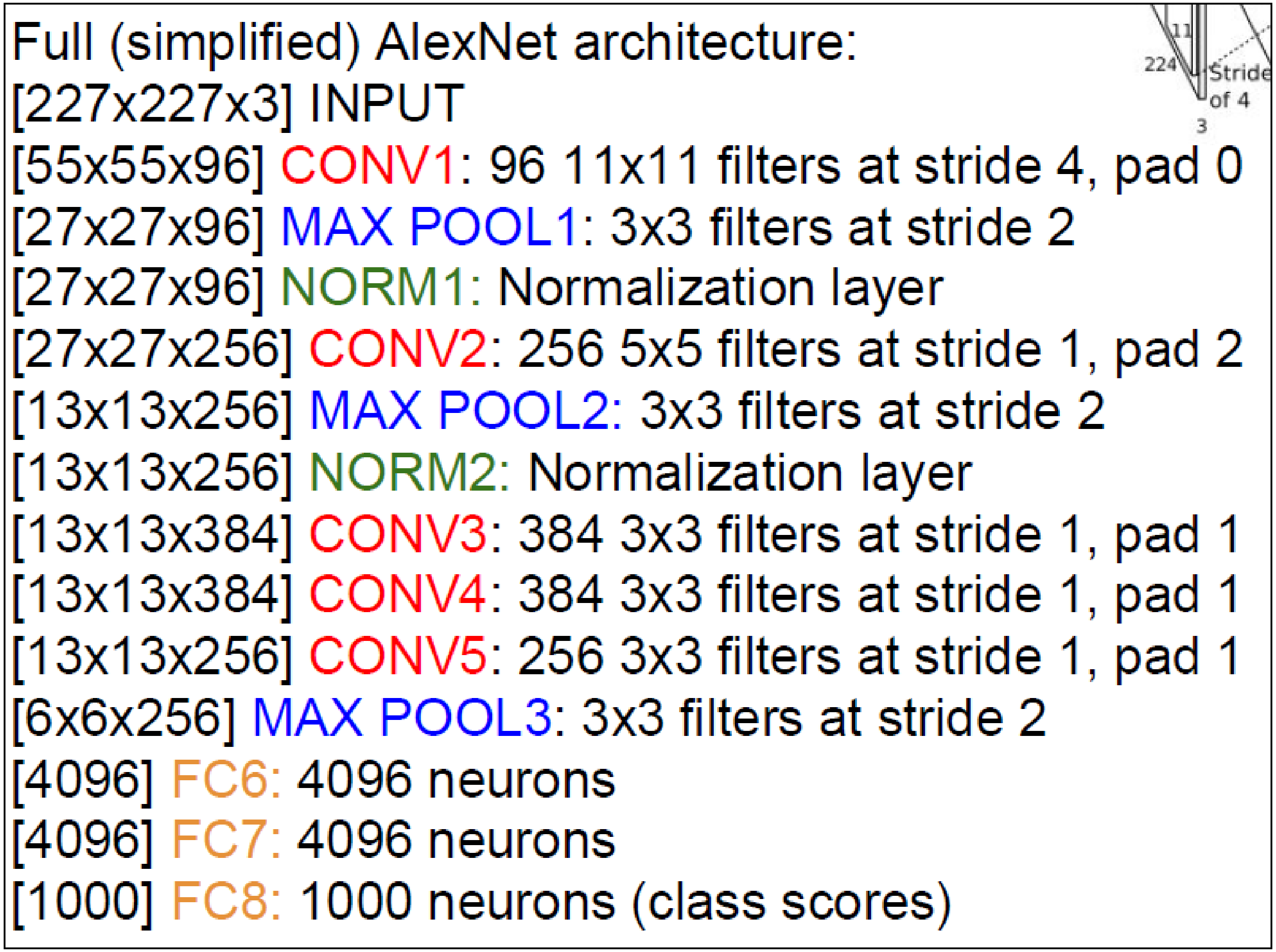
本日流程

1. AlexNet架構回憶
2. 讀取資料(kaggle)
3. Pytorch程式碼

AlexNet架構



Architecture:
CONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8



224
11
Stride
of 4
3

Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

AlexNet架構

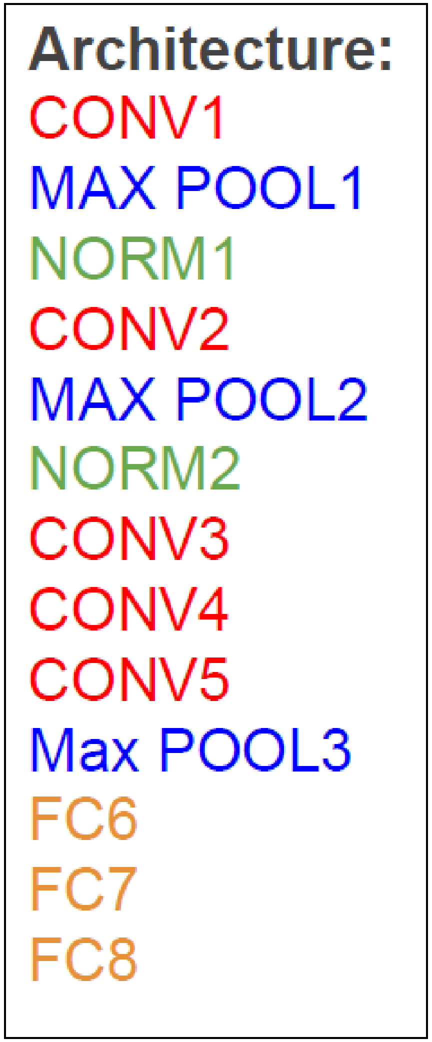
Architecture:

CONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8

Details/Retrospectives:

- first use of ReLU
- used Norm layers (not common anymore)
- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus

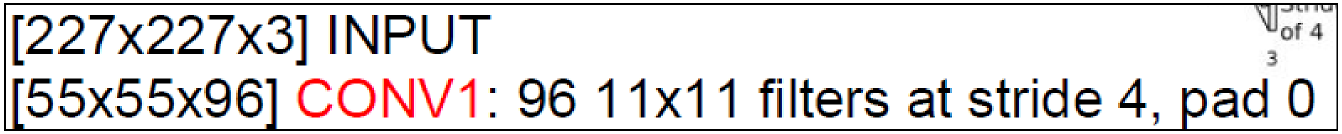
AlexNet架構



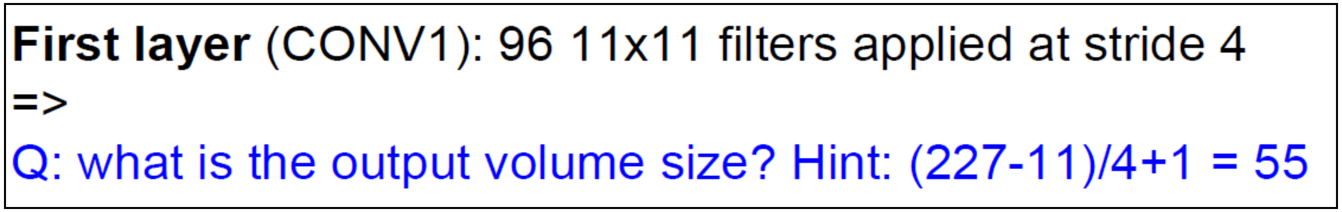
Architecture:

CONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8

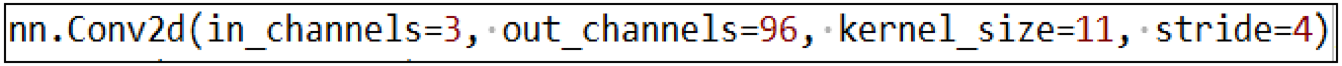
Conv. Layer



[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0

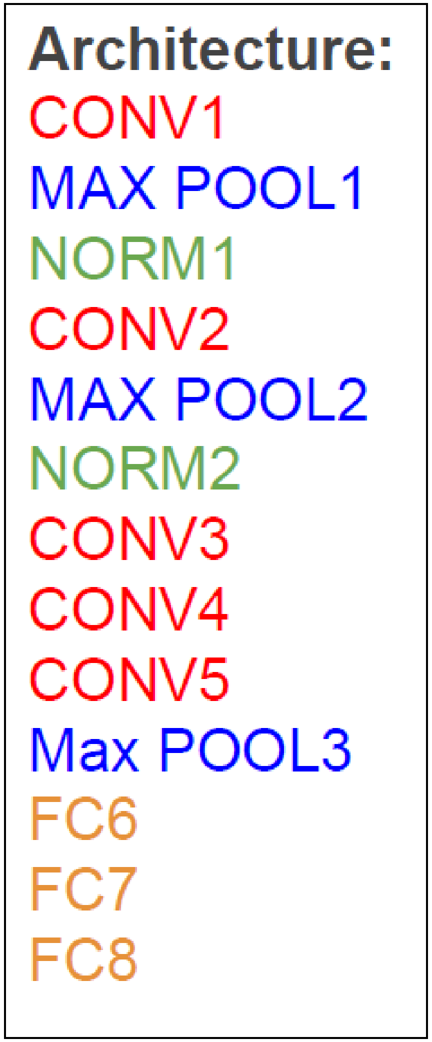


First layer (CONV1): 96 11x11 filters applied at stride 4
=>
Q: what is the output volume size? Hint: (227-11)/4+1 = 55

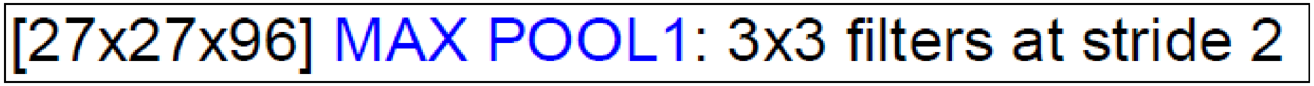


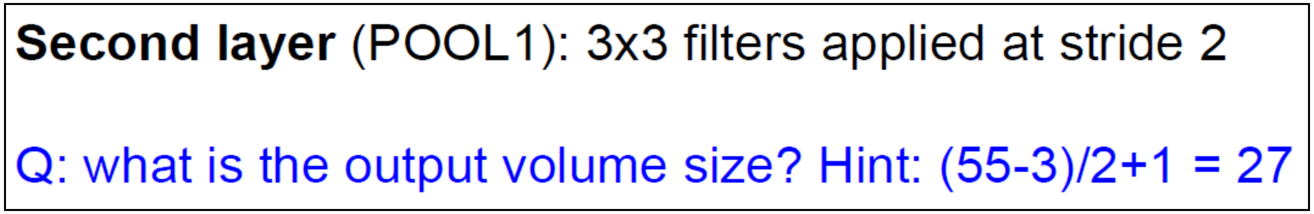
```
nn.Conv2d(in_channels=3, out_channels=96, kernel_size=11, stride=4)
```

AlexNet架構

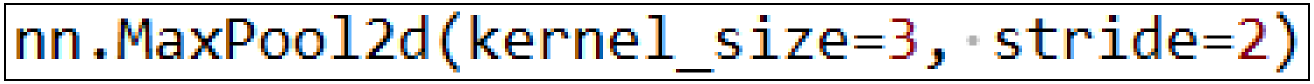

Architecture:
CONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8

Max Pooling Layer


[27x27x96] MAX POOL1: 3x3 filters at stride 2


Second layer (POOL1): 3x3 filters applied at stride 2

Q: what is the output volume size? Hint: (55-3)/2+1 = 27


```
nn.MaxPool2d(kernel_size=3, stride=2)
```

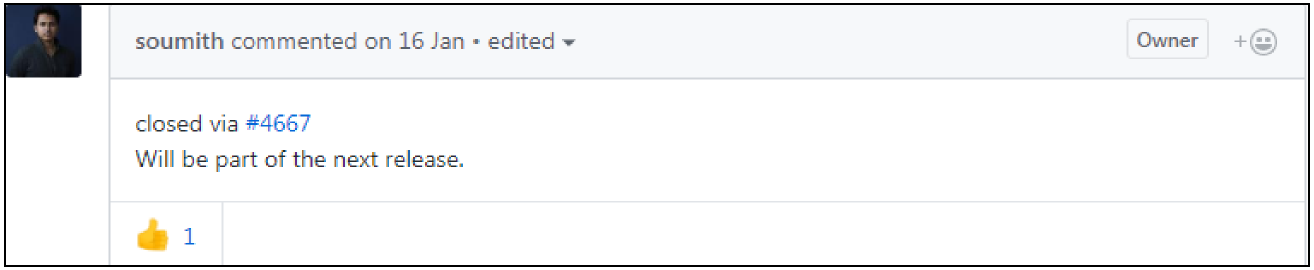
AlexNet架構

Architecture:
CONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8

Normalization Layer

[27x27x96] NORM1: Normalization layer

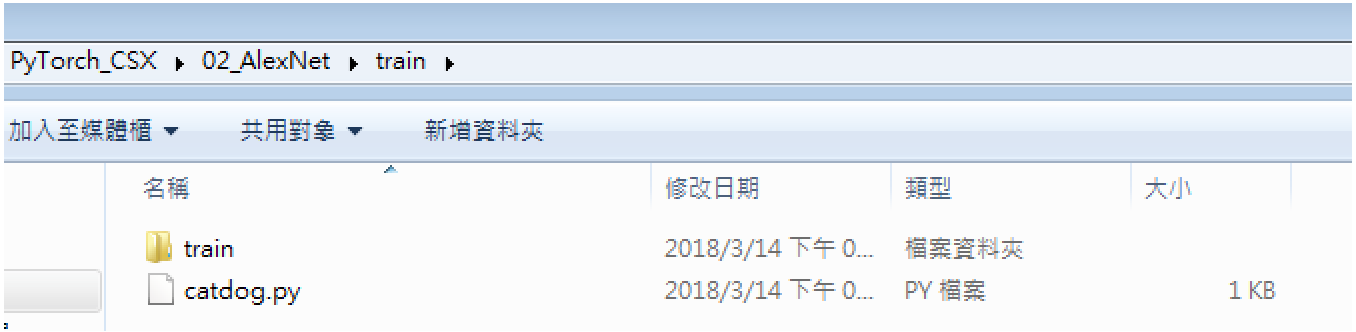
$$b_{x,y}^{i} = a_{x,y}^{i} / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a_{x,y}^{j})^2 \right)^{\beta}$$


soumith commented on 16 Jan • edited

closed via #4667
Will be part of the next release.

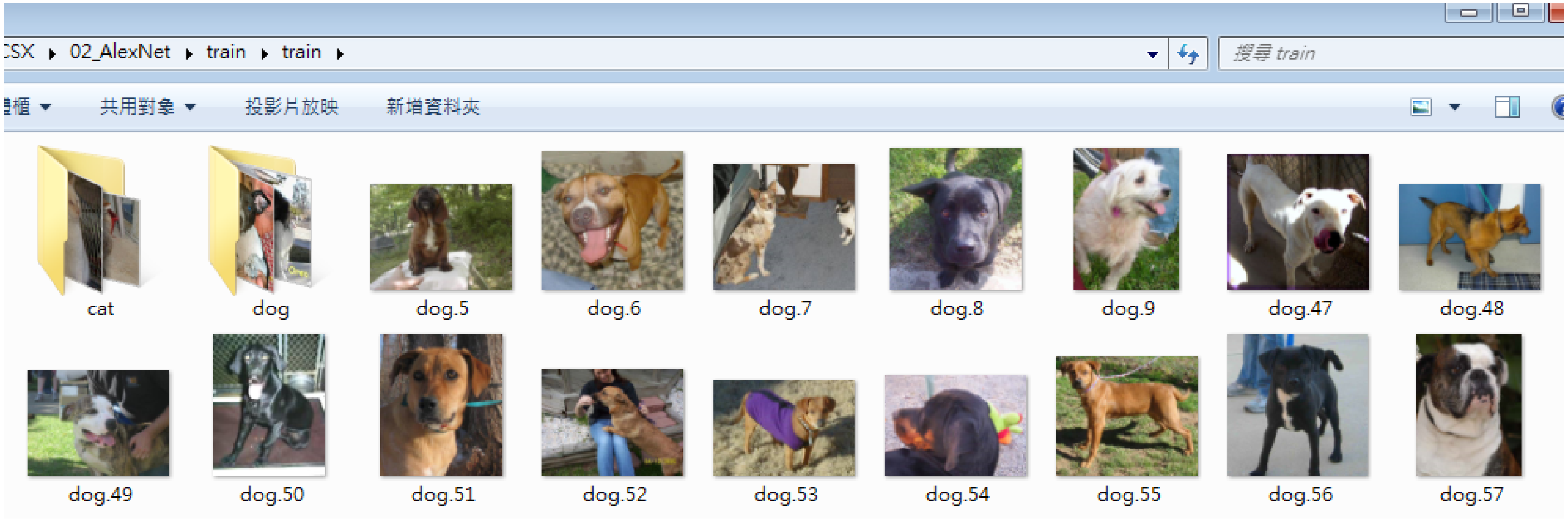
資料讀取

Kaggle 貓與狗



PyTorch_CSX ▸ 02_AlexNet ▸ train ▸

名稱	修改日期	類型	大小
train	2018/3/14 下午 0...	檔案資料夾	
catdog.py	2018/3/14 下午 0...	PY 檔案	1 KB



CSX ▸ 02_AlexNet ▸ train ▸ train ▸

cat, dog, dog.5, dog.6, dog.7, dog.8, dog.9, dog.47, dog.48, dog.49, dog.50, dog.51, dog.52, dog.53, dog.54, dog.55, dog.56, dog.57