4-6 Wednesday – 210-GD3

Special topics in Computer Science INT3121 20

Lecturer: Nguyen Thi Ngoc Diep, Ph.D.

Email: ngocdiep@vnu.edu.vn

Slide & Code: https://github.com/chupibk/INT3121-20

Image classification with convolutional neural networks

Week	Content	Class hour	Self-study hour
1 28/8/2019	Introduction Image classification problem and its applications A toy problem with CIFAR10	2	1
2	CNN model architectures and visualization	2	1
3	Training and tuning parameters Automatic parameter learning	2	1
4	Data augmentation Data generator	2	2-6
5	Transfer learning	2	2-6
6	Multi-output image classification	2	2-6
7	Building a training dataset How to write a report	1	2-6
8, 9, 10, 11	Seminar: Bag of tricks with CNN (as mid-term tests)	1	2-6
12, 13, 14	Final project presentations	1-3	2-6
15	Class summarization	1	open

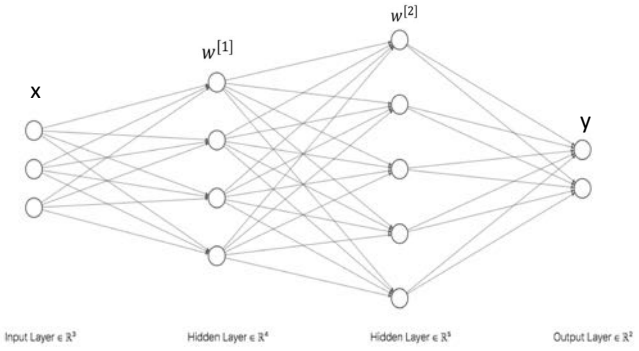
Week 1: Homework checklist

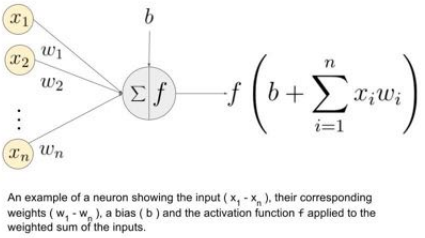
- Install all necessary environments?
  - Keras, tensorflow, python 3, jupyter notebook, numpy, sklearn
- Re-run all the scripts?
- Obtain the label names of CIFAR10 classes?
- Given an image, predict the label of the image?

→ Check lecture code for answers ☺

Layers in Neural Networks

A sample net with fully connected layers





An example of a neuron showing the input (x_1 - x_n), their corresponding weights (w_1 - w_n), a bias (b) and the activation function f applied to the weighted sum of the inputs.

Visualization tool: http://alexlenail.me/NN-SVG/index.html

Image credit: https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/

Code it with Keras

```
from keras.models import Sequential
from keras.layers import *

model = Sequential()
model.add(Dense(4, input_shape=(3,)))
model.add(Dense(5))
model.add(Dense(2))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_4 (Dense)              (None, 4)                 16
_________________________________________________________________
dense_5 (Dense)              (None, 5)                 25
_________________________________________________________________
dense_6 (Dense)              (None, 2)                 12
=================================================================
Total params: 53
Trainable params: 53
Non-trainable params: 0
_________________________________________________________________
```

Where does this come from?

Convolution Layers

LeNet example

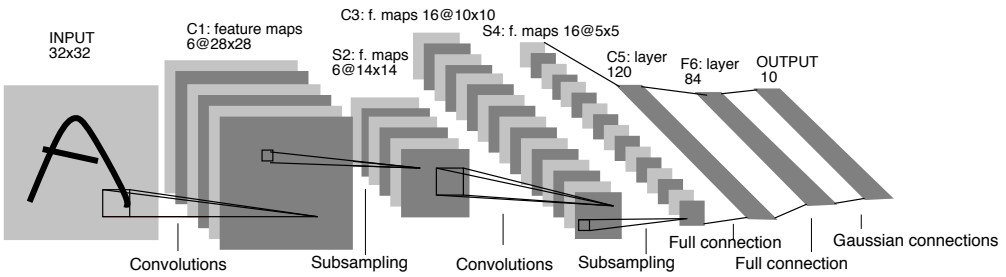
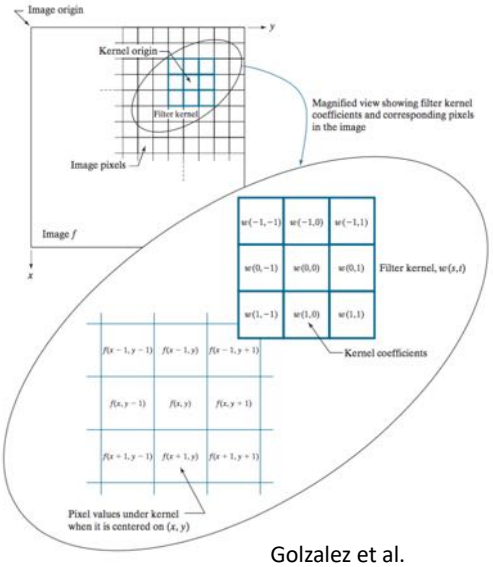


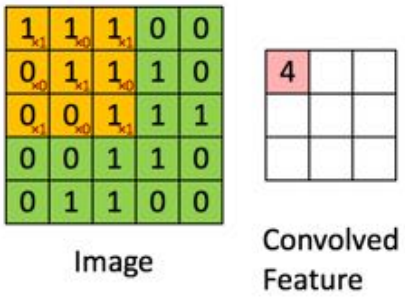
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Reference: LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.
http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf

Spatial filtering

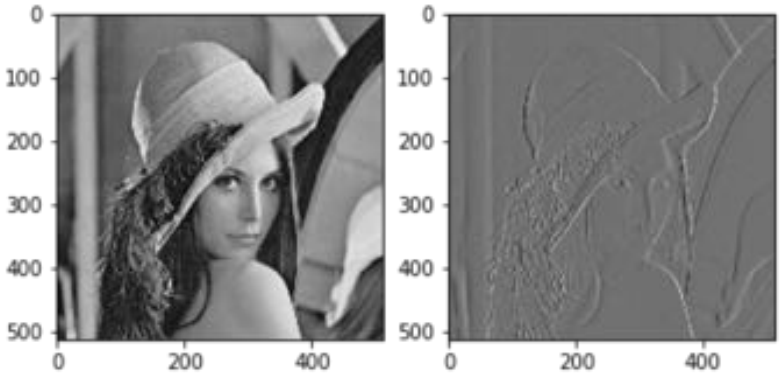


Golzalez et al.



Spatial filtering as feature extraction

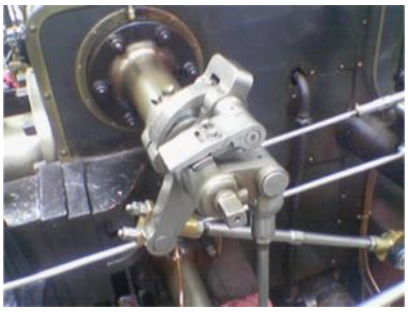
$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * I$$



Horizontal changes

Vertical changes

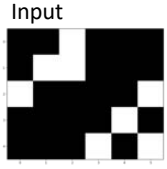
Sobel in both directions



https://en.wikipedia.org/wiki/Sobel_operator

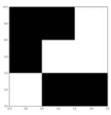
Kernel = function
Convolution = operation to combine 2 functions

Input



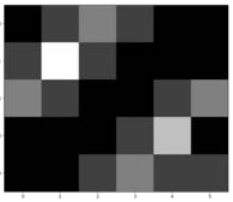
```
arr = [
    [0, 0, 1, 0, 0, 0],
    [0, 1, 1, 0, 0, 0],
    [1, 0, 0, 0, 0, 1],
    [0, 0, 0, 0, 1, 0],
    [0, 0, 0, 1, 0, 1]
]
```

Template

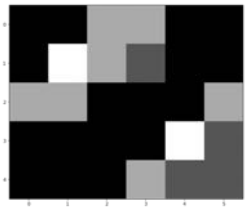


```
template = [
    [0, 0, 1],
    [0, 1, 1],
    [1, 0, 0]
]
```

Correlation operation



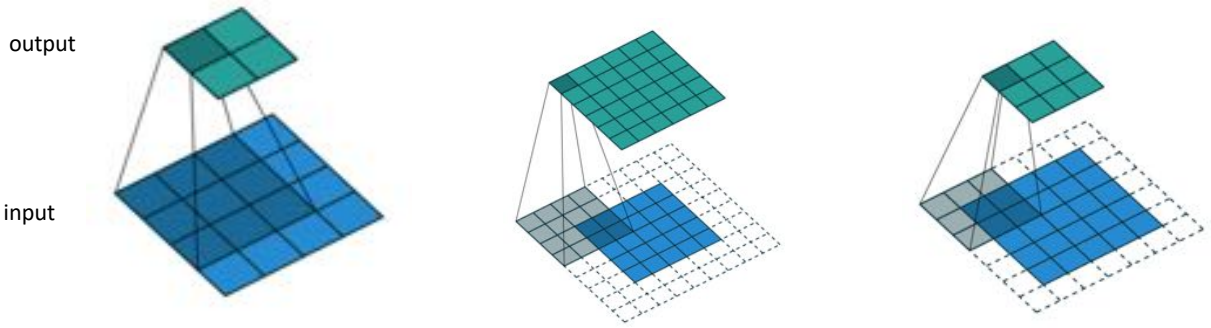
Convolution operation



Convolution layers are about kernels and their behaviors

- Having “gap” in kernels?
  - Traditional vs Dilated (a.k.a Atrous)
- Dealing with pixels at border?
  - Padding
- Moving step across images?
  - Stride
- Accelerating computation?
  - Separable
- Upsampling data?
  - Deconvolution (aka, Transposed convolution)

Stride & Padding



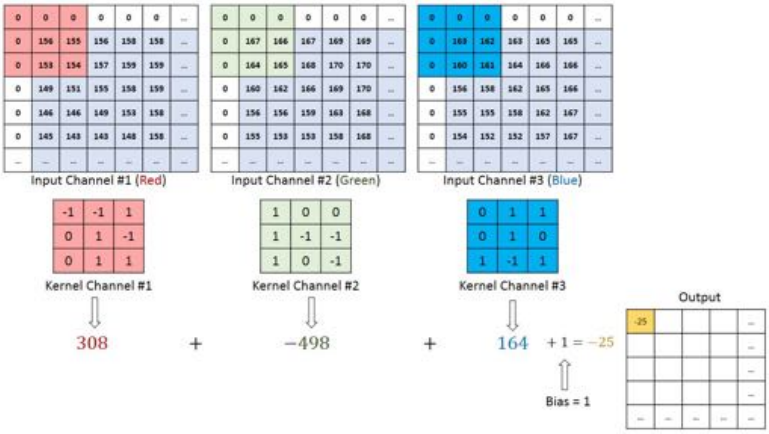
kernel_size=(3,3), padding=0, stride=1

kernel_size=(4,4), padding=2, stride=1

kernel_size=(3,3), padding=1, stride=2

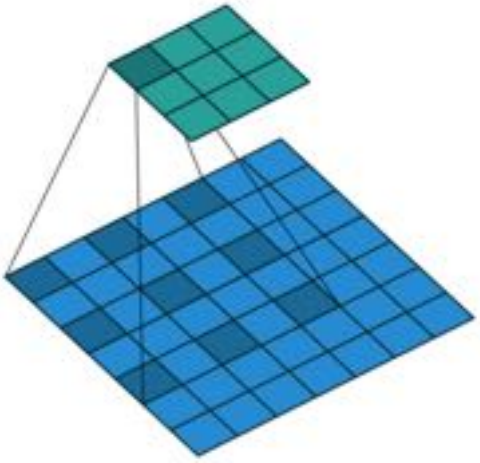
Credit: https://github.com/vdumoulin/conv_arithmetic

Convolution for multichannel input



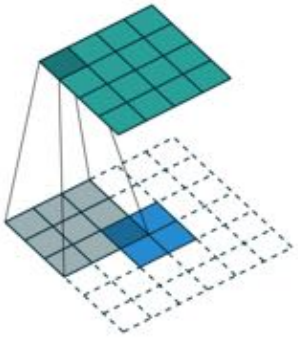
Credit: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Dilated convolution



Credit: https://github.com/vdumoulin/conv_arithmetic

Upsampling
Deconvolution (Transposed convolution)



Credit: https://github.com/vdumoulin/conv_arithmetic

Important keywords to remember

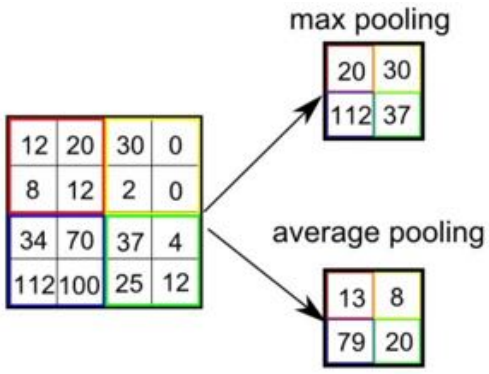
- Filter
- Kernel
- Stride
- Pad
- Receptive field
- Downsampling
- Upsampling
- Feature map

Types of filters

- (Traditional) Convolution
  - Too many parameters
- Dilated or Atrous convolution
  - Large receptive field
  - Less parameters
- Separable convolution
  - Even lesser parameters
- Deconvolution or transposed convolution
  - Low resolution -> high resolution

- Pooling -> not a convolution layer, but...

Pooling layers



Parameters of a Convolution layer

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

A common setting of the hyperparameters is $F = 3, S = 1, P = 1$. However, there are common conventions and rules of thumb that motivate these hyperparameters. See the ConvNet architectures section below.

Reference: http://cs231n.github.io/convolutional-networks/

CNN architectures

Some CNN architectures

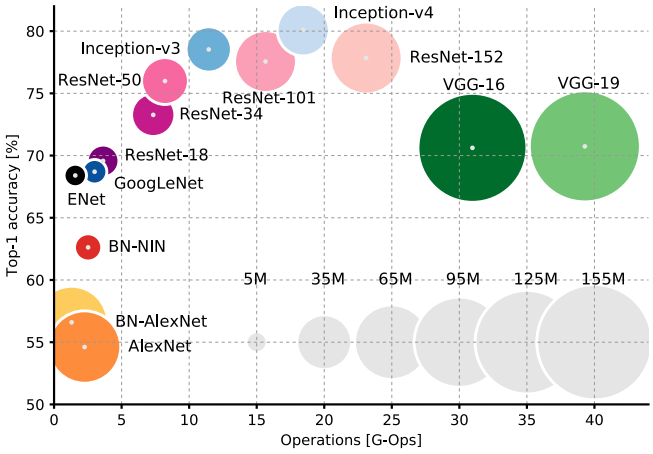


Figure 2: **Top1 *vs.* operations, size $\propto$ parameters.**

(Operations required for a single forward pass)

Reference: Canziani, Alfredo, Adam Paszke, and Eugenio Culurciello. "An analysis of deep neural network models for practical applications." *arXiv preprint arXiv:1605.07678* (2016).

Architectures

- LeNet-5 – LeCun et al., 1998
- AlexNet – Krizhevsky et al., 2012
- VGGNet – Simonyan et al., 2014
- GoogleNet/Inception – Szegedy et al., 2014
- Resnet – Kaiming et al., 2015

LeNet-5

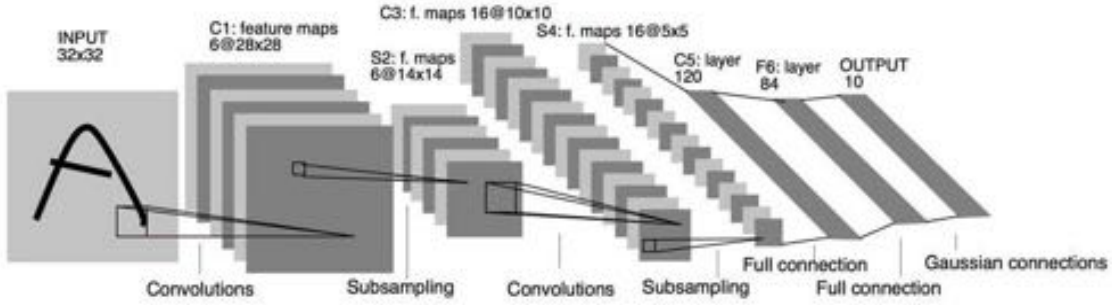


Image credit: LeCun

AlexNet

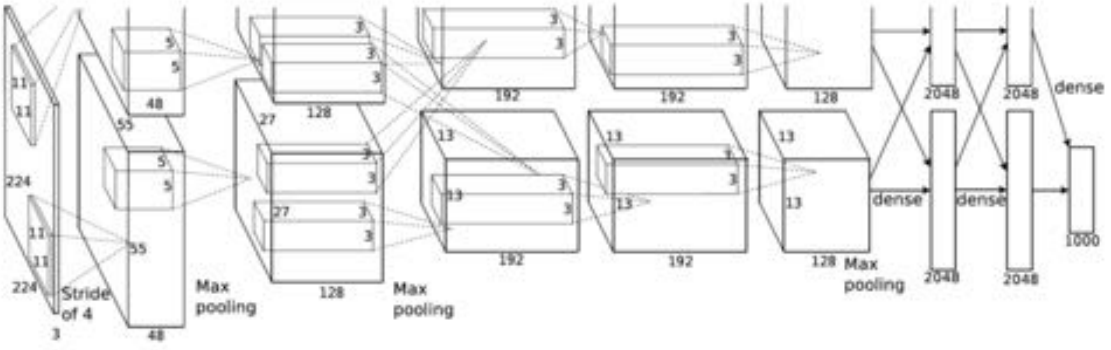
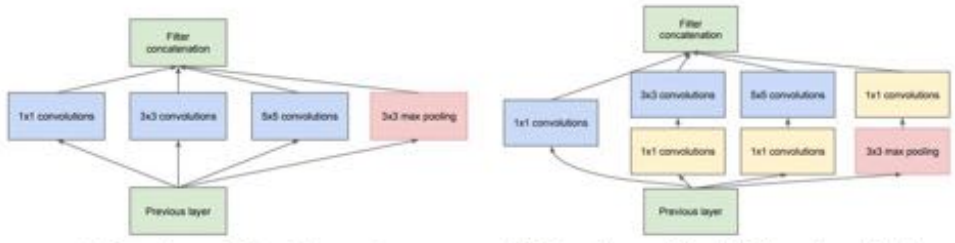


Image credit:...

VGG16

Image credit: ...

GoogLeNet/Inception



(a) Inception module, naïve version

(b) Inception module with dimension reductions

Image credit: ...

ResNet with “skip connections”

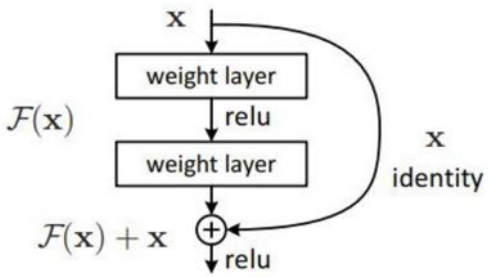


Image credit: ...