From Perceptron to Deep Learning

DLAI3 Workshop, 1 July 2020

Somnuk Phon-Amnuaisuk
School of Computing and Informatics
Centre for Innovative Engineering
Universiti Teknologi Brunei

1

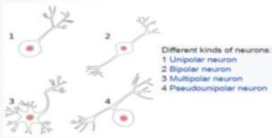
2

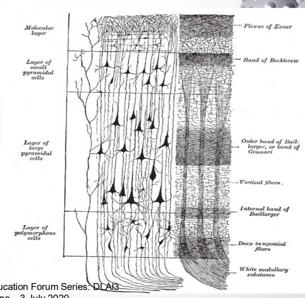
Outline

- Session 1: Overview of ANN & CNN
  - Actual Neurons and Perceptron
  - Multi-Layer Perceptron
  - Deep Learning: CNN Based Computer Vision
- Session 2: Representation of Input Image
- Session 3: TensorFlow

3

Neurons

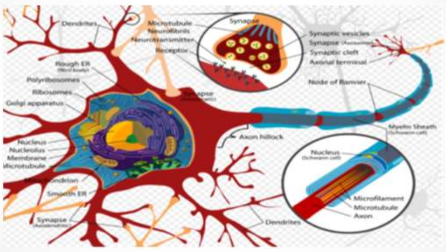




wikipedia

4

Neurons

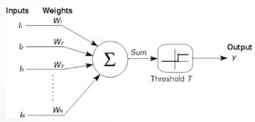


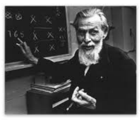
Wikipedia (approx. 21*10^9 [70-40] mm)

5

Threshold Logic Unit

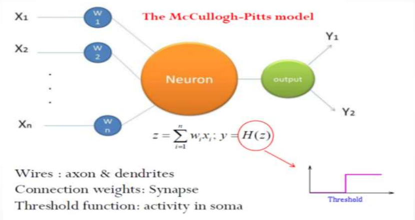
- The early artificial neuron was the Threshold Logic Unit (TLU), or Linear Threshold Unit, first proposed by Warren McCulloch and Walter Pitts in 1943.





6

Threshold Logic Unit



7

How to determine the values of W?

- How to solve for W

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w}\cdot\mathbf{x} + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

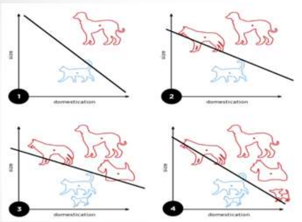
$$\mathbf{w}\cdot\mathbf{x} \text{ is } \sum_{i=1}^{m} w_i x_i$$

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

- Randomly
- Analytically
- Optimization algorithm e.g., gradient descent

8

Rosenblatt's Perceptron Learning Rule



$$w_i(t+1) = w_i(t) + \alpha(d_j - y_j(t))x_{j,i}$$

- The perceptron is an artificial neuron using the Heaviside step function as the activation function.
- The perceptron learning rule is an algorithm for learning a classifier function. It was invented in 1958 by Frank Rosenblatt.

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

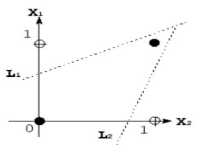
9

Limitation of a Perceptron

- The perceptron can distinguish a linearly separable function. Hence a single perceptron cannot deal with an xor function.

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

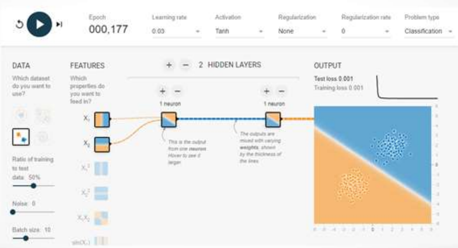
$Y = X_1 \oplus X_2$



10

TensorFlow ANN Playground

- https://playground.tensorflow.org

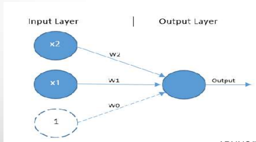


11

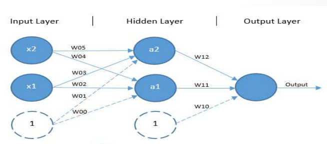
Limitation of Perceptrons

- Adding two perceptrons in a hidden layer, then it can handle an xor function.

perceptron

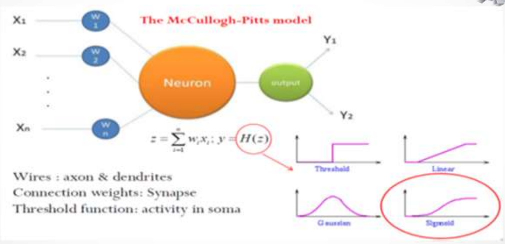


MLP



12

Activation Functions



13

Activation Functions

Binary step	$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a. Sigmoid or Soft step)	$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}$ [1]	$f'(x) = f(x)(1 - f(x))$
TanH	$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$	$f'(x) = 1 - f(x)^2$
Rectified linear unit (ReLU)[15]	$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ 1 & \text{for } x > 0 \end{cases}$
Exponential linear unit (ELU)[20]	$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} f(\alpha, x) + \alpha & \text{for } x \leq 0 \\ 1 & \text{for } x > 0 \end{cases}$

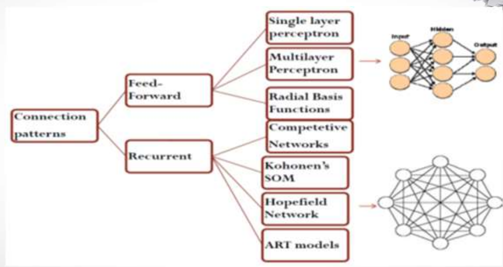
14

Artificial Neural Networks

- 1940: Mcculloch and Pitts
  - A logical calculus of the ideas immanent in nervous activity
- 1958: Frank Rosenblatt
  - The perceptron learning algorithm, convergence theorem
- 1960: Minsky and Papert
  - Perceptrons cannot deal with XOR
- 1980: Werbos and Rumelhart
  - Back-propagation learning algorithm
- 1980: Hopfield
  - Hopfield's energy approach (recurrent neural network)
- 1980 - now: Fukushima, LeCun, Hinton, Schmidhuber, etc

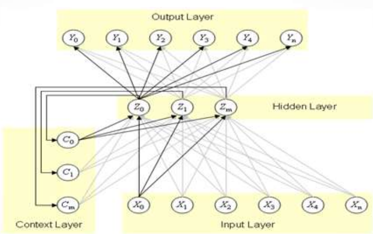
15

Artificial Neural Networks



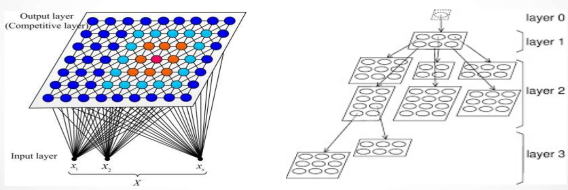
16

Artificial Neural Networks



17

Artificial Neural Networks



18

Learning

- Learning Paradigms
  - Supervised
  - Unsupervised
  - Hybrid
- Learning Approaches
  - Hebbian rules: cells that fire together wire together
  - Competitive learning: increasing specialization of cells
  - Hopfield networks: associative memory
  - Error correction: backpropagation

19

Backpropagation (Rumelhart 1986)

$$E = \frac{1}{2}(t-y)^2 \qquad \frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial o_j}\frac{\partial o_j}{\partial \mathrm{net}_j}\frac{\partial \mathrm{net}_j}{\partial w_{ij}}$$

$$o_j = \varphi(\mathrm{net}_j) = \varphi\left(\sum_{i=1}^{n} w_{ij} o_i\right)$$

$$\varphi(z) = \frac{1}{1+e^{-z}} \qquad \frac{d\varphi}{dz}(z) = \varphi(z)(1-\varphi(z))$$

$$\frac{\partial E}{\partial o_j} = \frac{\partial E}{\partial y} = \frac{\partial}{\partial y}\frac{1}{2}(t-y)^2 = y-t$$

$$\frac{\partial o_j}{\partial \mathrm{net}_j} = \frac{\partial}{\partial \mathrm{net}_j}\varphi(\mathrm{net}_j) = \varphi(\mathrm{net}_j)(1-\varphi(\mathrm{net}_j))$$

$$\frac{\partial \mathrm{net}_j}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}\left(\sum_{i=1}^{n} w_{ij} o_i\right) = \frac{\partial}{\partial w_{ij}} w_{ij} o_i = o_i$$

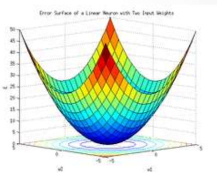
$$\frac{\partial E}{\partial w_{ij}} = (o_j - t)\, o_j(1-o_j)\, o_i$$

20

Square Error Loss

$$E = (t-y)^2$$

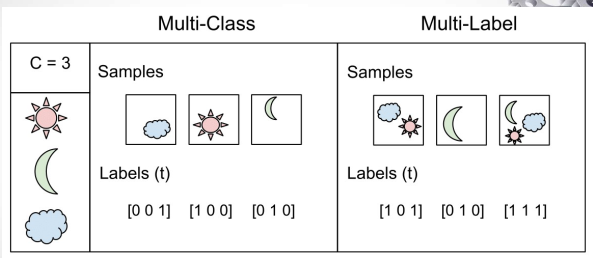
```
def MSE(t,y):
   return np.sum((t-y)**2) / y.size
```



- One-Hot encoding
- Use a binary vector of length n to represent labels of n possible classes
- Ex. [0 0 1], [1 0 0], [0 1 0]

21

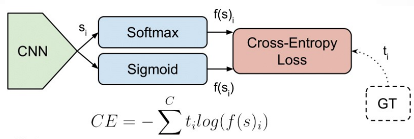
Multi-class / Multi-Label



https://gombru.github.io/2018/05/23/cross_entropy_loss/

22

Cross Entropy Loss



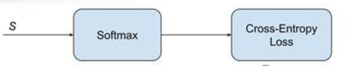
$$CE = -\sum_{i}^{C} t_i log(f(s)_i)$$

$$CE = -\sum_{i=1}^{C'=2} t_i log(f(s_i)) = -t_1 log(f(s_1)) - (1-t_1) log(1-f(s_1))$$

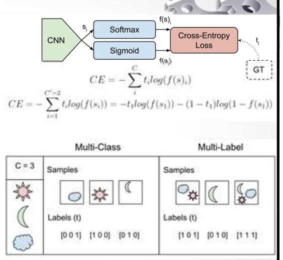
https://gombru.github.io/2018/05/23/cross_entropy_loss/

23

Categorical Cross Entropy Loss



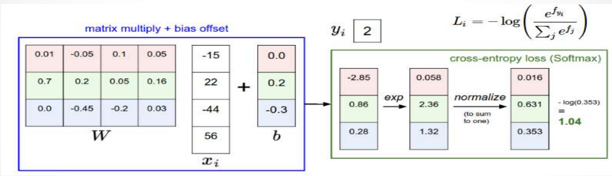
$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \qquad CE = -\sum_{i}^{C} t_i log(f(s)_i)$$



https://gombru.github.io/2018/05/23/cross_entropy_loss/

24

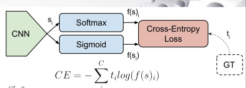
Categorical Cross Entropy Loss



softmax = tf.exp(logits) / tf.reduce_sum(tf.exp(logits), axis)

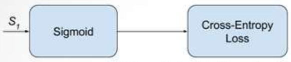
25

Binary Cross Entropy Loss



$$CE = -\sum_{i}^{C} t_i log(f(s)_i)$$

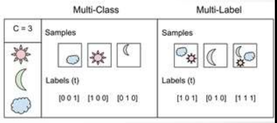
$$CE = -\sum_{i=1}^{C'=2} t_i log(f(s_i)) = -t_1 log(f(s_1)) - (1-t_1) log(1-f(s_1))$$



$$CE = -t_1 log(f(s_1)) - (1-t_1) log(1-f(s_1))$$

$$f(s_i) = \frac{1}{1+e^{-s_i}}$$

```
def BinaryCrossEntropy(yHat, y):
  if y == 1:
    return -log(yHat)
  else:
    return -log(1 - yHat)
```



https://gombru.github.io/2018/05/23/cross_entropy_loss/

26

Gradient and Stochastic Gradient Descent

- Stochastic gradient descent can be regarded as a stochastic approximation of gradient descent optimization, since it replaces the actual gradient (calculated from the entire data set) by an estimate thereof (calculated from a randomly selected subset of the data).
- Stochastic gradient descent is a popular algorithm for training a wide range of models in machine learning.
- When combined with the backpropagation algorithm, it is the de facto standard algorithm for training artificial neural networks.

Wikipedia

27

Gradient and Stochastic Gradient Descent

$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w),$ $Q_i(w)$ is the loss function

a standard (or "batch") gradient descent method

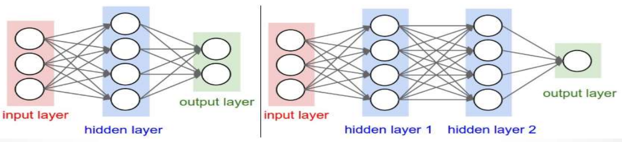
$$w := w - \eta \nabla Q(w) = w - \eta \sum_{i=1}^{n} \nabla Q_i(w)/n,$$

- A compromise between computing the true gradient and the gradient at a single example is to compute the gradient against more than one training example (called a "mini-batch") at each step. This can perform significantly better than "true" stochastic gradient descent described.

Wikipedia

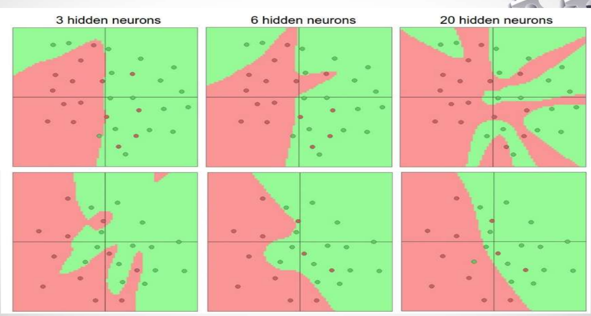
28

Summary: Feedforward Neural Networks



29

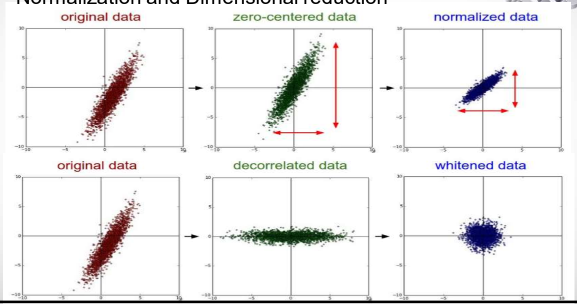
Effects of Network Structure



30

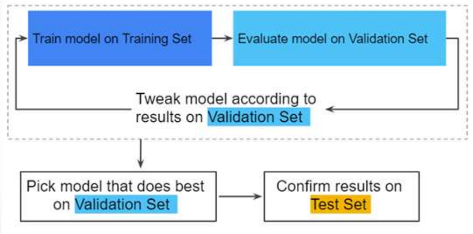
Data Preprocessing

- Normalization and Dimensional reduction



31

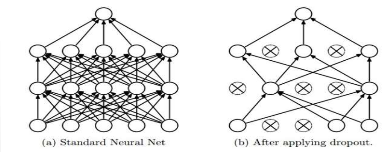
Training, Validation & Test Set



32

Overfitting

Deal with overfitting by
- Adding regularization terms
- Implementing dropout



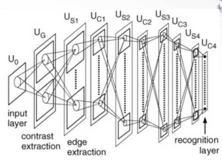
33

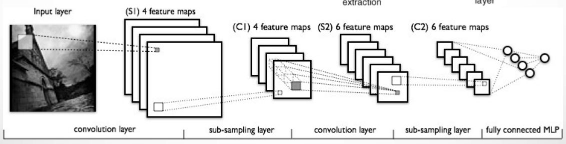
Deep Learning

CNN based computer vision

34

A Brief History of Deep Learning





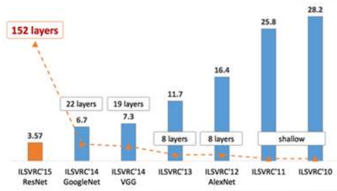
35

A Brief History of Deep Learning

- Hubel and Wiesel (1959) found two types of important cells in the visual primary cortex: *simple cell* and *complex cell.*
- The Neocognitron (Fukushima, 1979) was inspired by the model proposed by Hubel & Wiesel.
- Convolutional neural network (Yann LeCun, 1989) was also inspired by this line of concepts.

36

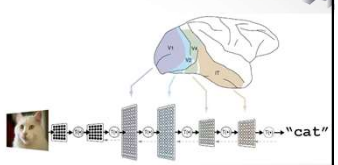
ImageNet (Fei-Fei Li, 2009)



37

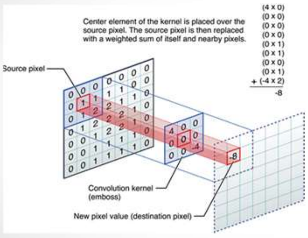
Why does CNN Perform Better?

- CNN shares many interesting commonalities with real brains.
  - Each neuron is connected to a small subset of neurons.
  - Each neuron perform simple function e.g., Sigmoid, ReLU.
  - Each subset of neurons learn specialised features (representation learning).



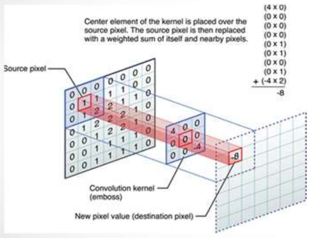
38

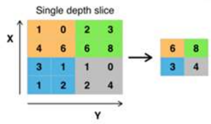
Convolutional Neural Network



39

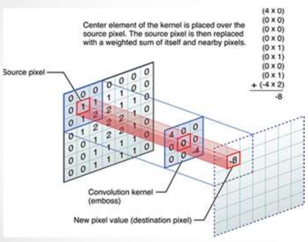
Convolutional Neural Network

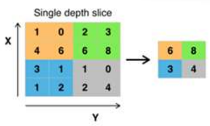


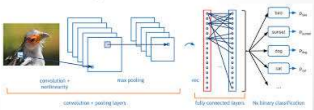


40

Convolutional Neural Network

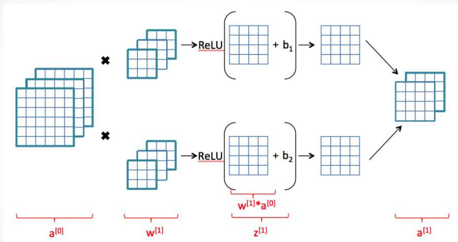






41

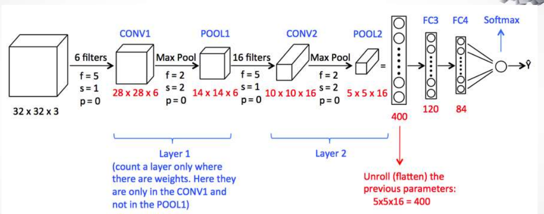
Convolutional Neural Network



https://medium.com/machine-learning-bites/deeplearning-series-convolutional-neural-networks-a9c2f2ee1524

42

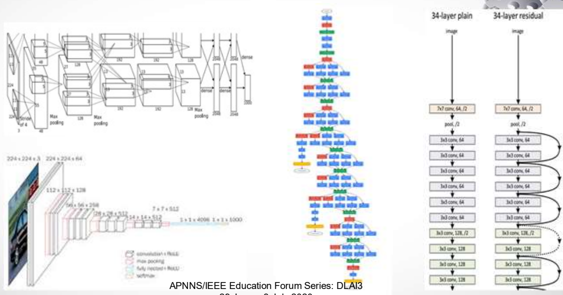
Convolutional Neural Network



https://medium.com/machine-learning-bites/deeplearning-series-convolutional-neural-networks-a9c2f2ee1524

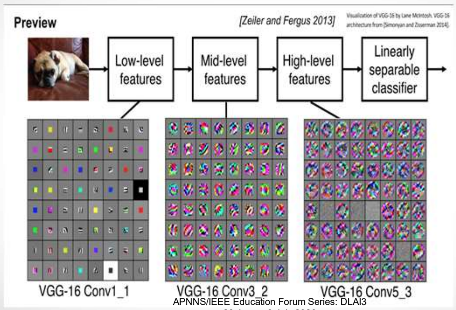
43

Convolutional Neural Networks



44

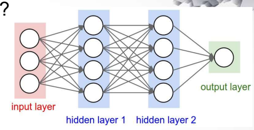
End-to-End Approach (self learned features)



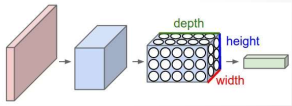
45

Recap: MLP & CNN

- Should MLP be wide rather than deep?
- Vanishing Gradient Problem



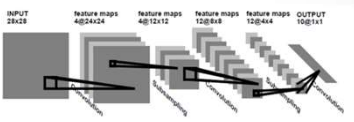
- Convolutional layer
- Pooling layer
- Normalization layer
- Dropout
- Fully connected layer



46

Deep Learning for Image Classification Tasks

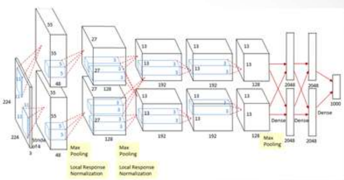
LeNet
AlexNet
ZFNet
VGGNet
SPPNet
GoogLeNet / Inception-v1
BN-Inception / Inception-v2
Inception-v3, Inception-v4
Xception
MobileNet
ResNet
DenseNet
NASNet



47

Deep Learning for Image Classification Tasks

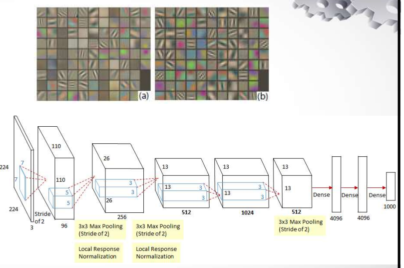
LeNet
AlexNet
ZFNet
VGGNet
SPPNet
GoogLeNet / Inception-v1
BN-Inception / Inception-v2
Inception-v3 Inception-v4
Xception
MobileNet
ResNet
DenseNet
NASNet



48

Deep Learning for Image Classification Tasks

- LeNet
- AlexNet
- **ZFNet**
- VGGNet
- SPPNet
- GoogLeNet / Inception-v1
- BN-Inception / Inception-v2
- Inception-v3 Inception-v4
- Xception
- MobileNet
- ResNet
- DenseNet
- NASNet

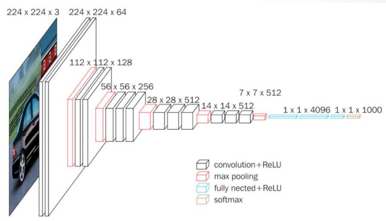


APNNS/IEEE Education Forum Series: DLAI3
29 June – 3 July 2020

49

Deep Learning for Image Classification Tasks

- LeNet
- AlexNet
- ZFNet
- **VGGNet**
- SPPNet
- GoogLeNet / Inception-v1
- BN-Inception / Inception-v2
- Inception-v3 Inception-v4
- Xception
- MobileNet
- ResNet
- DenseNet
- NASNet

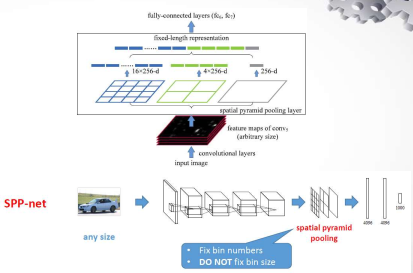


APNNS/IEEE Education Forum Series: DLAI3
29 June – 3 July 2020

50

Deep Learning for Image Classification Tasks

- LeNet
- AlexNet
- ZFNet
- VGGNet
- **SPPNet**
- GoogLeNet / Inception-v1
- BN-Inception / Inception-v2
- Inception-v3 Inception-v4
- Xception
- MobileNet
- ResNet
- DenseNet
- NASNet

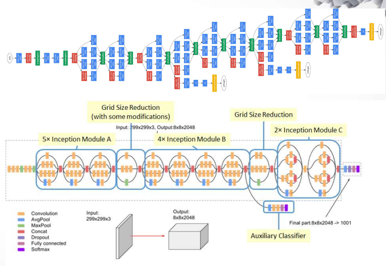


APNNS/IEEE Education Forum Series: DLAI3
29 June – 3 July 2020

51

Deep Learning for Image Classification Tasks

- LeNet
- AlexNet
- ZFNet
- VGGNet
- SPPNet
- **GoogLeNet / Inception-v1**
- **BN-Inception / Inception-v2**
- **Inception-v3 Inception-v4**
- Xception
- MobileNet
- ResNet
- DenseNet
- NASNet

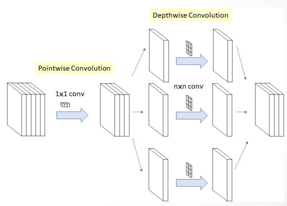


APNNS/IEEE Education Forum Series: DLAI3
29 June – 3 July 2020

52

Deep Learning for Image Classification Tasks

- LeNet
- AlexNet
- ZFNet
- VGGNet
- SPPNet
- GoogLeNet / Inception-v1
- BN-Inception / Inception-v2
- Inception-v3 Inception-v4
- **Xception**
- MobileNet
- ResNet
- DenseNet
- NASNet

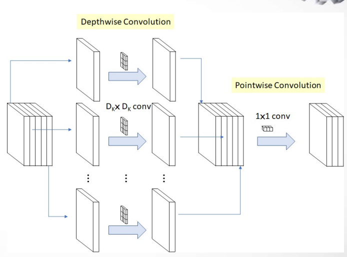


APNNS/IEEE Education Forum Series: DLAI3
29 June – 3 July 2020

53

Deep Learning for Image Classification Tasks

- LeNet
- AlexNet
- ZFNet
- VGGNet
- SPPNet
- GoogLeNet / Inception-v1
- BN-Inception / Inception-v2
- Inception-v3 Inception-v4
- Xception
- **MobileNet**
- ResNet
- DenseNet
- NASNet

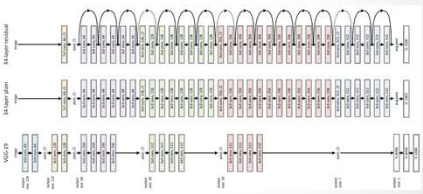


APNNS/IEEE Education Forum Series: DLAI3
29 June – 3 July 2020

54

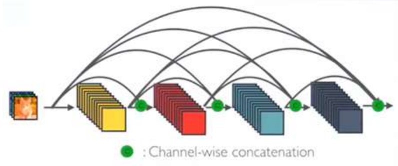
Deep Learning for Image Classification Tasks

LeNet
AlexNet
ZFNet
VGGNet
SPPNet
GoogLeNet / Inception-v1
BN-Inception / Inception-v2
Inception-v3 Inception-v4
Xception
MobileNet
ResNet
DenseNet
NASNet



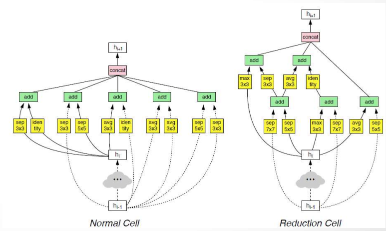
Deep Learning for Image Classification Tasks

LeNet
AlexNet
ZFNet
VGGNet
SPPNet
GoogLeNet / Inception-v1
BN-Inception / Inception-v2
Inception-v3 Inception-v4
Xception
MobileNet
ResNet
DenseNet
NASNet



Deep Learning for Image Classification Tasks

LeNet
AlexNet
ZFNet
VGGNet
SPPNet
GoogLeNet / Inception-v1
BN-Inception / Inception-v2
Inception-v3 Inception-v4
Xception
MobileNet
ResNet
DenseNet
NASNet



Some Characteristics of CNN

- The CNN architecture gets deeper and denser:
  - LeNet5 has 5 layers
  - VGG16 and VGG19 have 16 and 19 layers respectively
  - Residual Networks (ResNets) have more than 100 layers
- A deep network seems to have a better performance than a wide network, given the same amount of nodes.
- Skip connections between layers close to the input and those close to the output could improve training efficiency and performance of the CNN.
- Research trend → handle unlabeled data, learning visual semantics

The Resurgence of Deep Learning

- Ecosystem
  - The availability of data
  - The progress of hardware, software
  - The progress of ICT infrastructure

Matching of
Problems & Solutions

The Resurgence of Deep Learning

- Ecosystem
  - The availability of data
  - The progress of hardware, software
  - The progress of ICT infrastructure

Deep Learning
- Good performance
- End-to-end approach (representation learning)

Challenges
- Speed and accuracy in open-ended real life problems
- The whole ecosystem is still developing

61

Deep Learning Frameworks

Chainer
mxnet
Caffe2
Microsoft CNTK
TensorFlow
Keras
GLUON
PYTORCH
theano

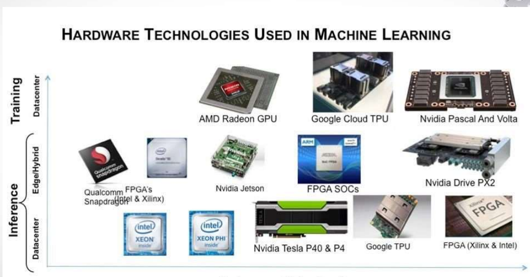
62

Hardware Landscape

- There are four major types of technology that can be used to accelerate the training and use of deep neural networks:
- CPUs
- GPUs
- TPUs
- Field-programmable gate arrays (FPGAs), and
- Application-specific integrated circuits (ASICs).

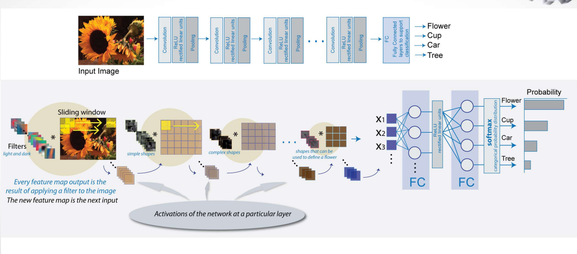
63

Hardware Landscape



64

Q & A



65