Lecture 4: Deep Learning

PUSL3123 AI and Machine Learning

Neamah Al-Naffakh

School of Engineering, Computing and Mathematics

Neamah.al-naffakh@plymouth.ac.uk

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Today's Topics

Deep Learning

Lesson learning outcomes: By the end of today's lesson, you would be able to:

- Explain the structure of a deep neural network and describe its potential uses
- Understand some of the difficulties involved in using deep learning
- Implement CNN

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Deep Learning Text Book

- Deep Learning, Goodfellow, Bengioand Courville, MIT Press, Nov. 2016

- Online version for free at https://www.deeplearningbook.org/

- Some slides in this lecture are adapted from my colleague **Dr. Lingfen Sun**

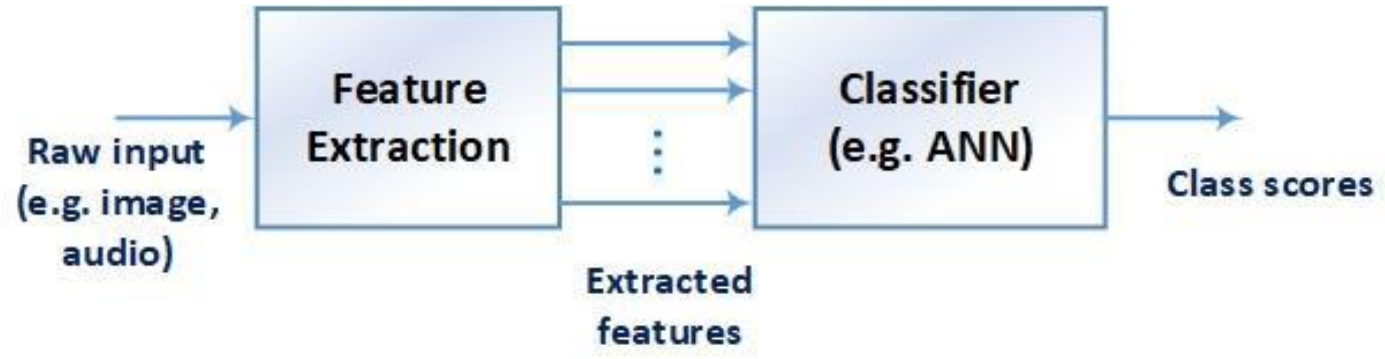
UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Introduction to Deep Learning

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Traditional Pattern Recognition

- Traditional Pattern Recognition consists of two parts
  - Feature extraction
  - Classifier (e.g. ANN)
- Feature extraction: need strong domain knowledge.
- Traditional classifiers were limited to low-dimensional space.
  - Classification performance is also affected by our understanding of features.



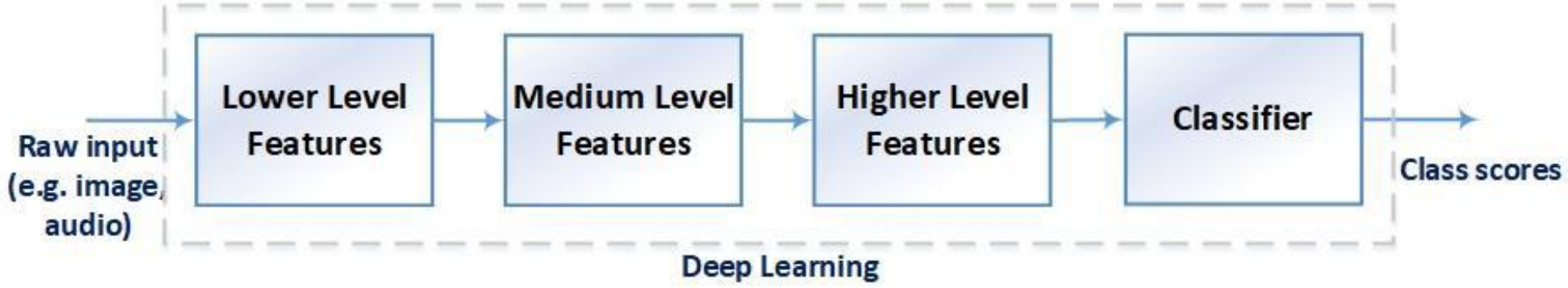
UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Deep Learning

- Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example

- Let the machine to learn the features from raw data, e.g. directly from pixel images, audio signal, EEG signal.

- Through many layers to extract different levels of features from raw data and then classify objects from these features.

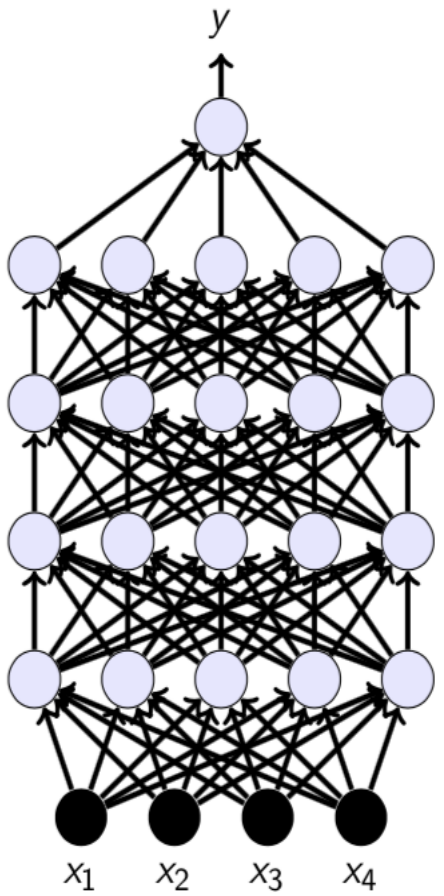


UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Deep Learning

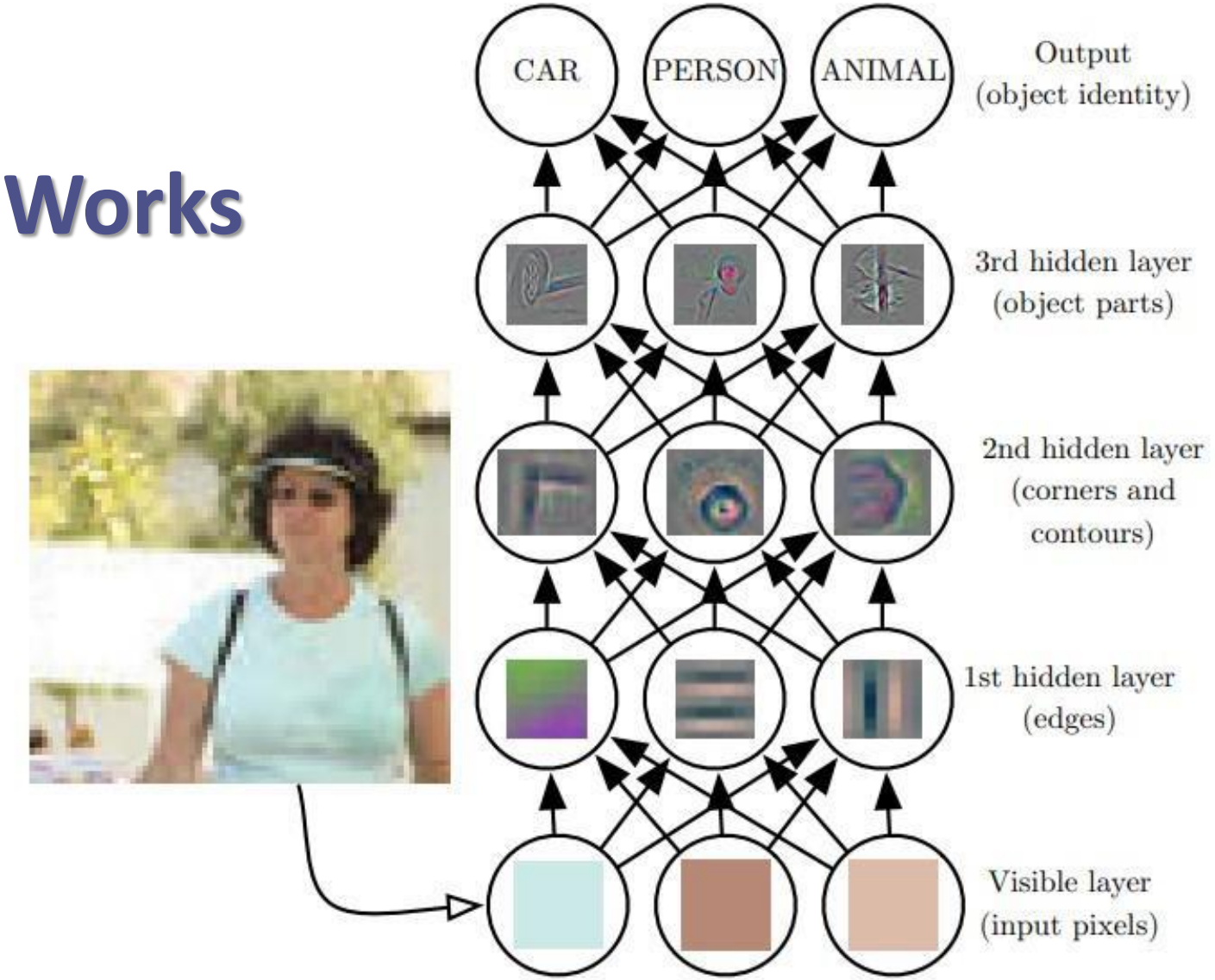
Neural networks with lots of layers

- Layers act as **feature extractors**
- Features are passed onto the next layer
- Raw input data is repeatedly processed before it reaches the output layer
- Result is a black box –so part of the drive behind **explainable AI**
- ANNs are prone to overfitting –so are deep learning models



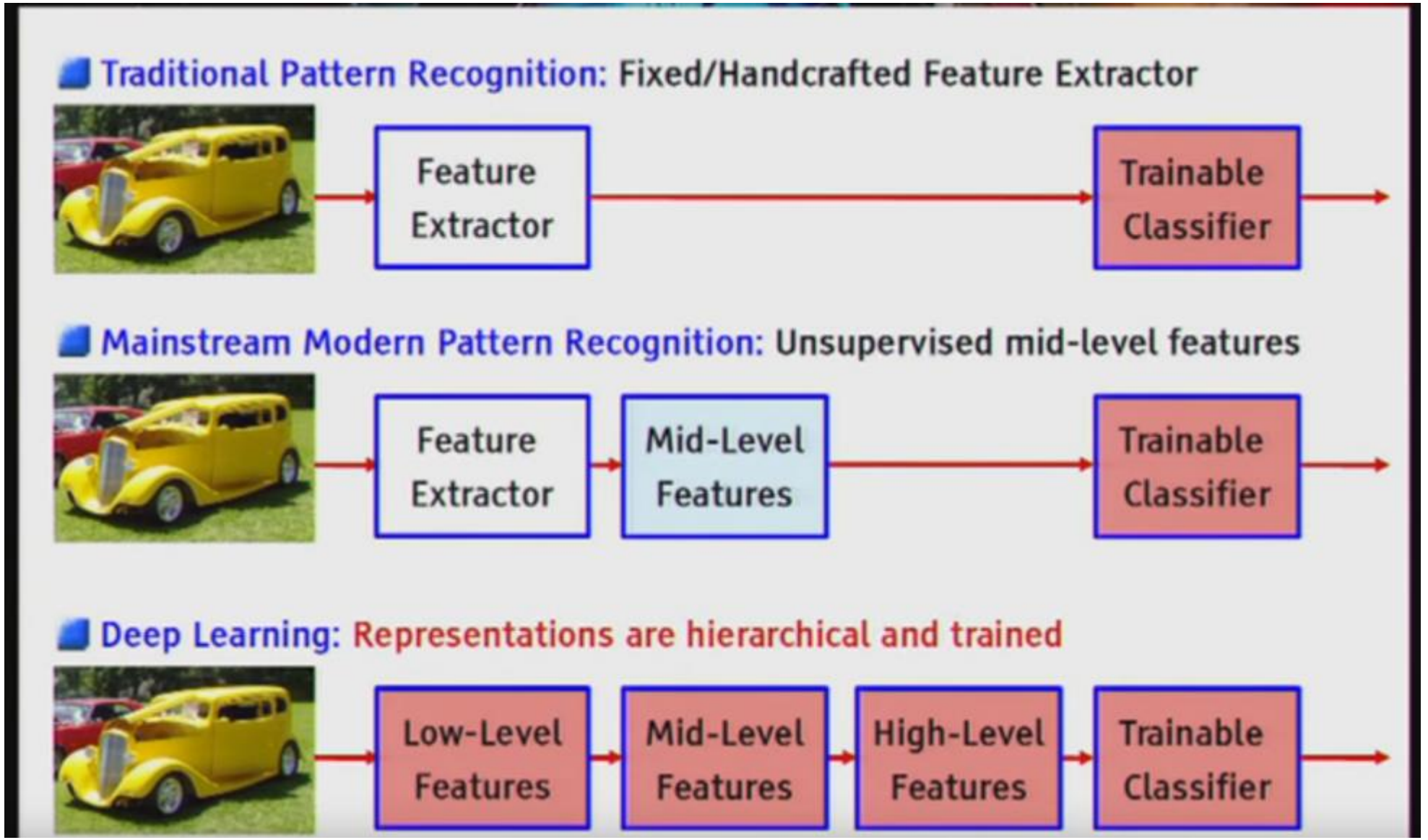
Deep Learning for Pattern Recognition

How It Works



UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

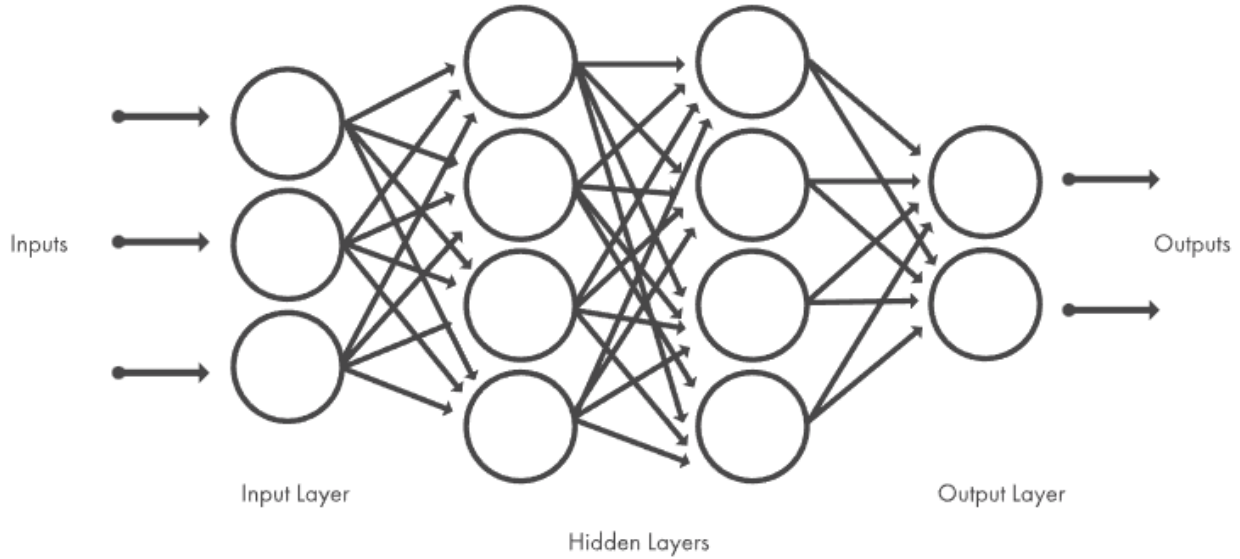
Deep Learning is Hierarchical Feature Learning



UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

How Deep Learning works

- Most deep learning methods use neural network architectures
- In comparison to the traditional neural networks (which contain 2-3 hidden layers), deep networks can have as many as 150.
- Deep learning models are trained by using large sets of labeled data and neural network architectures that learn features directly from the data without the need for manual feature extraction.

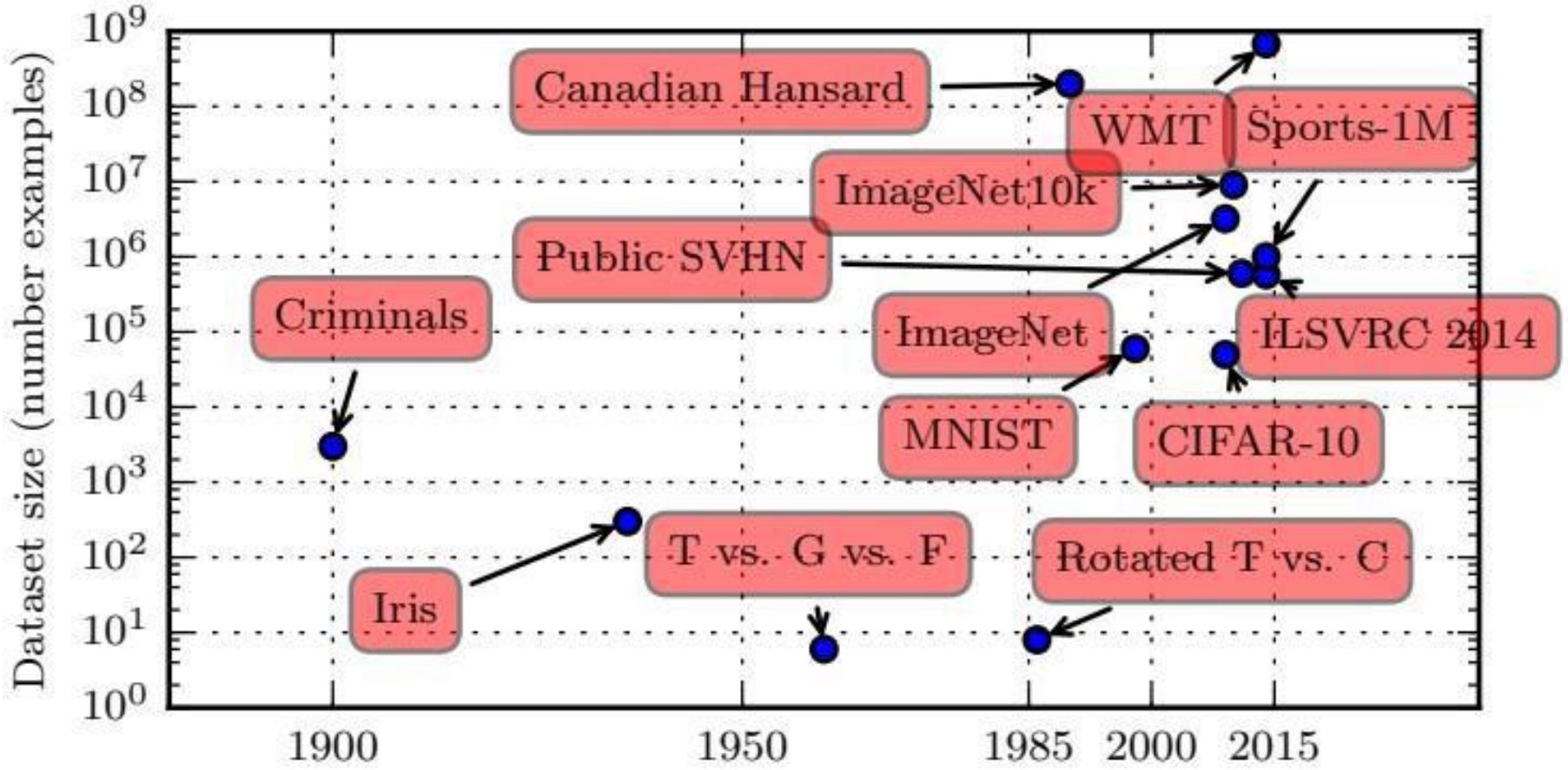


UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

The Growth of Deep Learning

- Increasing dataset sizes
- Increasing machine learning model sizes
- Availability of computational resources, especially GPU, which can handle high-dimensional inputs
- Availability of sufficient framework/platforms, such as TensorFlowFramework (open source) to coordinate the GPUs to work on complex ML tasks

Growing Dataset



MNIST: Modified NIST handwritten digits (0 –9) dataset, 70K samples
ImageNet: over 14M images

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

The MNIST Dataset

- The MNIST Dataset of handwritten digits, a subset of a larger set from NIST
- Training set: 60,000 samples; Test set: 10,000 samples
- Real data: digits written by high school students and employees of the US Census Bureau.
- Each image: 28x28 pixels, grey scale
- http://yann.lecun.com/exdb/mnist/

Sample example

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Image Net

- ImageNET: over 14M images, over 21K synsets indexed
- Average 1000 images per synset
- www.image-net.org
- Large resource for visual recognition research

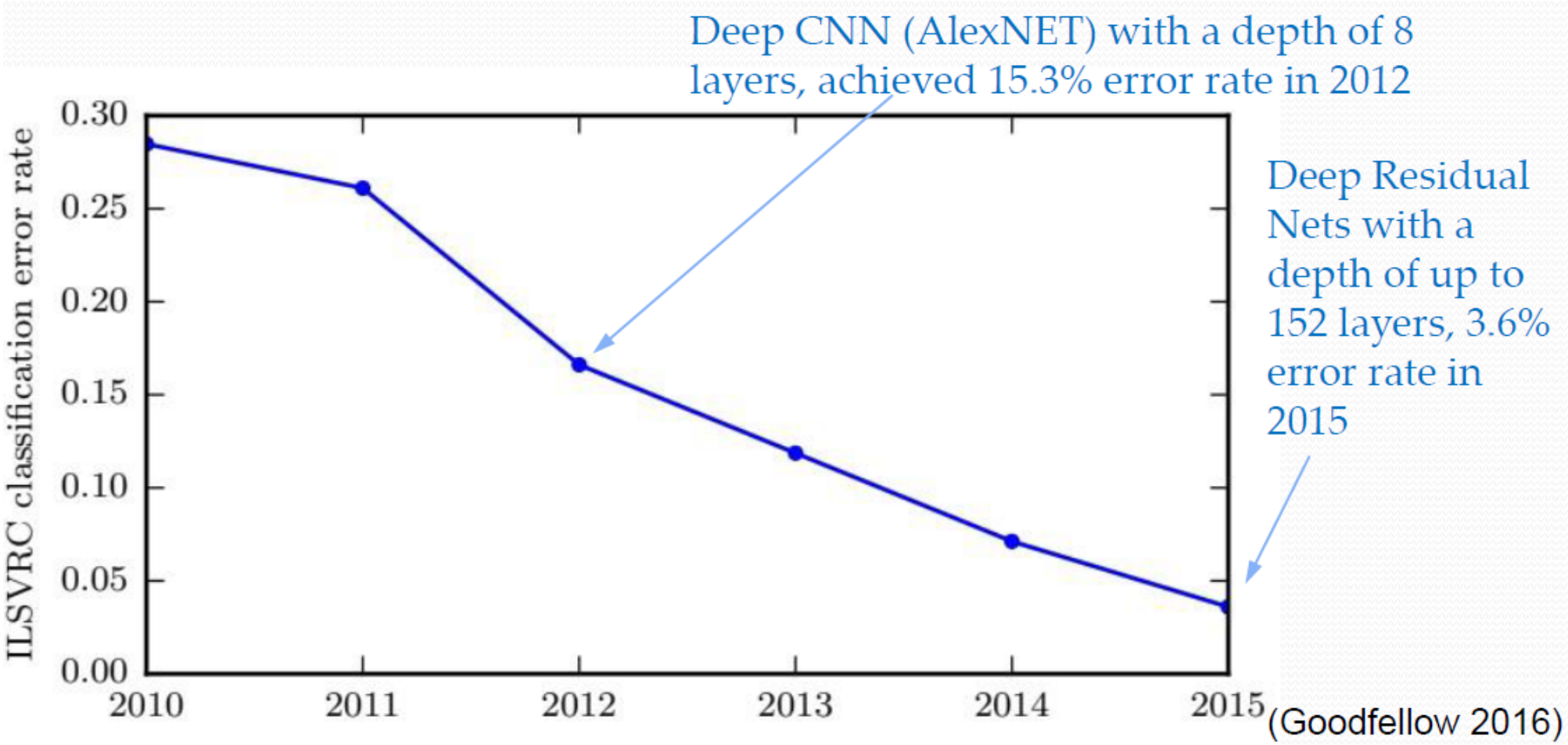
UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Deep Learning Applications

- Recognise things –based on CNNs (Convolutional Neural Networks), e.g.,
  - Object detection
  - Face recognition
  - Handwriting/character recognition
  - Speech recognition
  - Medical diagnosis
- Create things/creative ones based on GAN (Generative Adversarial Network), e.g.,
  - Draw pretty pictures/Fake images
  - Turn old black/white movie to HD movie
  - Data augmentation, e.g. medical data augmentation for diagnosis of certain disease

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Solving Object Recognition



ILSVRC: ImageNETLarge Scale Visual Recognition Challenge 2012
http://image-net.org/challenges/LSVRC/2012/supervision.pdf

K He, X Zhang, et al, Deep Residual Learning for Image Recognition, IEEE CVPR, 2016 (arxiv.org/abs/1512.03385)

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

How does deep learning attain such impressive results?

1) Deep learning requires large amounts of **labelled data**. For example, driverless car development requires millions of images and thousands of hours of video.

2) Deep learning requires substantial **computing power**. High-performance GPUs have a parallel architecture that is efficient for deep learning. When combined with clusters or cloud computing, this enables development teams to reduce training time for a deep learning network from weeks to hours or less.

How to Create and Train Deep Learning Models?

1) **Training from Scratch**

- Gather a very large labeled data set
- Design a network architecture that will learn the features and model.
- Less common approach because with the large amount of data and rate of learning, these networks typically take days or weeks to train.

2) **Feature Extraction**

- A slightly less common, more specialized approach to deep learning is to use the network as a feature extractor.
- Since all the layers are tasked with learning certain features from images, we can pull these features out of the network at any time

How to Create and Train Deep Learning Models?

3) Transfer learning

- It is a deep learning approach in which a model that has been trained for one task is used as a starting point for a model that performs a similar task.
- Benefits:
  - It trains models with less labelled data by reusing popular models that have already been trained on large datasets.
  - It can reduce training time and computing resources where the weights are not learned from scratch.
  - More details can be found here

Difficulties with deep learning

- Lots of data is required – this is becoming less of a challenge but there are still difficulties (e.g. getting labelled data).

- Training deep learning networks requires substantial computational power because of massive number of networks weights to calibrate – GPUs often used

- Deep neural networks are generally for supervised learning – if we want to achieve general AI we will need unsupervised learning models.

- "Deep" refers to the network architecture rather than understanding

Lab 5 Explanation

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Create and Train Deep Learning Models

```
% Deep Learning in 7 Lines of MATLAB Code
% Author Joe Hicklin
clear
picture=imread("PC.JPG");
nnet = alexnet; % Load the neural net
picture = imresize(picture,[227,227]); % Resize the picture
label = classify(nnet, picture); % Classify the picture
image(picture); % Show the picture
title(char(label)); % Show the label
drawnow;
```

Transfer Learning in 12 Lines of MATLAB Code

```
% Load Pre-trained Network (AlexNet)-- Author Joe Hicklin
% AlexNet is a pre-trained network trained on 1000 object categories.
alex = alexnet;
%% Review Network Architecture
layers = alex.Layers
% Modify Pre-trained Network
% AlexNet was trained to recognize 1000 classes, we need to modify it to
% Here we recognize just 2 classes.
layers(23) = fullyConnectedLayer(2); % change this based on # of classes
layers(25) = classificationLayer
allImages = imageDatastore('Images', 'IncludeSubfolders', true,'LabelSource', 'foldernames');
% Split data into training and test sets
[trainingImages, testImages] = splitEachLabel(allImages, 0.8, 'randomize');
%Perform Transfer Learning
opts = trainingOptions('sgdm', 'InitialLearnRate', 0.001,'MaxEpochs', 20, 'MiniBatchSize', 64);
```

Transfer Learning in 12 Lines of MATLAB Code

```
% Set custom read function
% One of the great things about imageDataStore it lets you specify a "custom" read function,
% in this case it is simply resizing the input images to 227x227 pixels which is what AlexNet expects.
% You can do this by specifying a function handle of a function with code to read and pre-process the image.
trainingImages.ReadFcn = @readFunctionTrain;

% Train the Network
% This process usually takes several minutes depending on the your GPU.
myNet = trainNetwork(trainingImages, layers, opts);

% Test Network Performance
% Now let's the test the performance of our new "snack recognizer" on the test set.
testImages.ReadFcn = @readFunctionTrain;
predictedLabels = classify(myNet, testImages);
accuracy = mean(predictedLabels == testImages.Labels)
```

15 Minutes Break

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Convolutional Neural Networks

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Convolutional Neural Networks

- Convolutional Neural Networks (CNNs or ConvNet), a specialised kind of Neural Networks for processing data with a grid-like topology
  - 1-D grid: taking samples at regular interval, e.g. speech signal
  - 2-D/3-D grid of pixels, such as 2-D/3-D image/video
- CNN is a network architecture for deep learning which learns directly from data, eliminating the need for manual feature extraction
- The pre-processing required in a CNN is much lower as compared to other classification algorithms

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Benefit of using CNN

- CNNs eliminate the need for manual feature extraction—the features are learned directly by the CNN.
- Reduce the feature dimensionality while maintaining high recognition rate.
- CNNs can be retrained for new recognition tasks, enabling you to build on pre-existing networks.
- CNNs Scale up neural networks to process very large images / video sequences

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

CNN Layers

1) Convolution Layer — The Kernel (feature detector): it puts the input images through a set of convolutional filters, each of which activates certain features from the images.

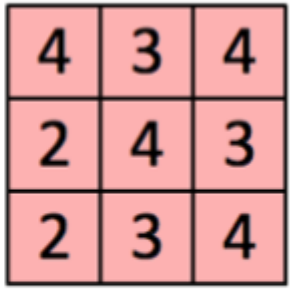
- The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction

```
Kernel/Filter, K =

1   0   1
0   1   0
1   0   1
```

X

Image



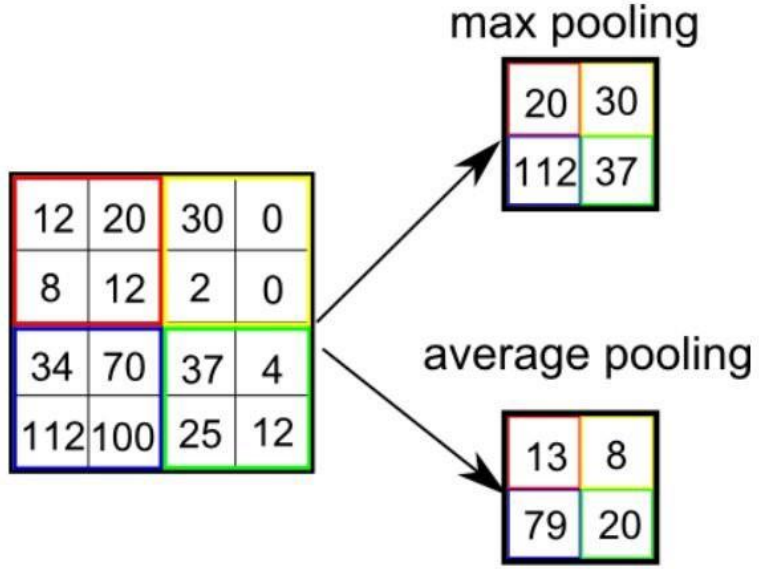
Convolved Feature

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

CNN Layers

2) **Pooling** simplifies the output by performing nonlinear downsampling, reducing the number of parameters that the network needs to learn.

- It is responsible for reducing the spatial size of the Convolved Feature
- It is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.
- Two types of Pooling: Max Pooling and Average Pooling. The former returns the maximum value from the portion of the image covered by the Kernel while the latter returns the average of all the values from the portion of the image covered by the Kernel.



CNN Layers

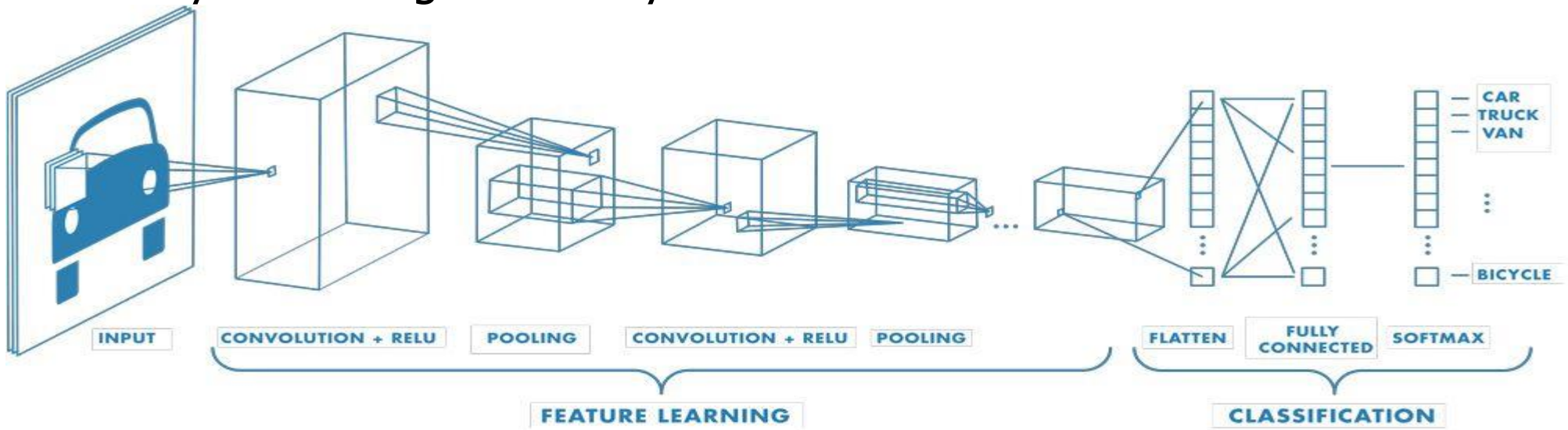
3) Rectified linear unit (ReLU) allows for faster and more effective training by mapping negative values to zero and maintaining positive values. This is sometimes referred to as activation, because only the activated features are carried forward into the next layer.

10	40
-5	4

→

10	40
0	4

These operations are repeated over tens or hundreds of layers, with each layer learning to identify different features



UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

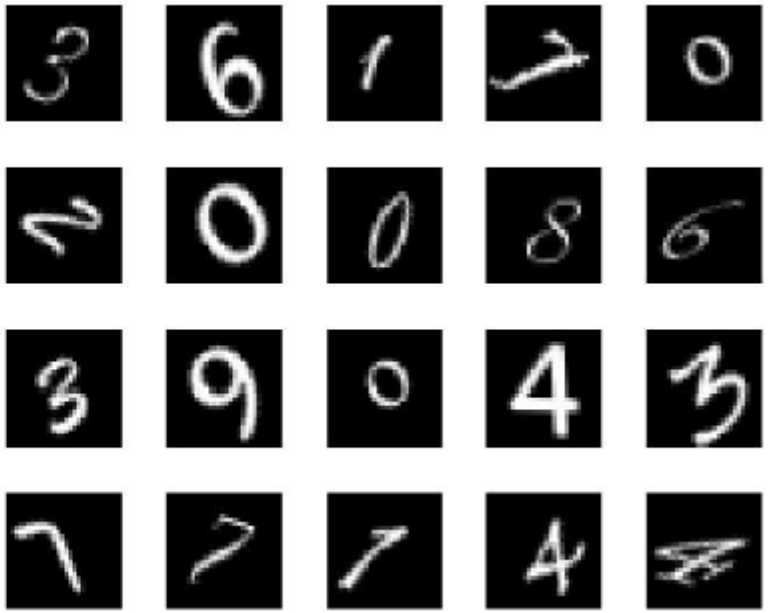
Lab 6 Explanation

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Matlab's Digit Database

- The digits data set consists of 10,000 synthetic grayscale images of handwritten digits.
- Each digit has 1000 images in respective folders
- Each image is 28x28 pixels with an associated label.
- C:\Program Files\MATLAB\R2022b\toolbox\nnet\nndemos\nndatasets\DigitDataset

Sample digits:
Randomly selected 20 images from the dataset and show



UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Preparing Training/Testing datasets

1) Read Data

- Training: take 750 images from each folder => total 7500 images
- Validation: remaining 250 for each digit, total 2500 images

```
% Giving path of dataset folder
digitDatasetPath = fullfile(matlabroot, 'toolbox', 'nnet',...
    'nndemos', 'nndatasets', 'DigitDataset');

%Reading Digit Images from Image Database Folder
imds = imageDatastore(digitDatasetPath, 'IncludeSubfolders', true,...
    'LabelSource', 'foldernames');

%Distributing Images in the set of Training and Testing
numTrain = 0.75; %75% for training; 25% for validation
[TrainImages, TestImages] = splitEachLabel(imds, numTrain, 'randomize');
```

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

Define Layers of CNN

2) Input Layer

- imageInputLayer(inputSize)
- Note: 28x28 is image size, 1 for grayscale (3 for colour image). Name for the layer is 'Input'. Default for other para.

```
imageInputLayer([28 28 1], 'Name', 'Input')
```

3) Convolutional Layer for feature extractions

- convolution2dLayer(filterSize, numFilters)
- Creates a 2D convolutional layer with 8 filters of size 3x3.

```
convolution2dLayer(3, 8, 'Name', 'Conv_1')
```

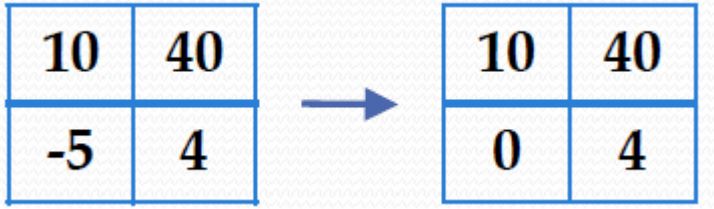
4) Batch Normalization Layer:

```
batchNormalizationLayer('Name' , 'BN_1')
```

Define Layers of CNN

5) ReLULayer to implement non-linear ReLU activation

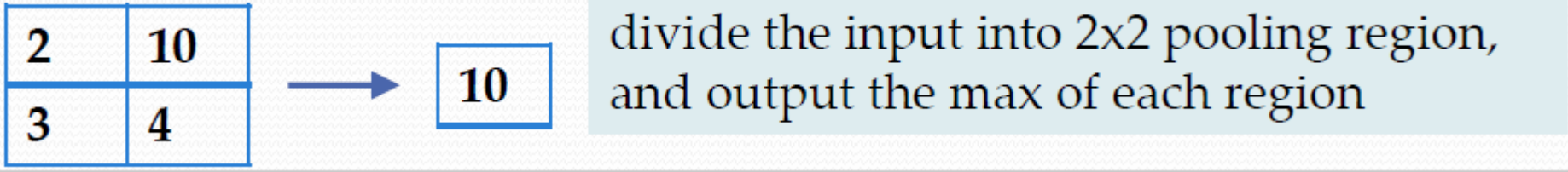
`reluLayer('Name', 'Relu_1')`



6) Max Poling Layer for down-sampling operation that reduces the spatial size of the feature map

- maxPooling2dLayer(PoolSize, 'Stride', n)
- Note: pooling size is 2x2; 'stride': step size from traversing vertically and horizontally.

`maxPooling2dLayer(2, 'Stride', 2, 'Name', 'MaxPool_1')`



Define Layers of CNN

7) Fully Connected Layer : combines features to classify the images of 10 classes

- fullyConnectedLayer(outputSize)

fullyConnectedLayer(10, 'Name', 'FC')

8) Softmax Layer: activation function for the output of the fully connected layer. The output consists of probability for each digit that sum to one, used for classification.

- softmaxLayer('Name', Name)

softmaxLayer('Name' , 'SoftMax')

9) Classification Layer: do classification and compute the loss

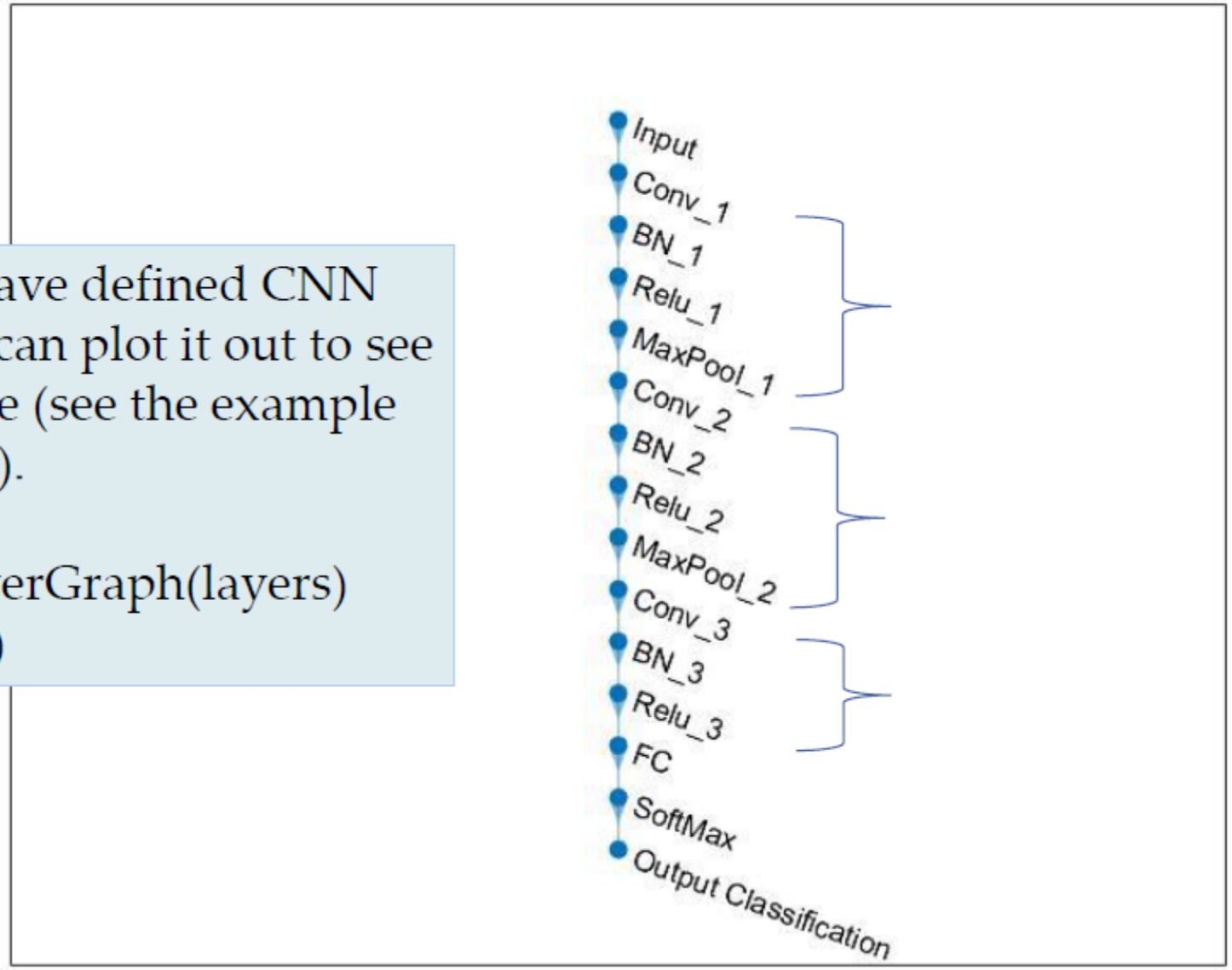
- classificationLayer('Name', Name)

classificationLayer('Name', 'Output Classification')

Plot CNN Network Structure

After you have defined CNN layers, you can plot it out to see the structure (see the example on the right).

```
lgraph = layerGraph(layers)
plot(lgraph)
```



Define Training Options

10) Training Parameters

- trainingOptions(solverName, Name, Value)
- Possible solverName:
  - ‘sgdm’ –Stochastic gradient descent with momentum
  - ‘rmsprop’ –Root mean square propagation
  - ‘adam’ –Adaptive moment estimation
- InitialLearnRate: 0.01 (default)
- LearnRateSchedule: none (default, learningRatedoes not change).

trainingOptions(‘sgdm’, ‘MaxEpochs’, 4, ‘ValidationData’, TestImages, ‘ValidationFrequency’, 30, ‘Plots’, ‘training-progress’)

…

- Hardware options:
- ‘ExecutionEnvironment’ –Hardware resource for training network
- ‘auto’(default) |’cpu’ |’gpu’|’multi-gpu’|’parallel’

Train CNN

After you have defined CNN layer architecture and network options, you can

11) Train the Network

```
layers = [imageInputLayers([28 28 1]), ‘Name’, ‘Input’)
…]

options = trainingOptions(‘sgdm’, ‘MaxEpoches’, 4,
 …)

trainedNet= trainNetwork(TrainImages, layers, options)

save trainedNet     %save the trained network for future use
```

MATLAB Code

```
clear all, clc, close all;
% Path of the dataset
digitDatasetPath=fullfile(matlabroot,'toolbox','nnet',...
    'nndemos','nndatasets','DigitDataset');

% Reading Image data from the Image database folder
imds=imageDatastore(digitDatasetPath, 'IncludeSubfolders', true,...
    'LabelSource', 'foldernames');

numTrain=0.75;
[trainImages, testImages] = splitEachLabel(imds, numTrain, 'randomize');
%% Creating CNN
layers = [
    imageInputLayer([28 28 1],'Name','Input')
    convolution2dLayer(3,8,'Padding','same','Name','Conv_1')
    batchNormalizationLayer('Name','BN_1')
    reluLayer('Name','Relu_1')
    maxPooling2dLayer(2,'Stride',2,'Name','MaxPool_1')

    convolution2dLayer(3,16,'Padding','same','Name','Conv_2')
    batchNormalizationLayer('Name','BN_2')
    reluLayer('Name','Relu_2')
    maxPooling2dLayer(2,'Stride',2,'Name','MaxPool_2')

    convolution2dLayer(3,32,'Padding','same','Name','Conv_3')
    batchNormalizationLayer('Name','BN_3')
    reluLayer('Name','Relu_3')

    fullyConnectedLayer(10,'Name','FC')
    softmaxLayer('Name','SoftMax')
    classificationLayer('Name','Output Classification')
    ];
```

MATLAB Code

```
lgraph=layerGraph(layers);
% Ploting Network Structure
plot(lgraph);    %
% Training Option
options=trainingOptions('sgdm','MaxEpochs',4,...
    'ValidationData',testImages,'ValidationFrequency',30,...
    'Plots','training-progress');

% Training Network
trainedNet=trainNetwork(trainImages,layers,options);
CNNDigit1=trainedNet
save CNNDigit1

% You can load the trained net, to test on test images
load CNNDigit1;
net = CNNDigit1;
I =imread ('image1037.png');
figure;
imshow (I);
label = classify(net, I);
disp (label);
title(['The digit is ' char(label)]);
```

Summary

- **Deep neural networks**
  - Lots of hidden layers
  - Layers have specific roles –e.g. **convolution** and **pooling**
  - The more data, the better the deep learning models perform
  - it is recommended to use deep neural networks with many layers to ensure getting the best performance in terms of accuracy and other important metrics respectively

- **ImageNet**
  - Massive dataset for object recognition and challenge 2010-2017