University of Salford MANCHESTER

Artificial Intelligent – Week 9

Dr Salem Ameen

S.A.AMEEN1@SALFORD.AC.UK

Week 9: Artificial Intelligent – Machine Learning

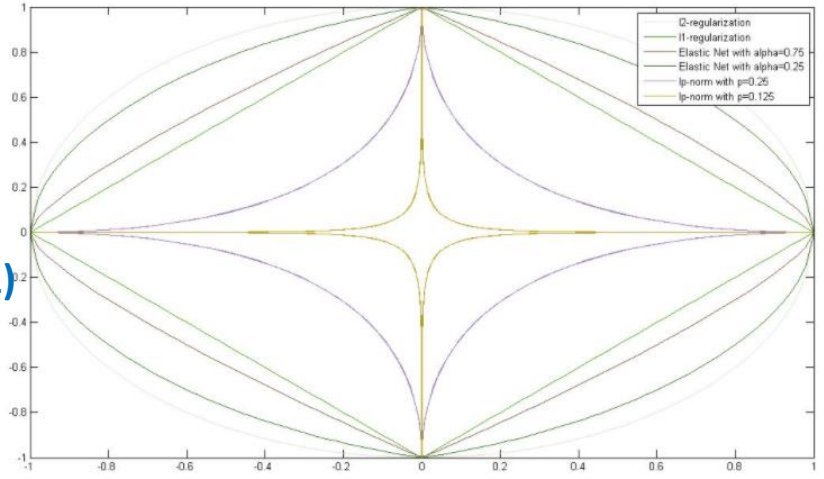
University of Salford MANCHESTER

Loss and Regularizer	Comments
Ordinary Least Squares $\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^\top x_i - y_i)^2$	◦ Squared Loss ◦ No Regularization ◦ Closed form solution: ◦ $\mathbf{w} = (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y}^\top$ ◦ $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ ◦ $\mathbf{y} = [y_1, \ldots, y_n]$
Ridge Regression $\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^\top x_i - y_i)^2 + \lambda\|w\|_2^2$	◦ Squared Loss ◦ l_2-Regularization ◦ $\mathbf{w} = (\mathbf{X}\mathbf{X}^\top + \lambda\mathbb{I})^{-1}\mathbf{X}\mathbf{y}^\top$
Lasso $\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \lambda\|\mathbf{w}\|_1$	◦ + sparsity inducing (good for feature selection) ◦ + Convex ◦ - Not strictly convex (no unique solution) ◦ - Not differentiable (at 0) ◦ Solve with (sub)-gradient descent or SVEN
Elastic Net $\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \alpha\|\mathbf{w}\|_1 + (1-\alpha)\|\mathbf{w}\|_2^2$ $\alpha \in [0, 1)$	◦ ADVANTAGE: Strictly convex (i.e. unique solution) ◦ + sparsity inducing (good for feature selection) ◦ + Dual of squared-loss SVM, see SVEN ◦ DISADVANTAGE: - Non-differentiable

$$\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n} \underbrace{l(h_{\mathbf{w}}(\mathbf{x}_i), y_i)}_{\text{Loss}} + \underbrace{\lambda r(w)}_{\text{Regularizer}},$$

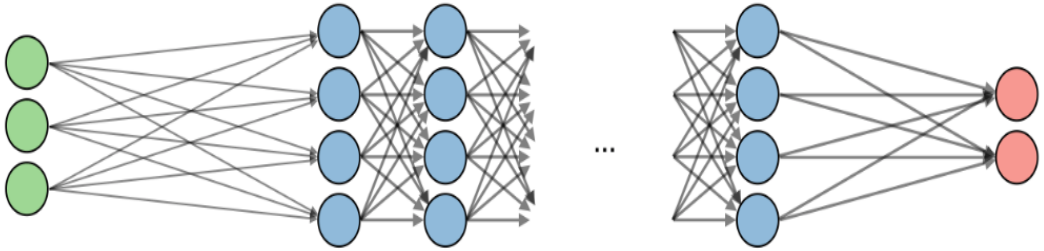
Loss and Regularizer	Comments
Logistic Regression $\min_{\mathbf{w},b} \frac{1}{n}\sum_{i=1}^{n}\log\left(1 + e^{-y_i(\mathbf{w}^\top \mathbf{x}_i + b)}\right)$	◦ Often l_1 or l_2 Regularized ◦ Solve with gradient descent. ◦ $\Pr(y\|x) = \frac{1}{1+e^{-y(\mathbf{w}^\top x + b)}}$
Linear Support Vector Machine $\min_{\mathbf{w},b} C\sum_{i=1}^{n}\max[1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b), 0] + \|\mathbf{w}\|_2^2$	◦ Typically l_2 regularized (sometimes l_1). ◦ Quadratic program. ◦ When kernelized leads to **sparse** solutions. ◦ Kernelized version can be solved very efficiently with specialized algorithms (e.g. SMO)

tf.keras.regularizers.L1(0.3)
tf.keras.regularizers.L2(0.1)
tf.keras.regularizers.L1L2(l1=0.01, l2=0.01)



Week 9: Artificial Intelligent – Deep Learning

Features									output
x1	x2	x3	x4	x5	x6	x6	x7	x8	y



Input layer Hidden layer 1 ... Hidden layer k Output layer

By noting i the i^{th} layer of the network and j the j^{th} hidden unit of the layer, we have:

$$z_j^{[i]} = w_j^{[i]^T} x + b_j^{[i]}$$

Week 9: Artificial Intelligent – Deep Learning

University of Salford MANCHESTER

- **Architecture of a CNN (Computer Vision):**

- **Architecture of a RNN (Natural Language Processing):**

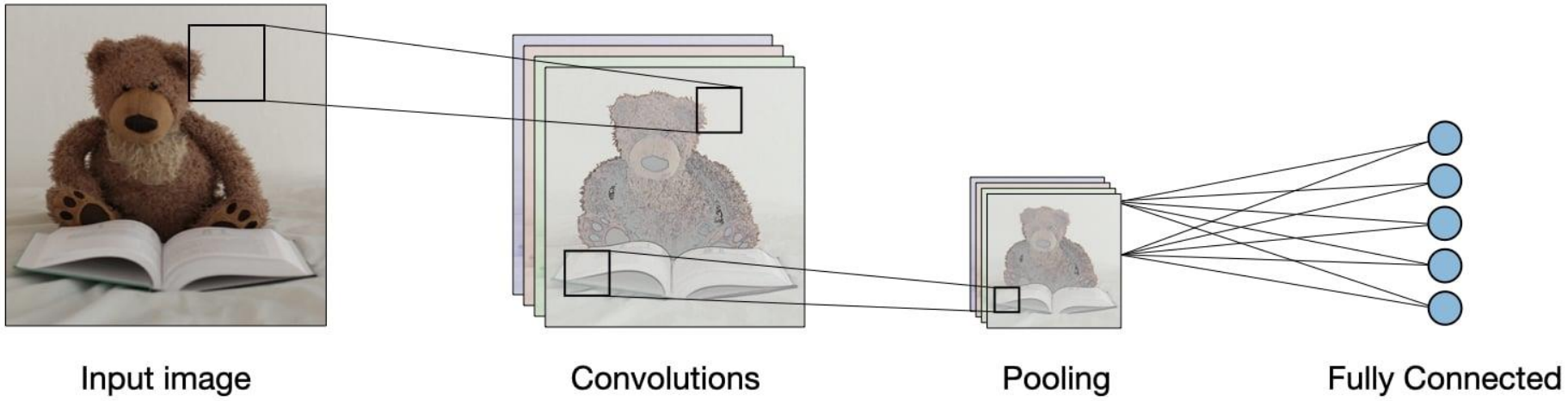
Week 9: Artificial Intelligent – Deep Learning – CNN - Vision

University of Salford MANCHESTER

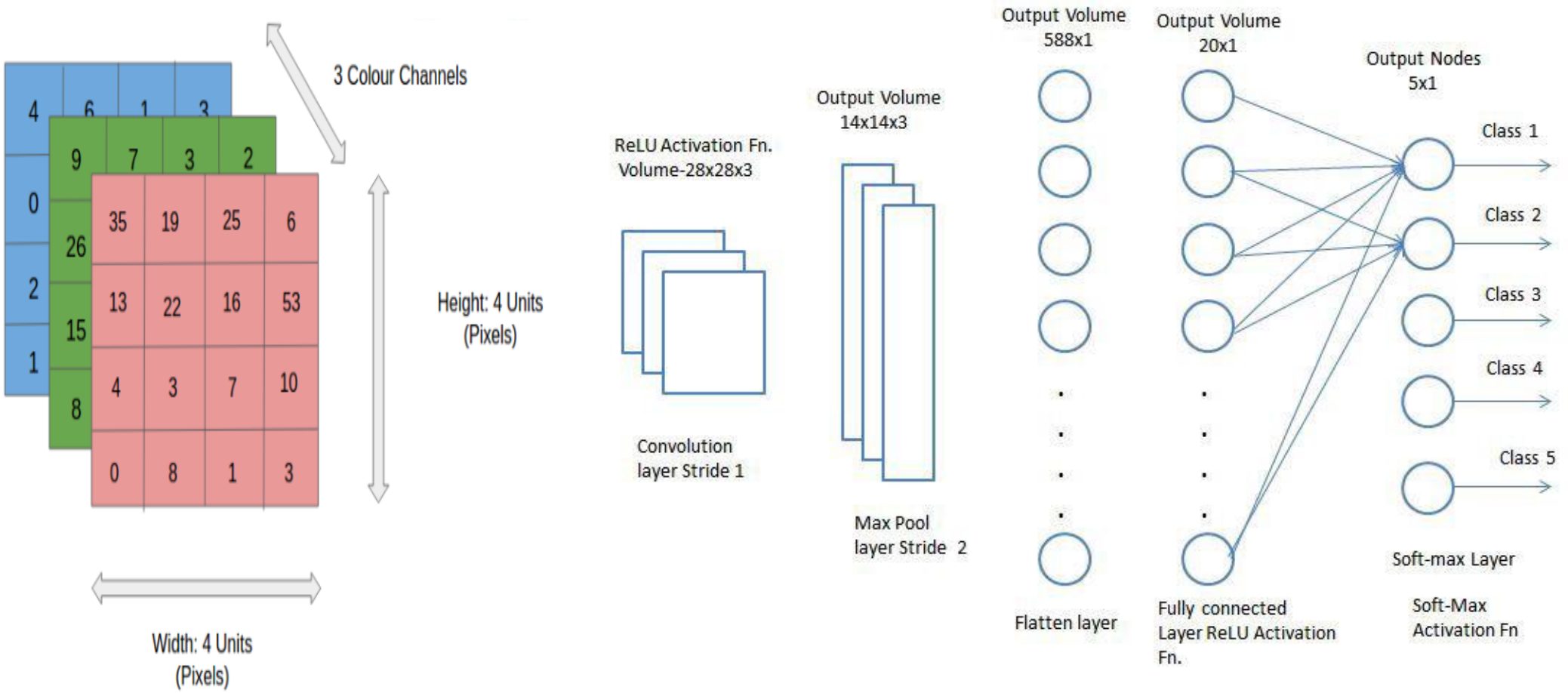
- **Architecture of a traditional CNN:** Convolutional neural networks, also known as CNNs, are a specific type of neural networks that are generally composed of the following layers:

Week 9: Artificial Intelligent – Deep Learning – CNN - Vision

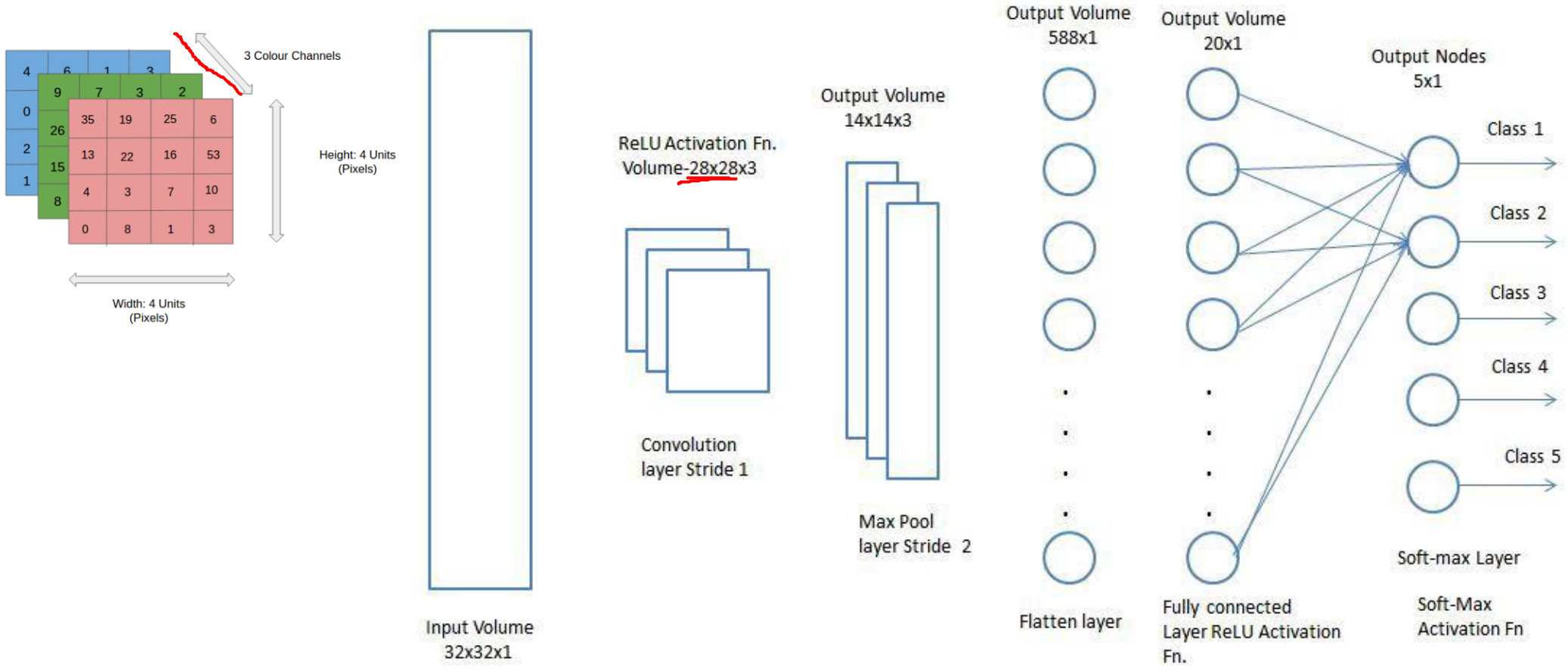
- **Architecture of a traditional CNN:** Convolutional neural networks, also known as CNNs, are a specific type of neural networks that are generally composed of the following layers:



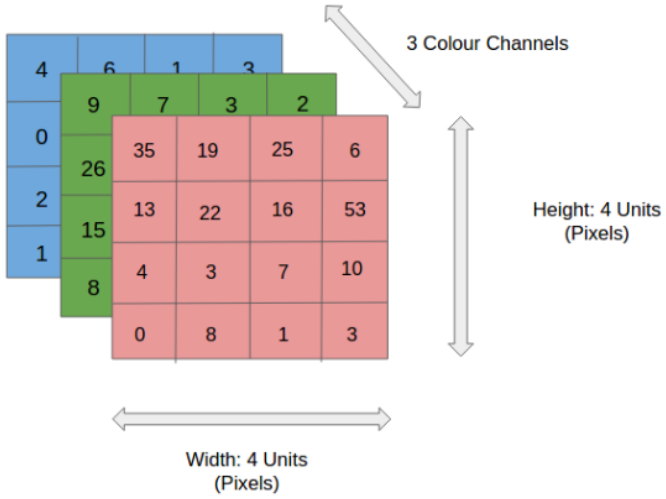
Week 9: Artificial Intelligent – Deep Learning – CNN - Vision



Week 9: Artificial Intelligent – Deep Learning – CNN - Vision



Week 9: Artificial Intelligent – CNN – Convolution layer



0	0	0	0	0	0	...
0	156	155	156	158	158	...
0	153	154	157	159	159	...
0	149	151	155	158	159	...
0	146	146	149	153	158	...
0	145	143	143	148	158	...
...	...	...	...	...	...	...

Input Channel #1 (Red)

0	0	0	0	0	0	...
0	167	166	167	169	169	...
0	164	165	168	170	170	...
0	160	162	166	169	170	...
0	156	156	159	163	168	...
0	155	153	153	158	168	...
...	...	...	...	...	...	...

Input Channel #2 (Green)

0	0	0	0	0	0	...
0	163	162	163	165	165	...
0	160	161	164	166	166	...
0	156	158	162	165	166	...
0	155	155	158	162	167	...
0	154	152	152	157	167	...
...	...	...	...	...	...	...

Input Channel #3 (Blue)

-1	-1	1
0	1	-1
0	1	1

Kernel Channel #1

1	0	0
1	-1	-1
1	0	-1

Kernel Channel #2

0	1	1
0	1	0
1	-1	1

Kernel Channel #3

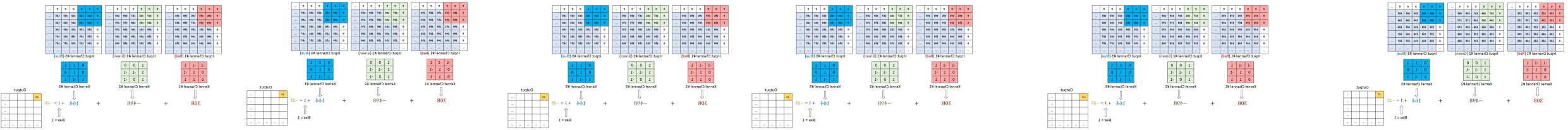
$308 + (-498) + 164 + 1 = -25$

Bias = 1

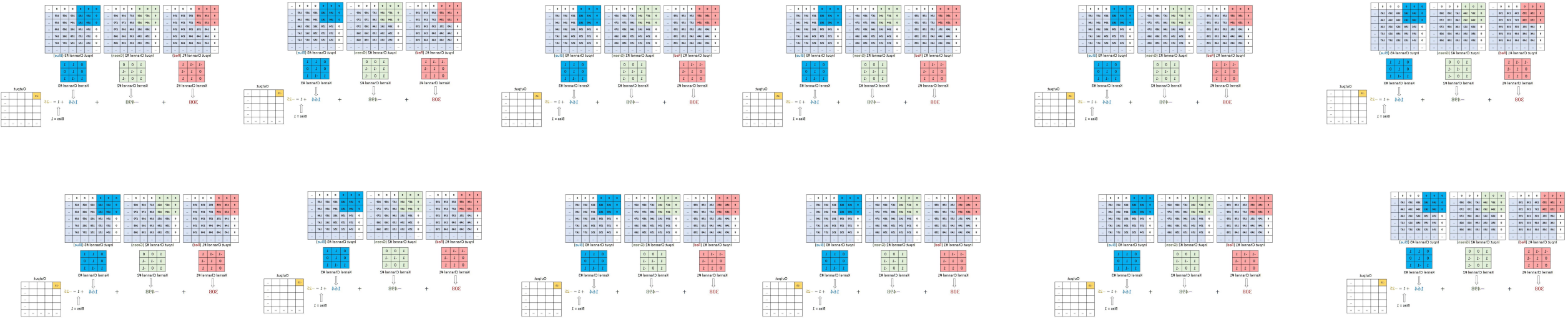
Output

-25				...
				...
				...
				...
...	...	...	...	...

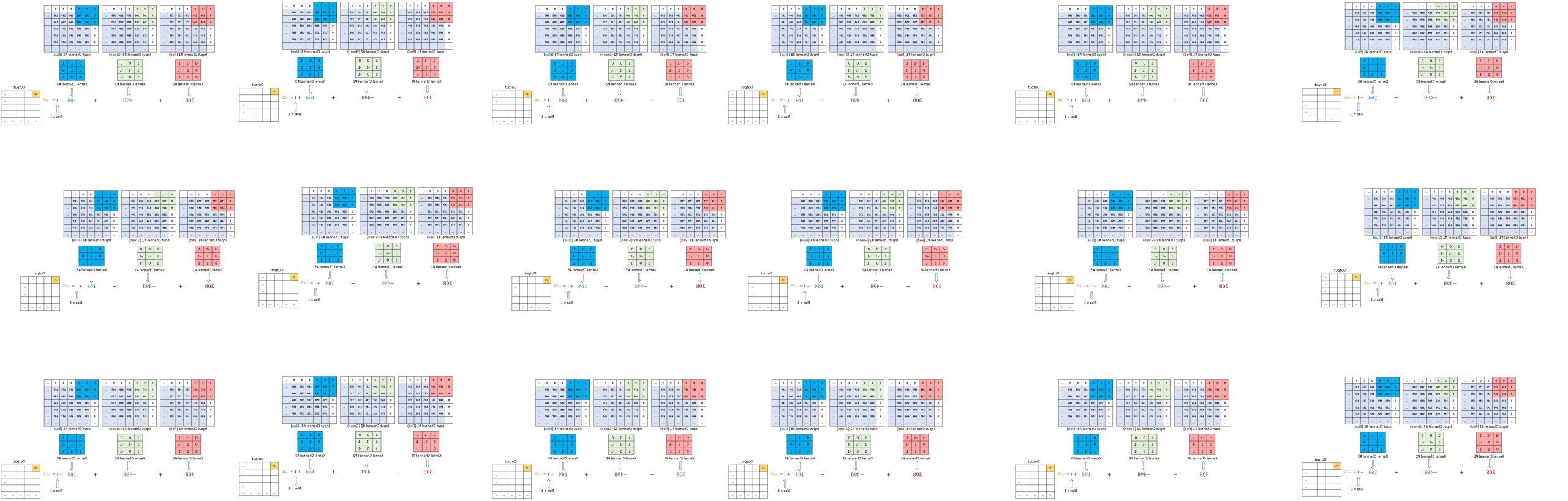
Week 9: Artificial Intelligent – CNN – Convolution layer



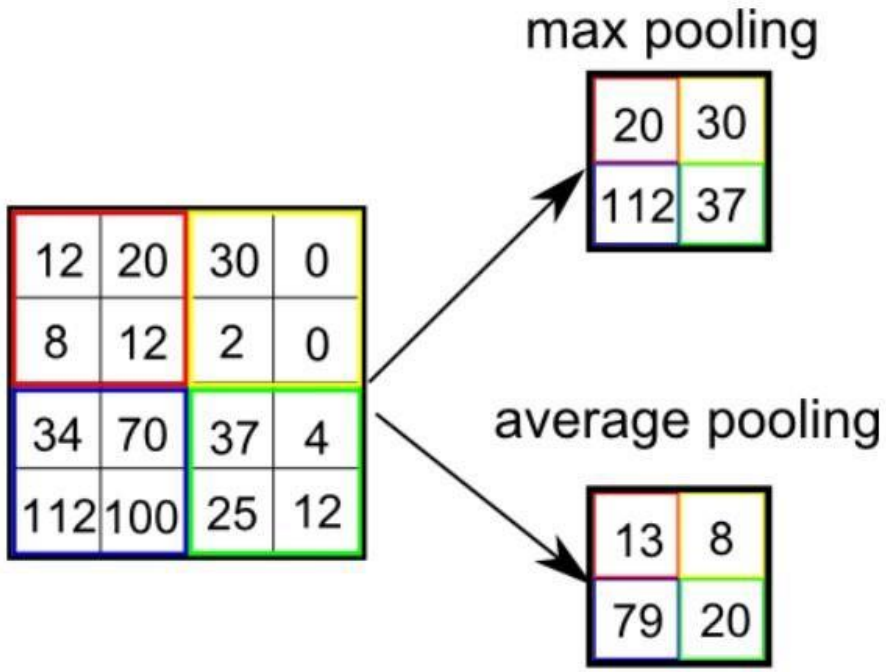
Week 9: Artificial Intelligent – CNN – Convolution layer

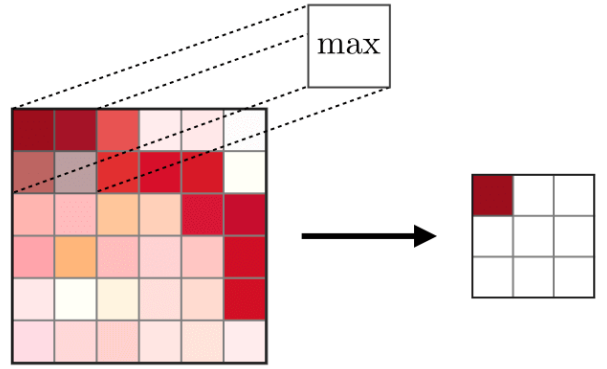


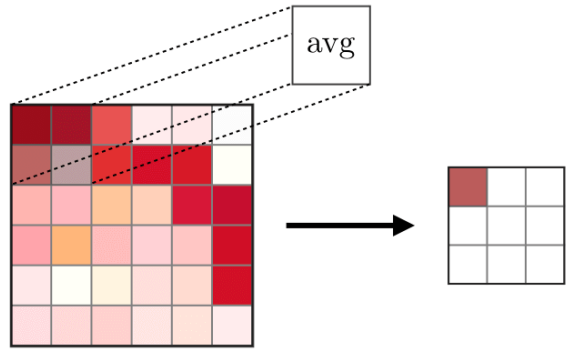
Week 9: Artificial Intelligent – CNN – Convolution layer



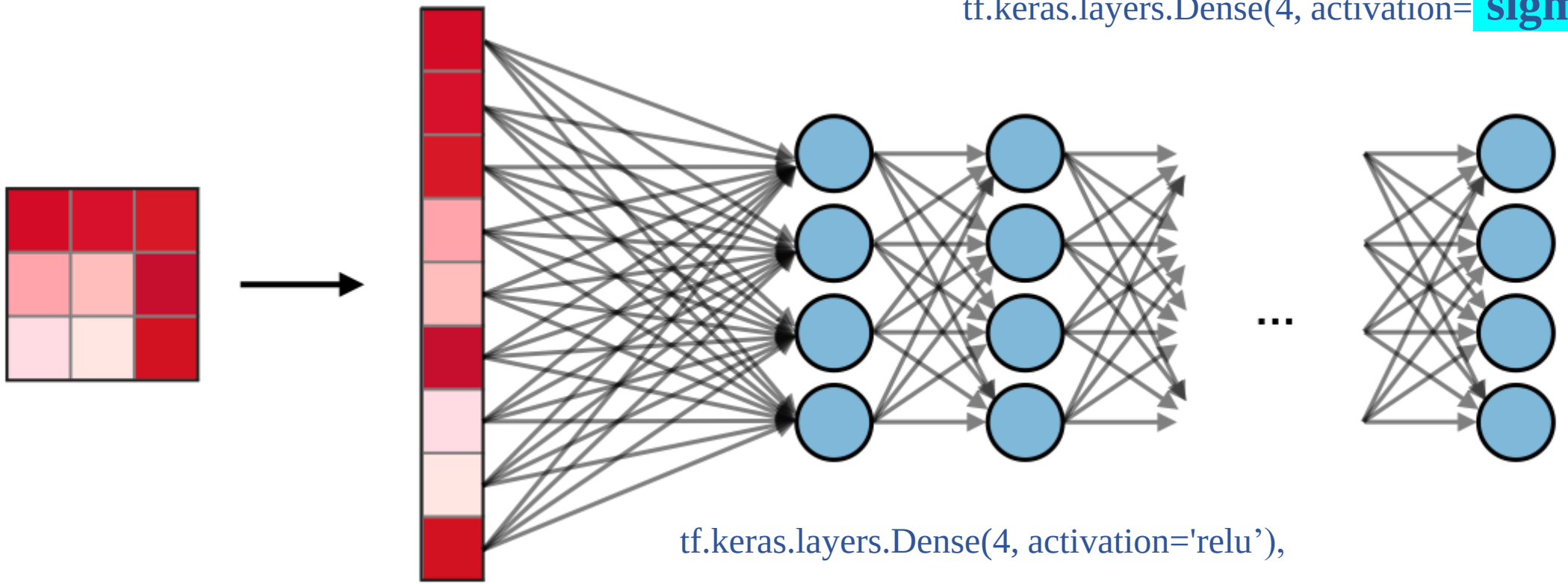
Week 9: Artificial Intelligent – CNN – Pooling Layer





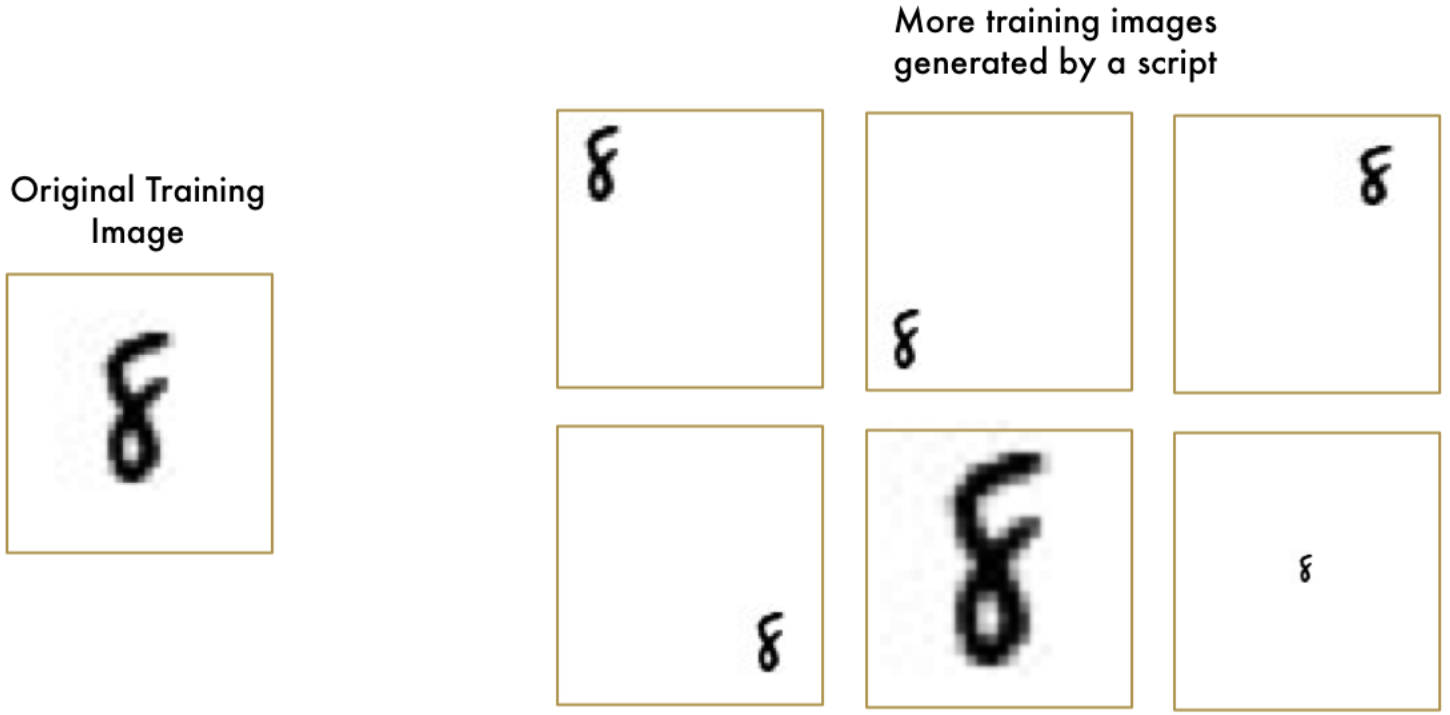


Week 9: Artificial Intelligent – CNN – Fully Connected Layer



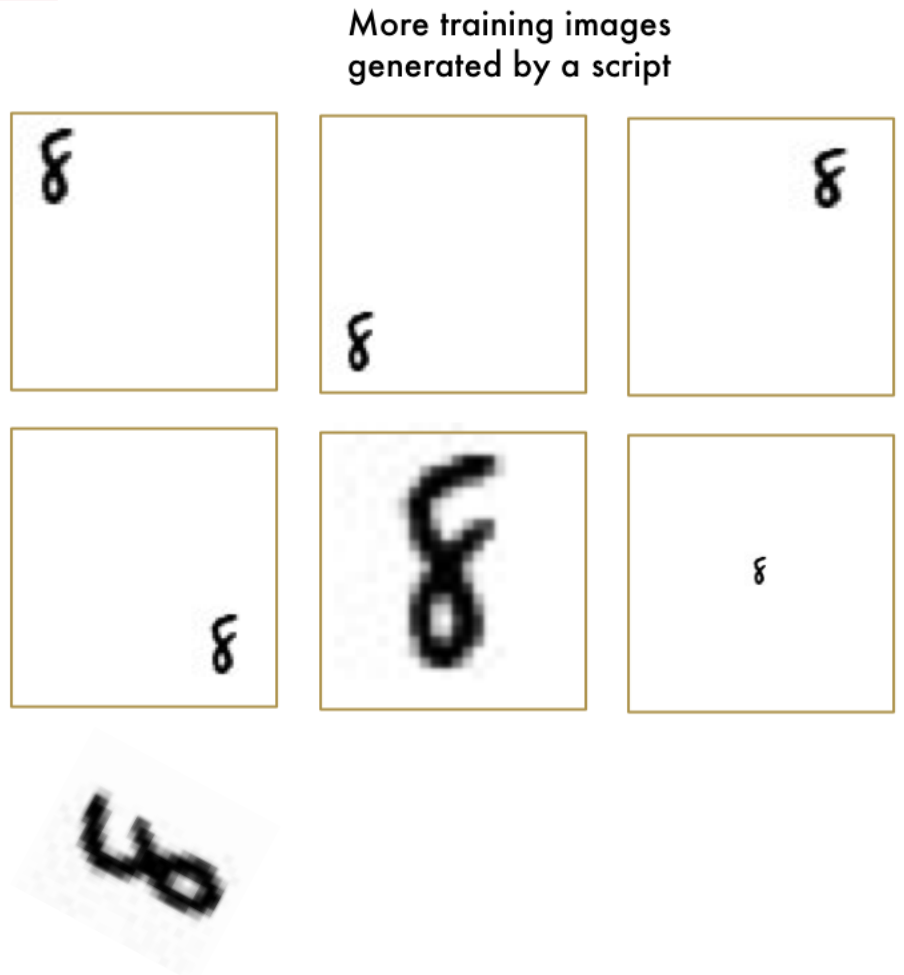
Week 9: Artificial Intelligent – CNN – Data processing

Week 9: Artificial Intelligent – CNN – Data processing



Week 9: Artificial Intelligent – CNN – Data processing

```
from tensorflow.keras.preprocessing.image import
ImageDataGenerator

# All images will be rescaled by 1./255
train_datagen = ImageDataGenerator(
      rescale=1./255,
      rotation_range=40,
      width_shift_range=0.2,
      height_shift_range=0.2,
      shear_range=0.2,
      zoom_range=0.2,
      horizontal_flip=True,
      fill_mode='nearest')

validation_datagen = ImageDataGenerator(rescale=1/255)
```



Week 9: Artificial Intelligent – CNN – Data processing

University of Salford MANCHESTER

- **Data augmentation**: Deep learning models usually need a lot of data to be properly trained. It is often useful to get more data from the existing ones using data augmentation techniques.

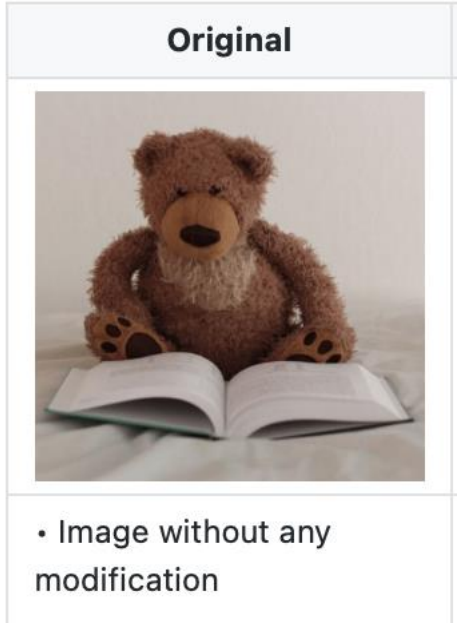
Session Code:
2675
9432

Week 9: Artificial Intelligent – CNN – Data processing

- **Data augmentation**: Deep learning models usually need a lot of data to be properly trained. It is often useful to get more data from the existing ones using data augmentation techniques.
- The main ones are summed up in the table below. More precisely, given the following input image,

Week 9: Artificial Intelligent – CNN – Data processing

- **Data augmentation**: Deep learning models usually need a lot of data to be properly trained. It is often useful to get more data from the existing ones using data augmentation techniques.
- The main ones are summed up in the table below. More precisely, given the following input image, here are the techniques that we can apply:



Week 9: Artificial Intelligent – CNN – Data processing

University of Salford MANCHESTER

Original	Flip	Rotation	Random crop
			
• Image without any modification	• Flipped with respect to an axis for which the meaning of the image is preserved	• Rotation with a slight angle • Simulates incorrect horizon calibration	• Random focus on one part of the image • Several random crops can be done in a row

Color shift	Noise addition	Information loss	Contrast change
			
• Nuances of RGB is slightly changed • Captures noise that can occur with light exposure	• Addition of noise • More tolerance to quality variation of inputs	• Parts of image ignored • Mimics potential loss of parts of image	• Luminosity changes • Controls difference in exposition due to time of day

Week 9: Artificial Intelligent – CNN – Modelling

University of Salford MANCHESTER

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(300, 300, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Week 9: Artificial Intelligent – CNN – Training

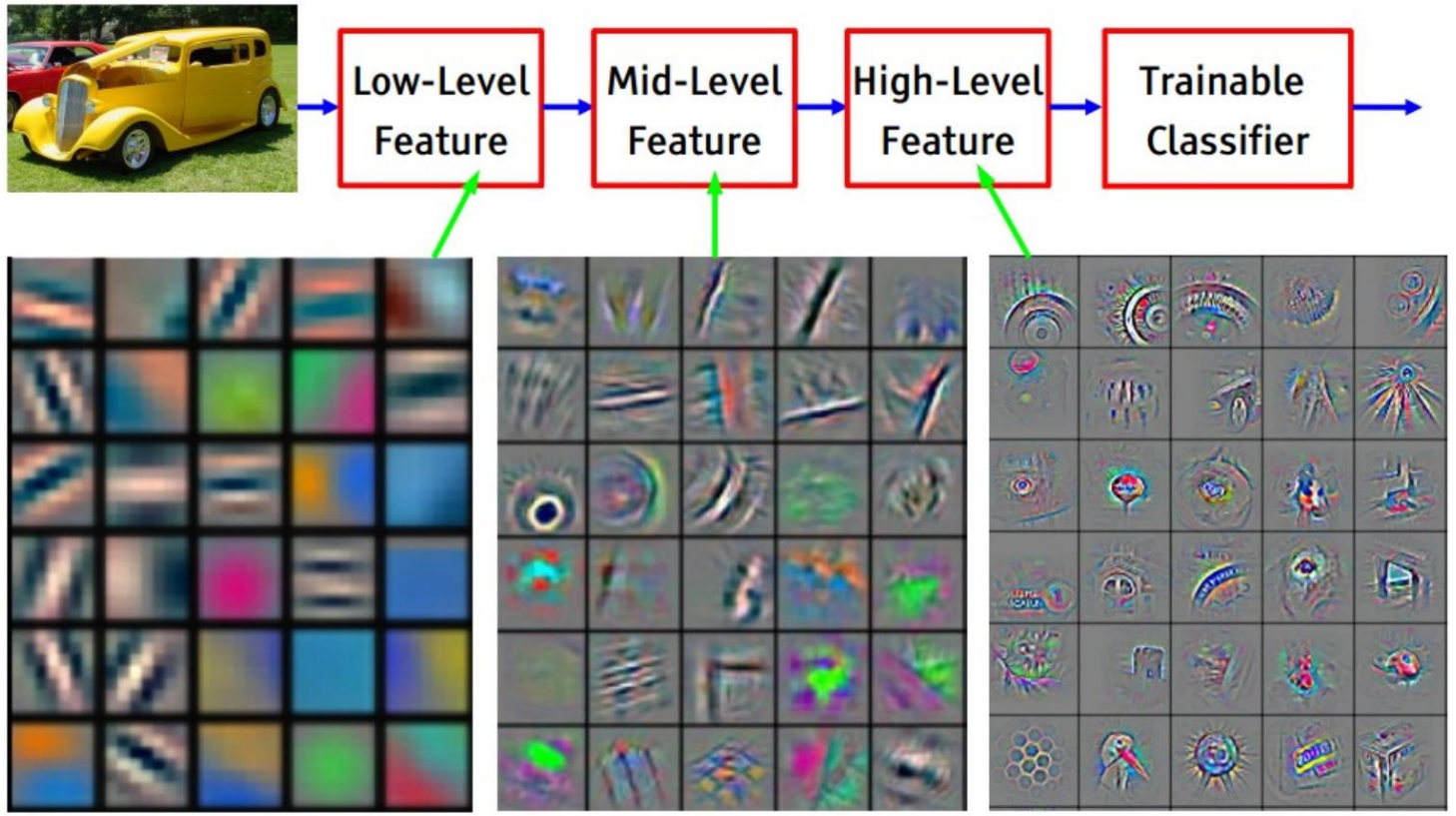
University of Salford MANCHESTER

from tensorflow.keras.optimizers import Adam, **SGD, RMSProp, RMSProp, Adagrad**

model.compile(loss='binary_crossentropy', optimizer=Adam, metrics=['accuracy'])

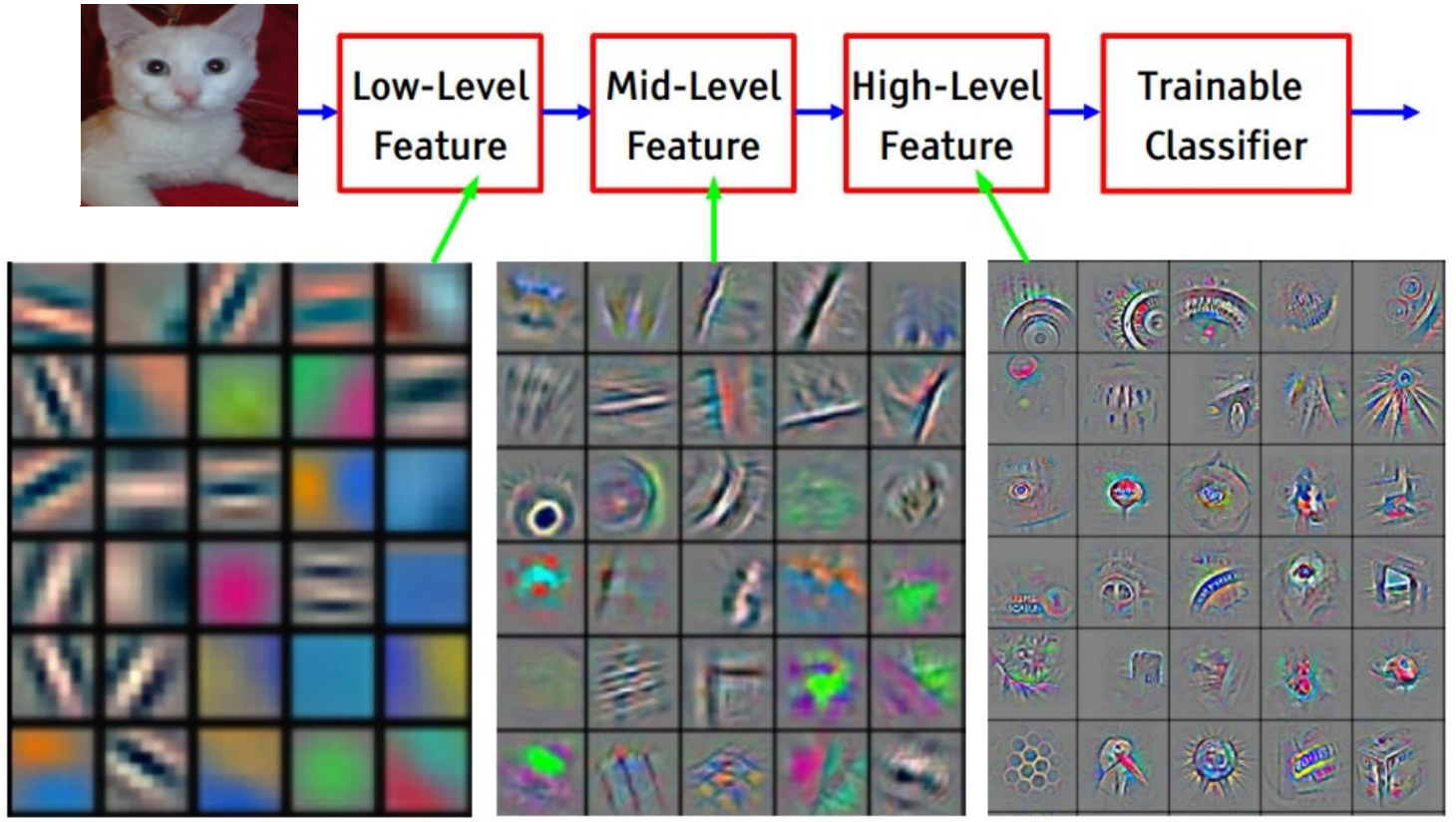
history = model.fit(train_generator, steps_per_epoch=8, epochs=30,verbose=1,
validation_data = validation_generator, validation_steps=8)

Week 9: Artificial Intelligent – CNN – Feature Detector



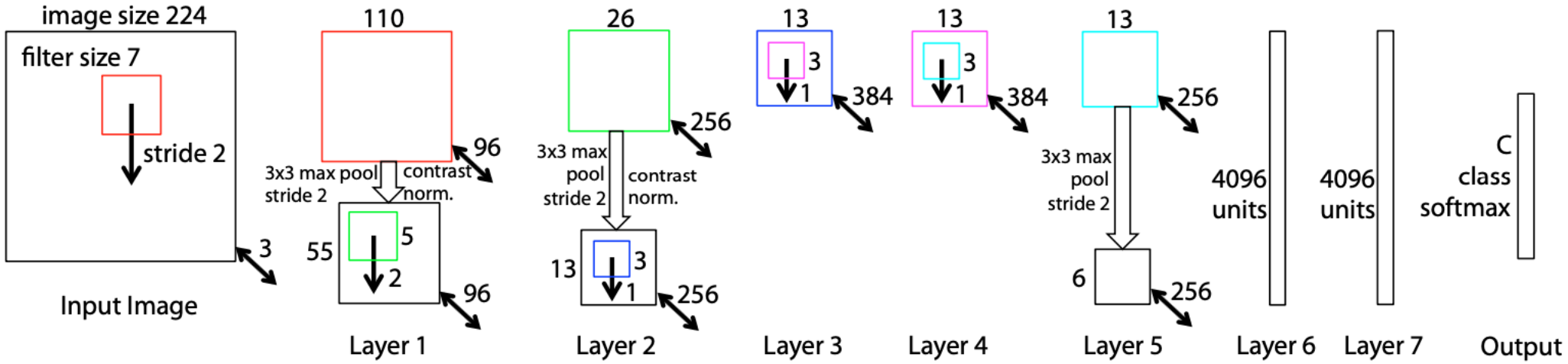
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Week 9: Artificial Intelligent – CNN – Feature Detector



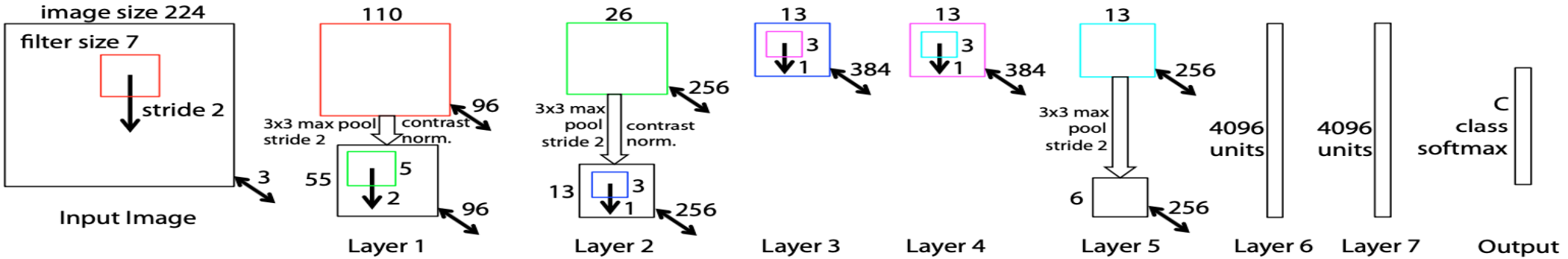
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

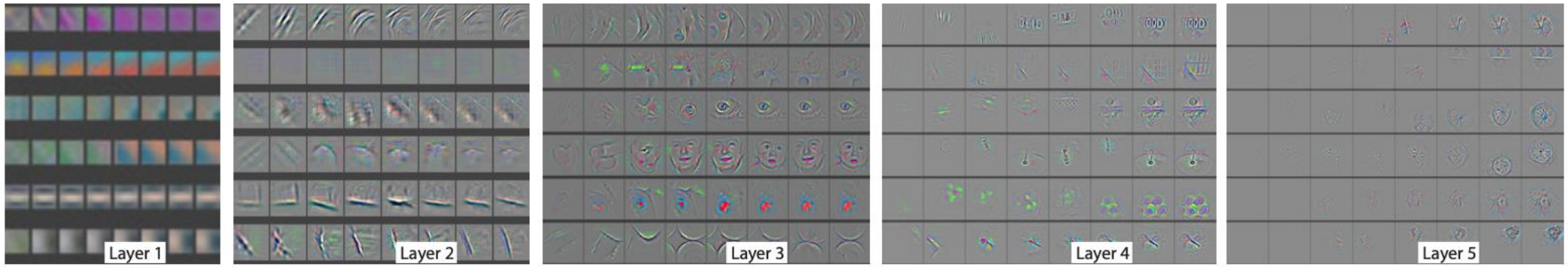
Week 9: Artificial Intelligent – CNN – Feature Detector “ImageNet”



Visualizing and Understanding Convolutional Networks MD Zeiler · 2013

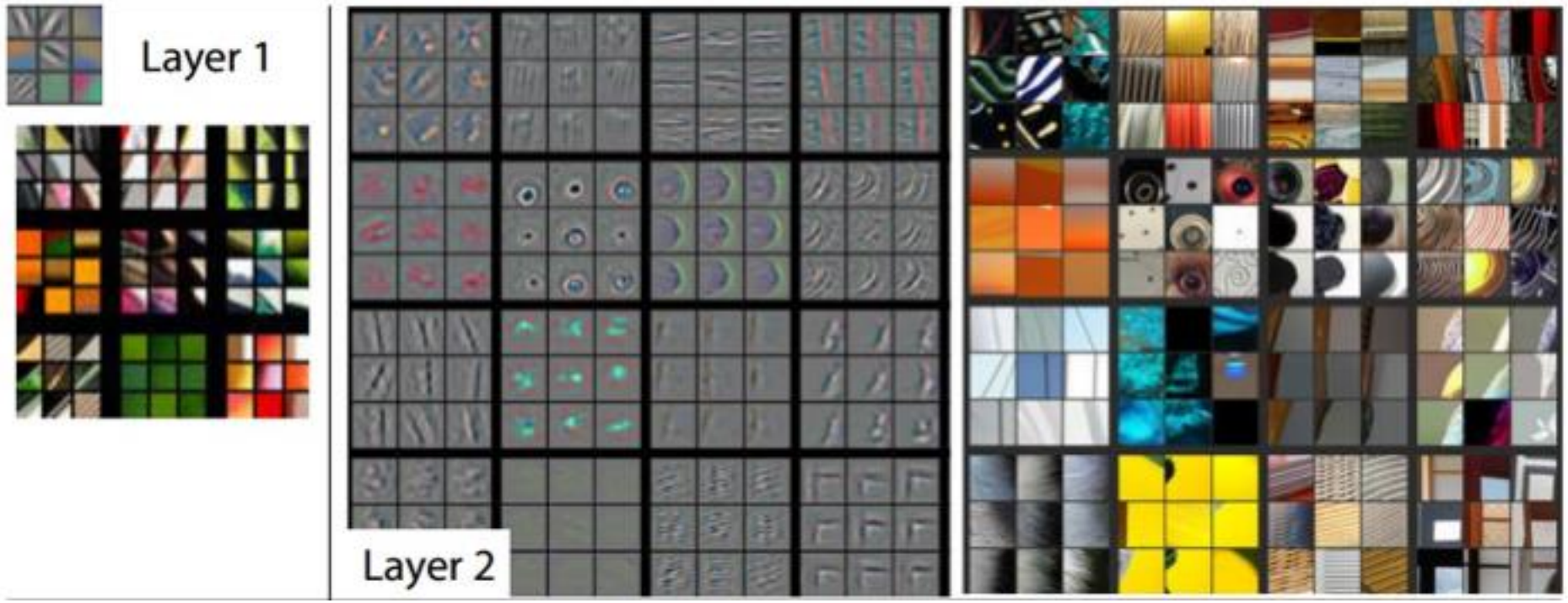
Week 9: Artificial Intelligent – CNN – Feature Detector “ImageNet”





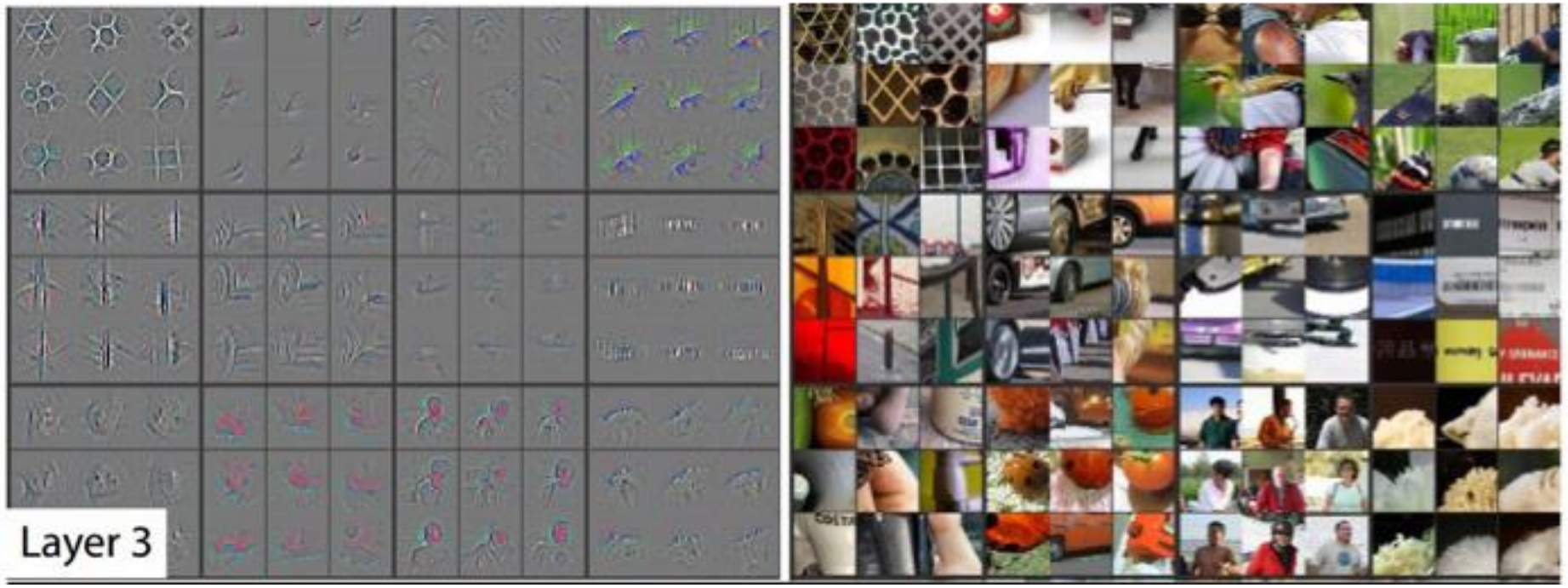
Visualizing and Understanding Convolutional Networks MD Zeiler · 2013

Week 9: Artificial Intelligent – CNN – Feature Detector “ImageNet”



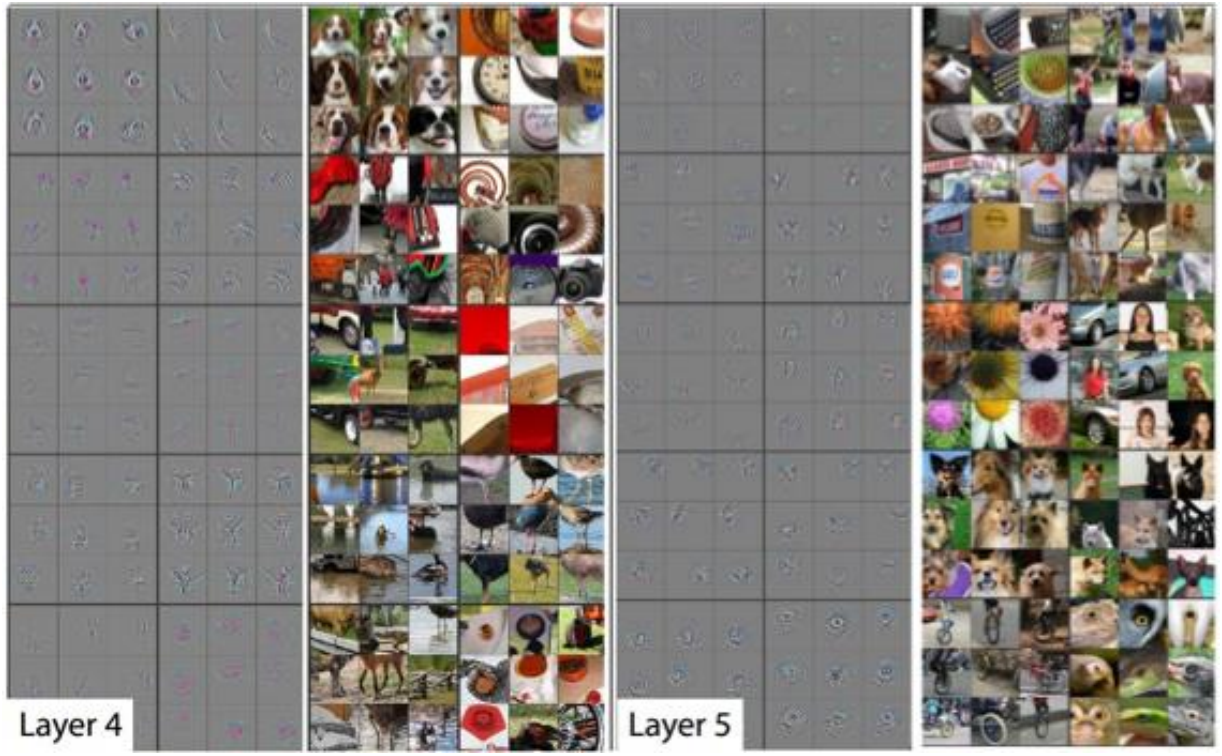
Visualizing and Understanding Convolutional Networks MD Zeiler · 2013

Week 9: Artificial Intelligent – CNN – Feature Detector “ImageNet”



Visualizing and Understanding Convolutional Networks MD Zeiler · 2013

Week 9: Artificial Intelligent – CNN – Feature Detector “ImageNet”



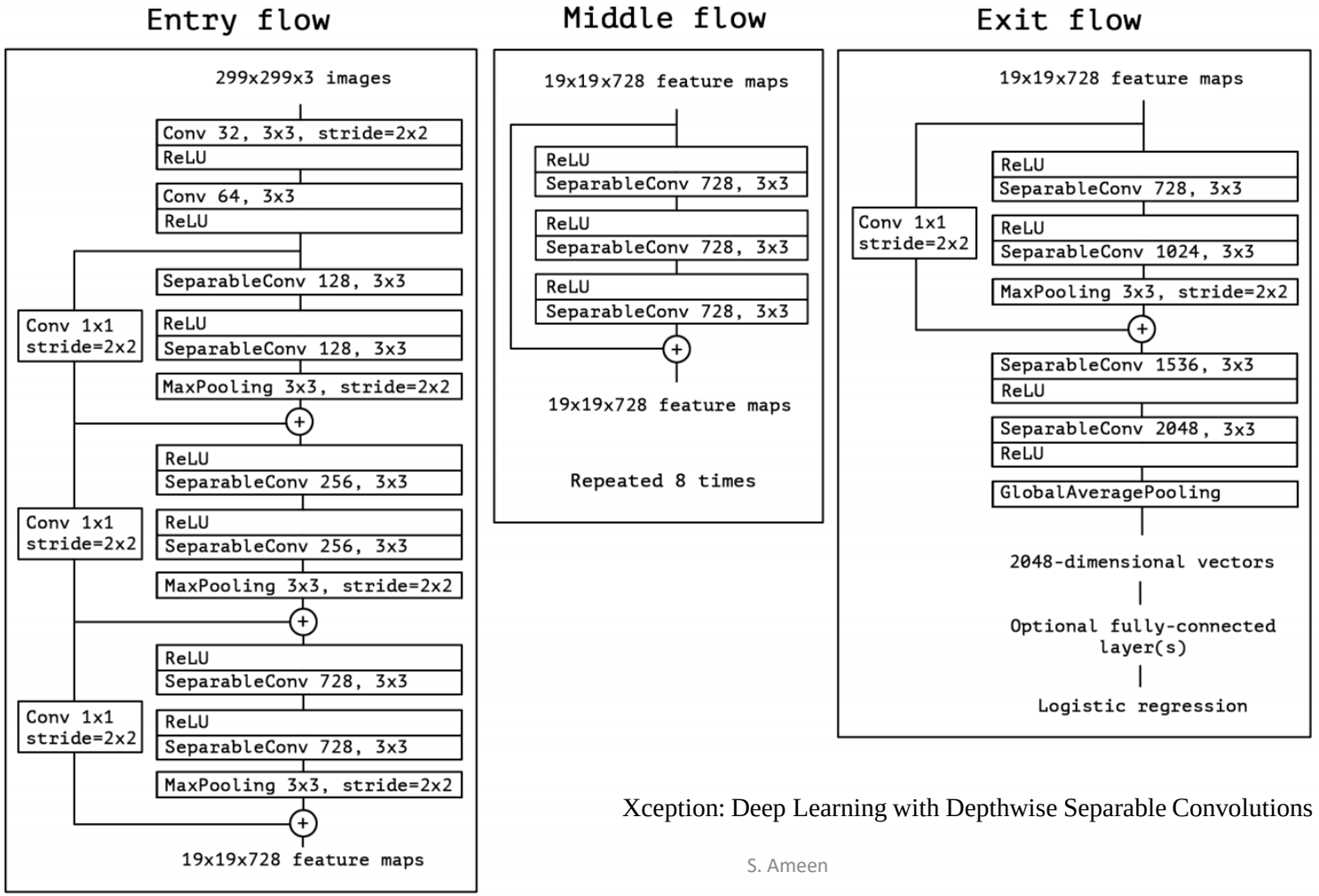
Visualizing and Understanding Convolutional Networks MD Zeiler · 2013

Week 9: Artificial Intelligent – CNN – Transfer Learning

University of Salford MANCHESTER

	very similar dataset	**very different dataset**
very little data	Use Linear Classifier on top layer	You're in trouble… Try linear classifier from different stages
quite a lot of data	Finetune a few layers	Finetune a larger number of layers

Week 9: Artificial Intelligent – CNN – Transfer Learning - Xception



Xception: Deep Learning with Depthwise Separable Convolutions Franc¸ois Chollet Google, Inc.

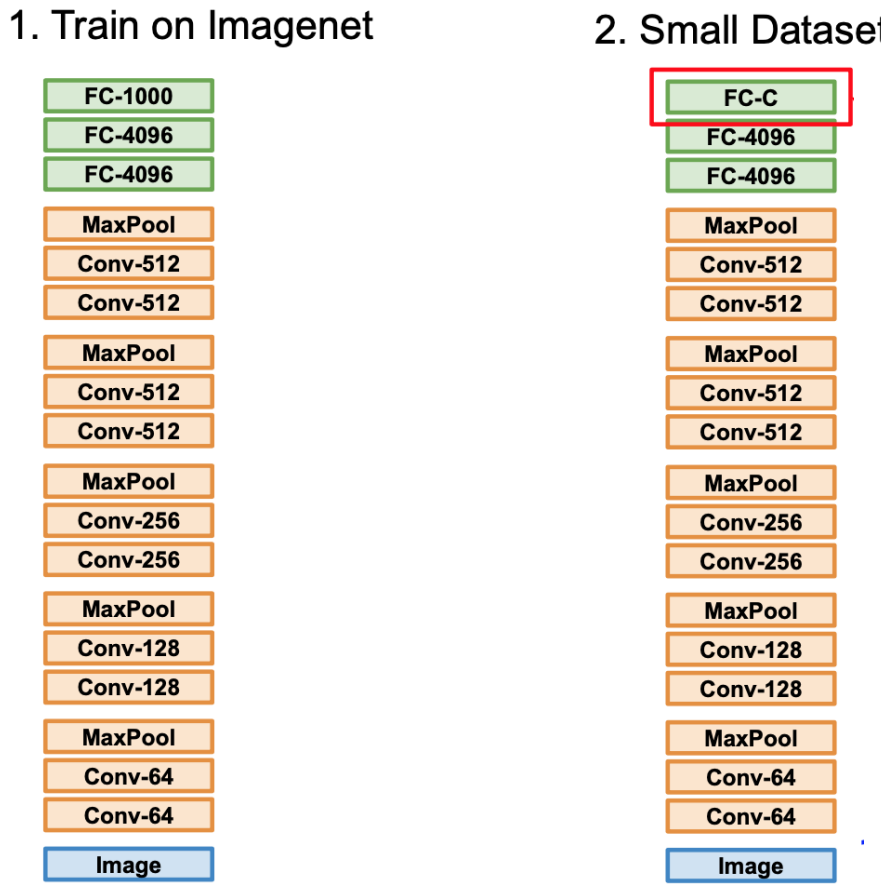
Week 9: Artificial Intelligent – CNN – Transfer Learning

University of Salford MANCHESTER

1. Train on Imagenet

AlexNet, VGG, ResNet, SqueezeNet, DenseNet, Inception v3, GoogLeNet, ShuffleNet v2, MobileNet v2, ResNeXt, Wide ResNet and MNASNet

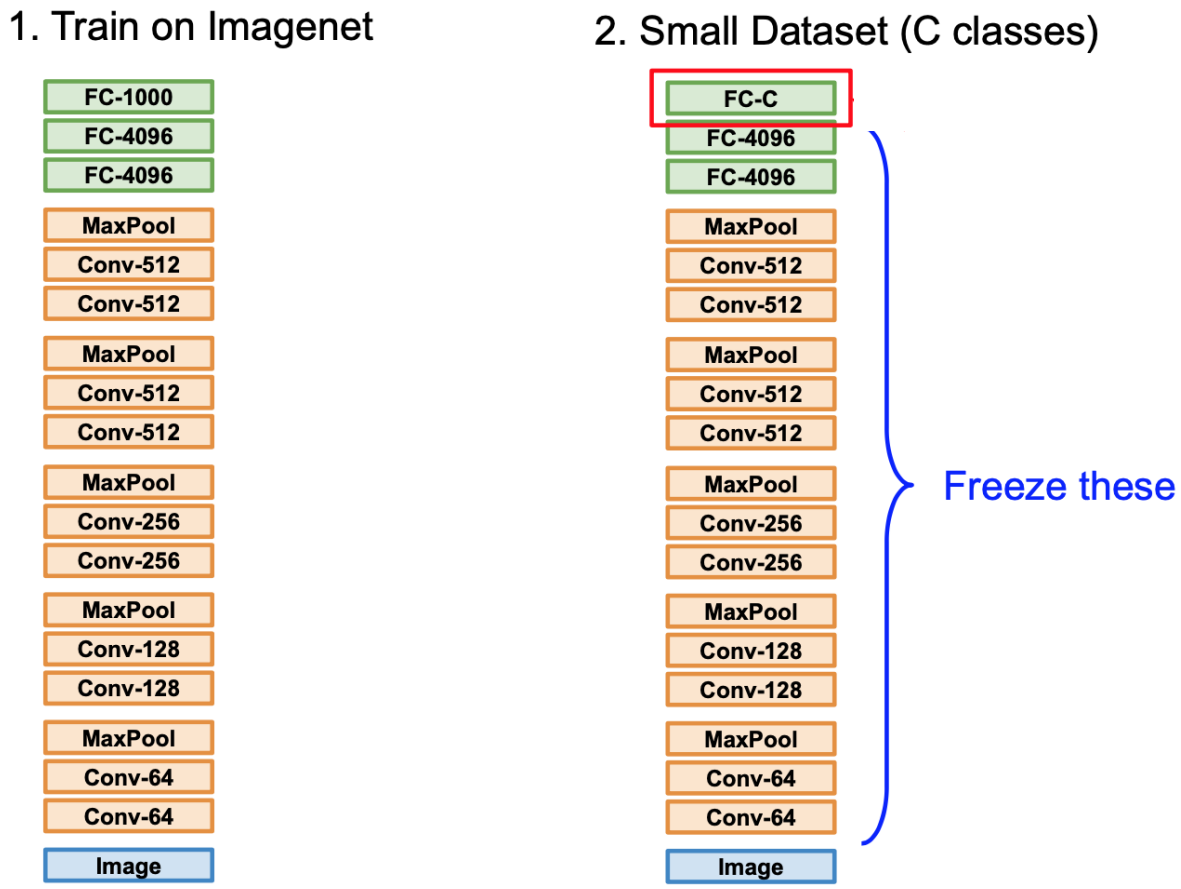
Week 9: Artificial Intelligent – CNN – Transfer Learning



VGG, AlexNet, …

```
base_model = keras.applications.Xception(
    weights='imagenet',  # Load weights pre-trained on ImageNet.
    input_shape=(150, 150, 3),
    include_top=False)  # Do not include the ImageNet classifier at the top.
```

Week 9: Artificial Intelligent – CNN – Transfer Learning

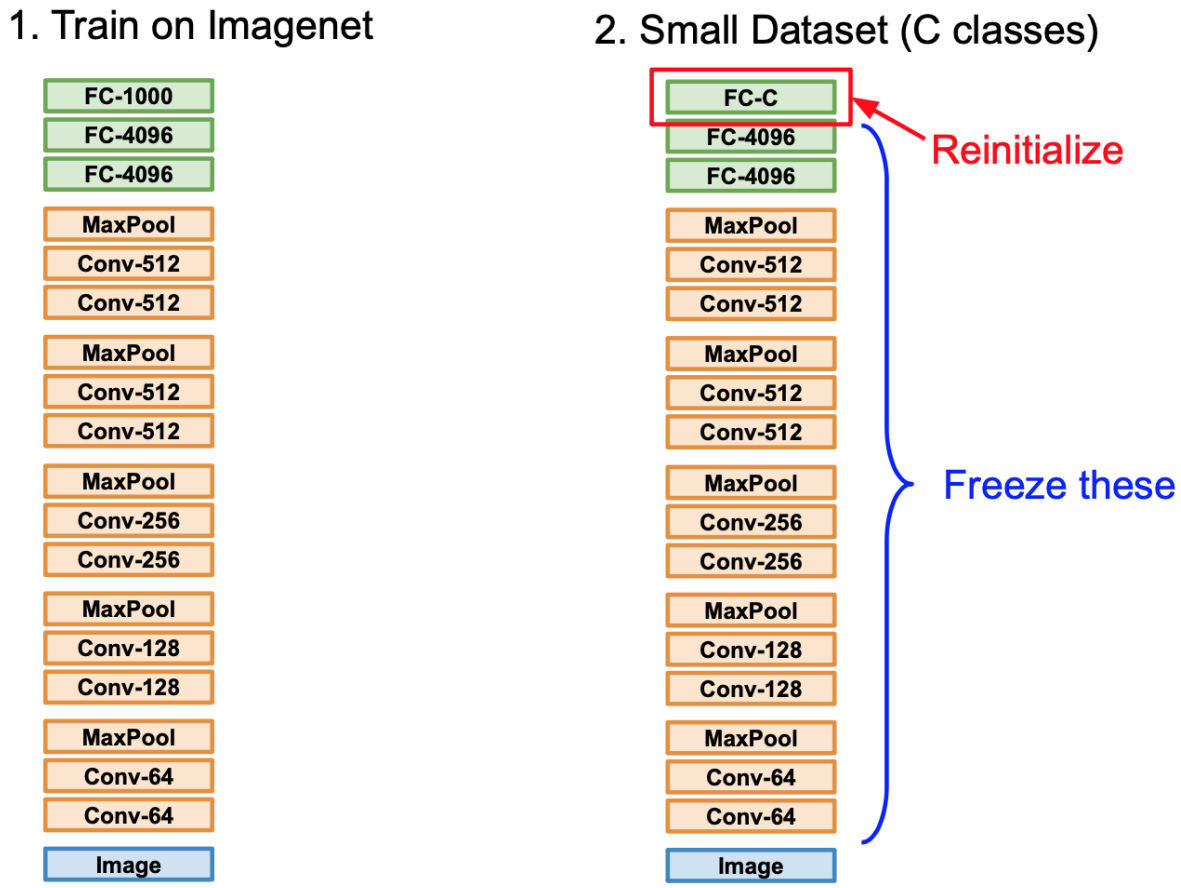



```
base_model = keras.applications.Xception(
    weights='imagenet',
    input_shape=(150, 150, 3),
    include_top=False)
```

```
base_model.trainable = False
```

Session Code:
2675
9432

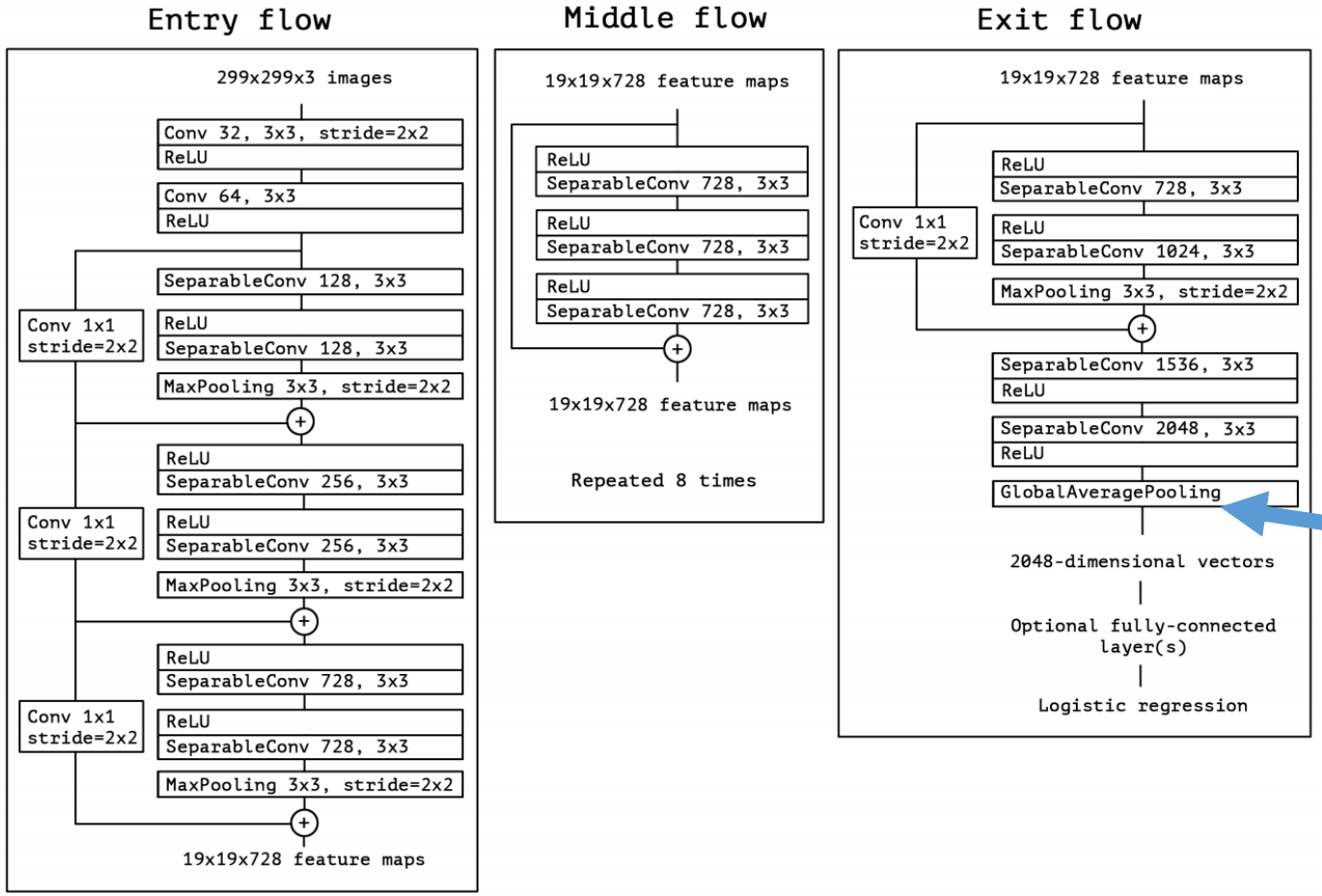
Week 9: Artificial Intelligent – CNN – Transfer Learning




```
base_model = keras.applications.Xception(
    weights='imagenet',
    input_shape=(150, 150, 3),
    include_top=False)

base_model.trainable = False
```

```
inputs = keras.Input(shape=(150, 150, 3))
x = base_model(inputs, training=False)
x = keras.layers.GlobalAveragePooling2D()(x)
outputs = keras.layers.Dense(1)(x)
model = keras.Model(inputs, outputs)
```

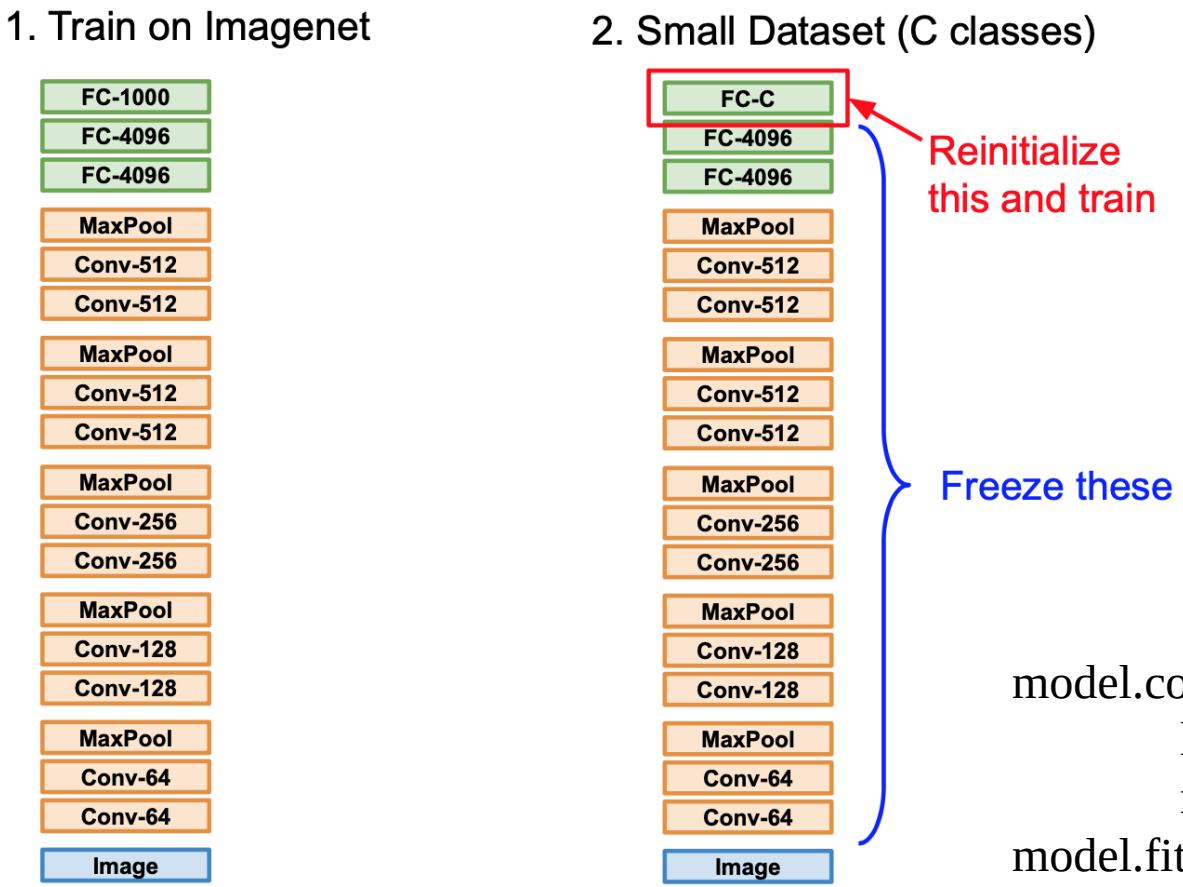
Week 9: Artificial Intelligent – CNN – Transfer Learning




```
base_model = keras.applications.Xception(
    weights='imagenet',
    input_shape=(150, 150, 3),
    include_top=False)

base_model.trainable = False
```

```
inputs = keras.Input(shape=(150, 150, 3))
x = base_model(inputs, training=False)
x = keras.layers.GlobalAveragePooling2D()(x)
outputs = keras.layers.Dense(1)(x)
model = keras.Model(inputs, outputs)
```

Week 9: Artificial Intelligent – CNN – Transfer Learning




```
base_model = keras.applications.Xception(
    weights='imagenet',
    input_shape=(150, 150, 3),
    include_top=False)
base_model.trainable = False

inputs = keras.Input(shape=(150, 150, 3))
x = base_model(inputs, training=False)
x = keras.layers.GlobalAveragePooling2D()(x)
outputs = keras.layers.Dense(1)(x)
model = keras.Model(inputs, outputs)
```

```
model.compile(optimizer=keras.optimizers.Adam(),
              loss=keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=[keras.metrics.BinaryAccuracy()])
model.fit(new_dataset, epochs=20)
```

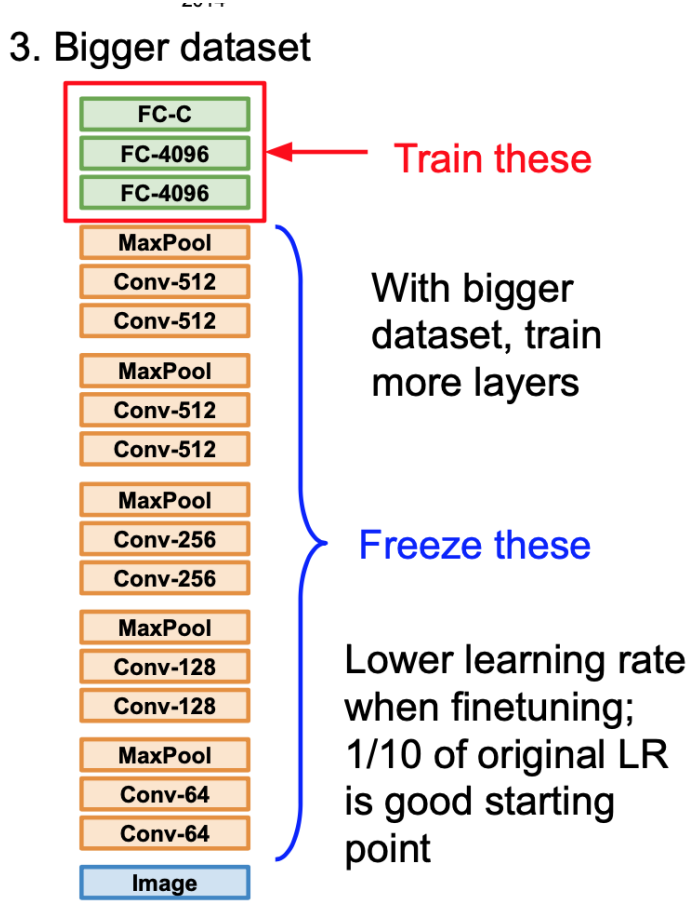
Week 9: Artificial Intelligent – CNN – Transfer Learning

University of Salford MANCHESTER

	very similar dataset	**very different dataset**
very little data	Use Linear Classifier on top layer	You're in trouble… Try linear classifier from different stages
quite a lot of data	Finetune a few layers	Finetune a larger number of layers

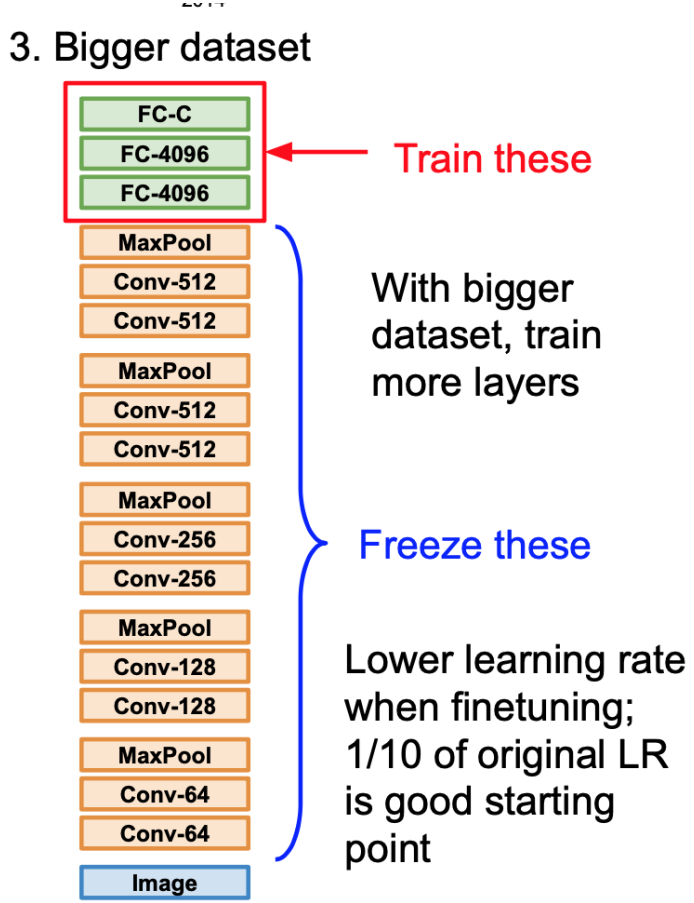
Week 9: Artificial Intelligent – CNN – Transfer Learning

base_model.trainable = True



Week 9: Artificial Intelligent – CNN – Transfer Learning

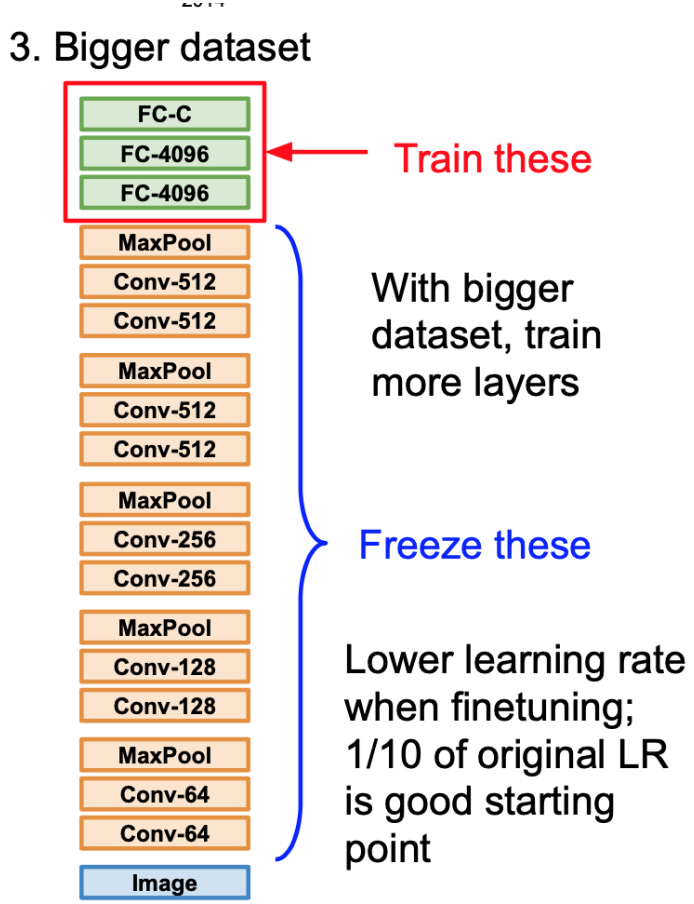
```
base_model.trainable = True

# Let's take a look to see how many layers are in the base model
print("Number of layers in the base model: ",
len(base_model.layers))

# Fine-tune from this layer onwards
fine_tune_at = 100

# Freeze all the layers before the `fine_tune_at` layer
for layer in base_model.layers[:fine_tune_at]:
  layer.trainable =  False
```



Week 9: Artificial Intelligent – CNN – Transfer Learning

University of Salford MANCHESTER

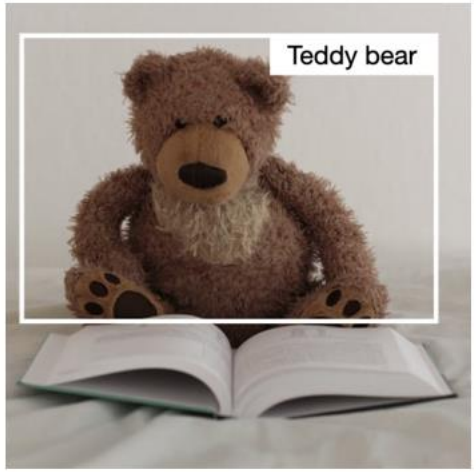
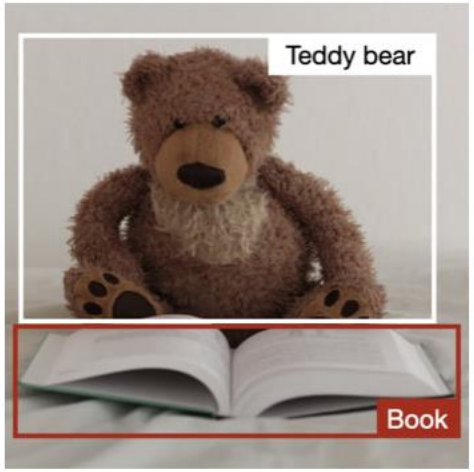
```
base_model.trainable = True


len(base_model.layers))
fine_tune_at = 100
for layer in base_model.layers[:fine_tune_at]:
  layer.trainable =  False

model.compile(loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),
          optimizer = tf.keras.optimizers.RMSprop(lr=base_learning_rate/10),
          metrics=['accuracy'])

history_fine = model.fit(train_dataset,
                         epochs=total_epochs,
                         initial_epoch=history.epoch[-1],
                         validation_data=validation_dataset)
```



Week 9: CNN – Object detection

University of Salford MANCHESTER

❑ **Types of models** — There are 3 main types of object recognition algorithms, for which the nature of what is predicted is different. They are described in the table below:

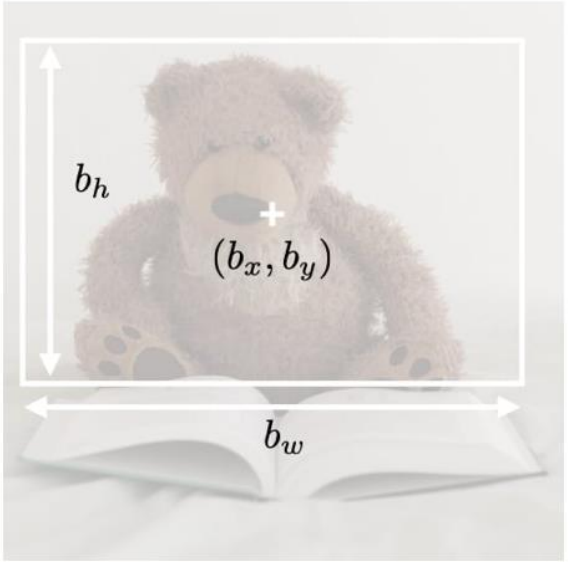
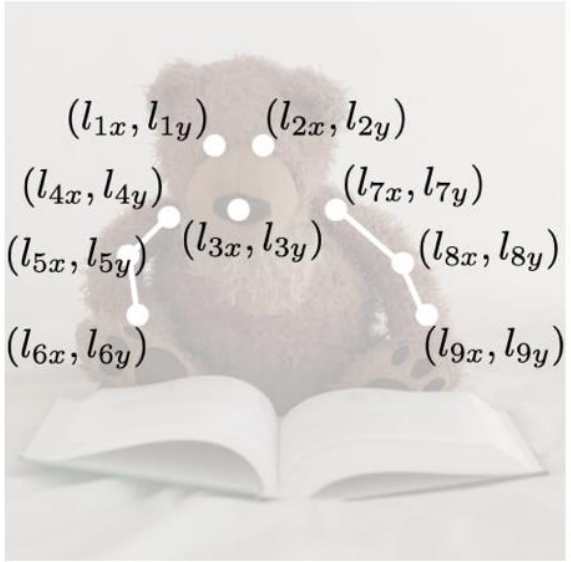
Image classification	Classification w. localization	Detection
		
• Classifies a picture • Predicts probability of object	• Detects an object in a picture • Predicts probability of object and where it is located	• Detects up to several objects in a picture • Predicts probabilities of objects and where they are located
Traditional CNN	Simplified YOLO, R-CNN	YOLO, R-CNN

S. Ameen

Week 9: CNN – Object detection

University of Salford MANCHESTER

□ **Detection** — In the context of object detection, different methods are used depending on whether we just want to locate the object or detect a more complex shape in the image. The two main ones are summed up in the table below:

Bounding box detection	Landmark detection
• Detects the part of the image where the object is located	• Detects a shape or characteristics of an object (e.g. eyes) • More granular
	
Box of center (b_x, b_y), height b_h and width b_w	Reference points (l_{1x}, l_{1y}), ..., (l_{nx}, l_{ny})

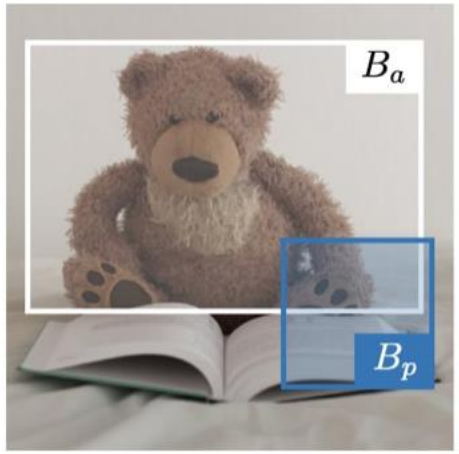
Week 9: CNN – Object detection

University of Salford MANCHESTER

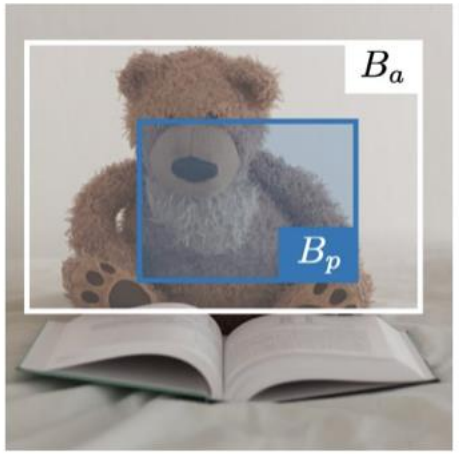
□ **Intersection over Union** — Intersection over Union, also known as IoU, is a function that quantifies how correctly positioned a predicted bounding box B_p is over the actual bounding box B_a. It is defined as:

$$\mathrm{IoU}(B_p, B_a) = \frac{B_p \cap B_a}{B_p \cup B_a}$$

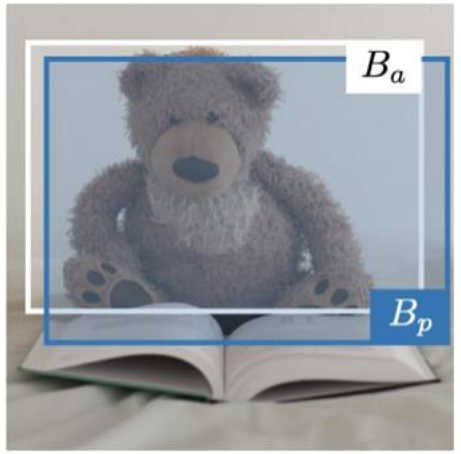
Remark: we always have $\mathrm{IoU} \in [0, 1]$. *By convention, a predicted bounding box* B_p *is considered as being reasonably good if* $\mathrm{IoU}(B_p, B_a) \geqslant 0.5$.



$\mathrm{IoU}(B_p, B_a) = 0.1$



$\mathrm{IoU}(B_p, B_a) = 0.5$

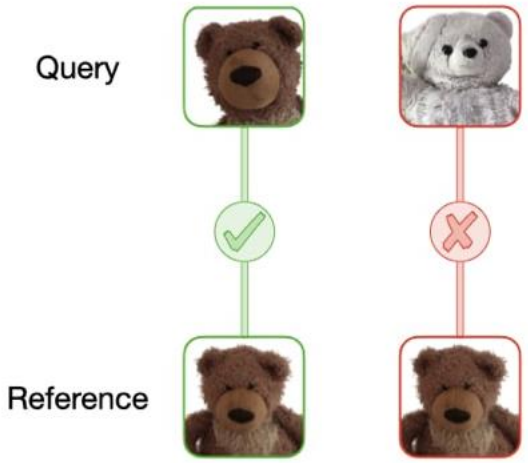
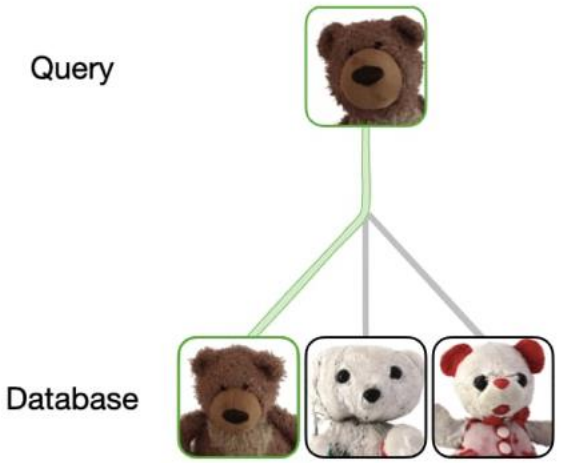


$\mathrm{IoU}(B_p, B_a) = 0.9$

Week 9: CNN – Face verification and recognition

University of Salford MANCHESTER

□ **Types of models** — Two main types of model are summed up in table below:

Face verification	Face recognition
• Is this the correct person? • One-to-one lookup	• Is this one of the K persons in the database? • One-to-many lookup
	

Week 9: CNN – Neural style transfer

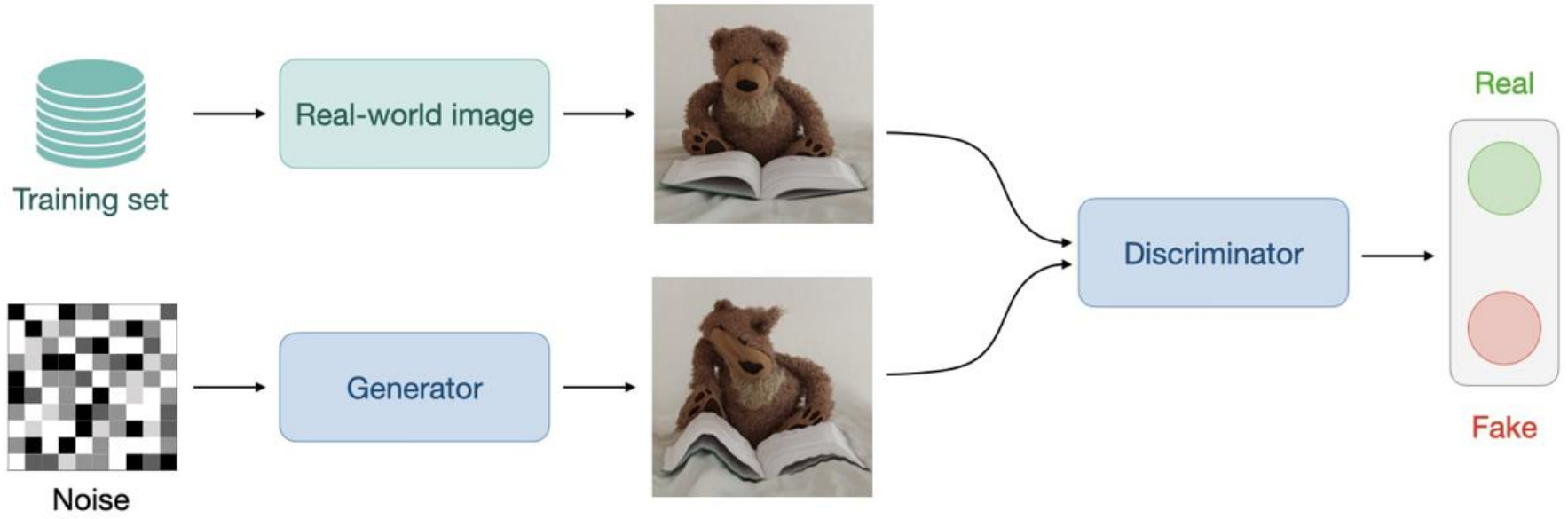
□ **Motivation** — The goal of neural style transfer is to generate an image G based on a given content C and a given style S.

Content C Style S Generated image G

Week 9: CNN – GAN

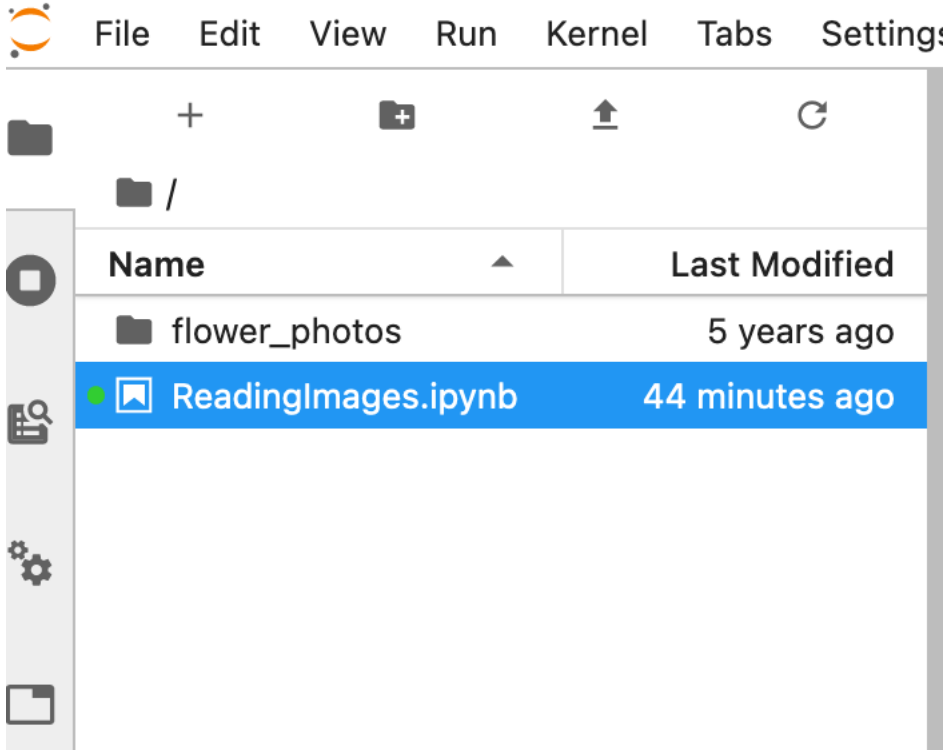
University of Salford MANCHESTER

❑ **Generative Adversarial Network** — Generative adversarial networks, also known as GANs, are composed of a generative and a discriminative model, where the generative model aims at generating the most truthful output that will be fed into the discriminative which aims at differentiating the generated and true image.

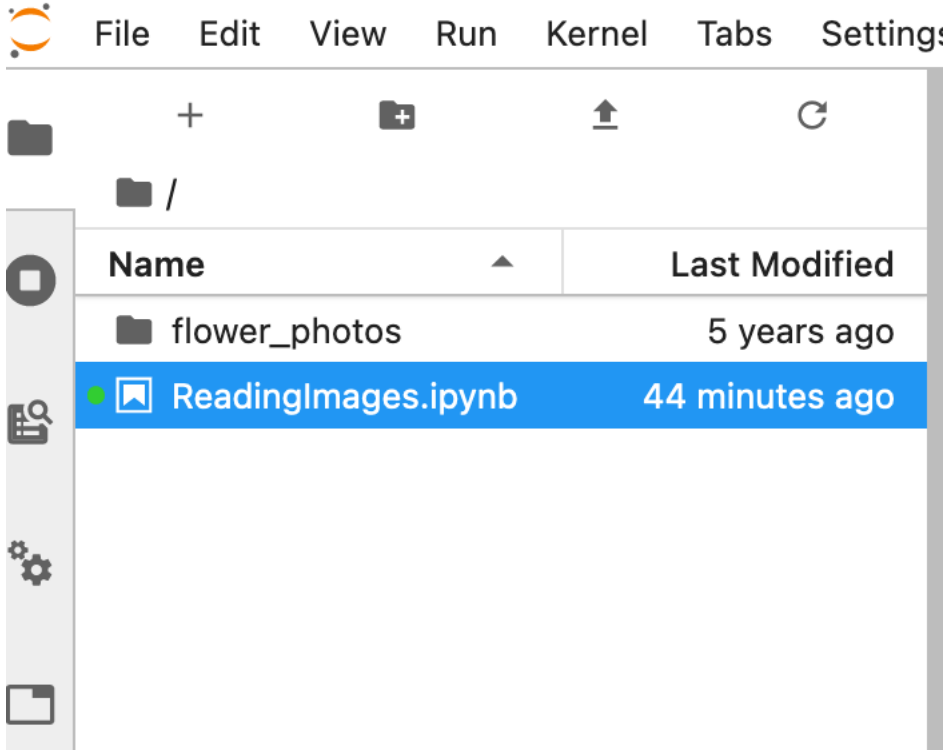


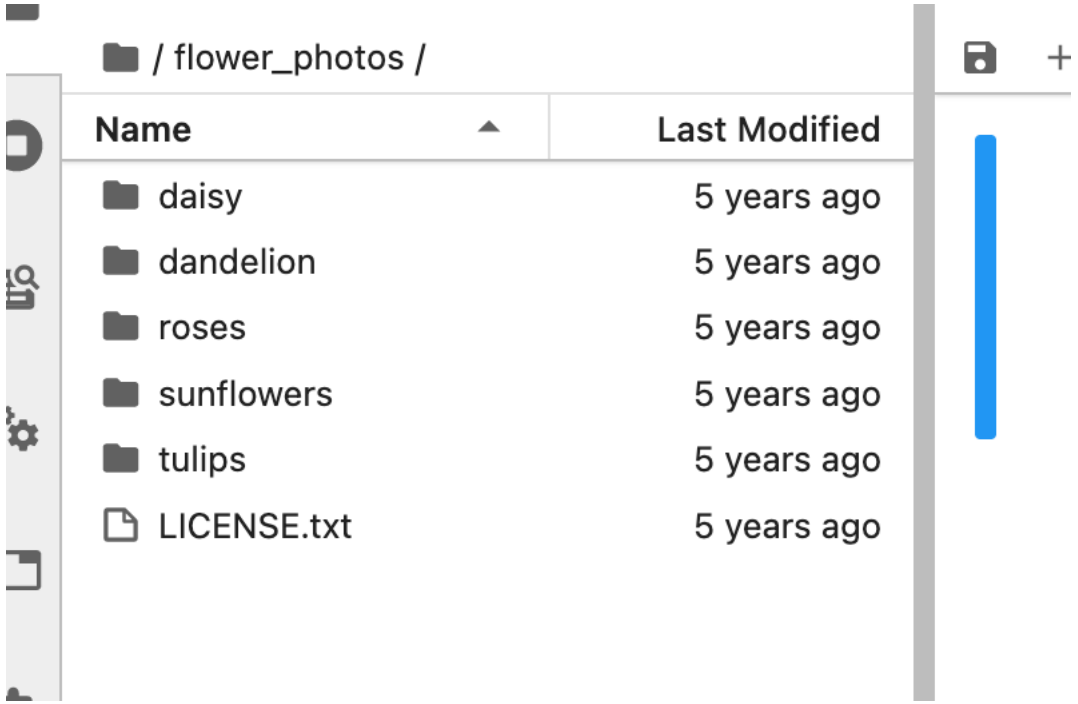
Remark: use cases using variants of GANs include text to image, music generation and synthesis.

Week 9 – Reading the Images



Week 9 – Reading the Images





Week 9 – Reading the Images

University of Salford MANCHESTER

[1]:
```
import numpy as np
import os
import PIL
import PIL.Image
import tensorflow as tf
import tensorflow_datasets as tfds
```

[2]:
```
data_dir = "flower_photos"
```

[4]:
```
batch_size = 32
img_height = 180
img_width = 180
```

Week 9 – Reading the Images Loading

University of Salford MANCHESTER

```
train_ds = tf.keras.preprocessing.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="training",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

```
Found 3670 files belonging to 5 classes.
Using 2936 files for training.
```

```
val_ds = tf.keras.preprocessing.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="validation",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

Week 9 – Reading the Images Loading

University of Salford MANCHESTER

```
from tensorflow.keras import layers

normalization_layer = tf.keras.layers.experimental.preprocessing.Rescaling(1./255)
```

Week 9 – Reading the Images Loading

University of Salford MANCHESTER

```
AUTOTUNE = tf.data.AUTOTUNE

train_ds = train_ds.cache().prefetch(buffer_size=AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)
```

Week 9 – Reading the Images Loading / Modelling

University of Salford MANCHESTER

```
num_classes = 5

model = tf.keras.Sequential([
  layers.experimental.preprocessing.Rescaling(1./255),
  layers.Conv2D(32, 3, activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, activation='relu'),
  layers.MaxPooling2D(),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])
```

Week 9 – Reading the Images Loading / Modelling

University of Salford MANCHESTER

```
model.compile(
  optimizer='adam',
  loss=tf.losses.SparseCategoricalCrossentropy(from_logits=True),
  metrics=['accuracy'])
```

Week 9 – Reading the Images Loading / Modelling / Training

University of Salford MANCHESTER

```
model.fit(
  train_ds,
  validation_data=val_ds,
  epochs=1
)
```

```
92/92 [==============================] - 47s 501ms/step - loss: 1.5055 - accuracy: 0.3228 - val_loss: 1.1538 - val_accurac
y: 0.5368
```

Lecture - Week 9 : AI

University of Salford MANCHESTER

- Next - RNN

Week 9 – : Deep Learning
Modelling Sequences

University of Salford MANCHESTER

Sentences:

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}
Paris	*Talks*	*Set*	*Stage*	*for*	*Action*	*as*	*Risks*	*to*	*the*	*Climate*	*Rise*

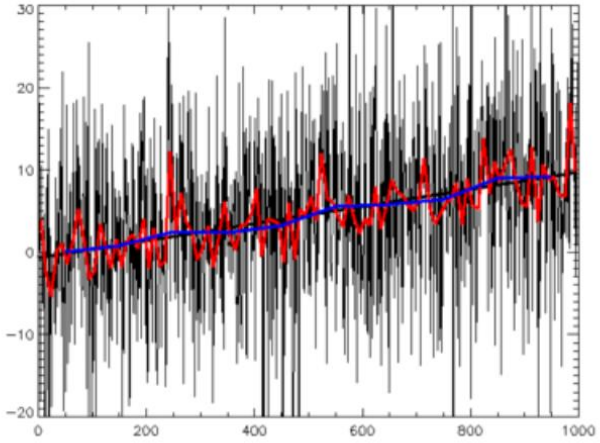
Week 9 – : Deep Learning
Modelling Sequences

University of Salford MANCHESTER

Sentences:

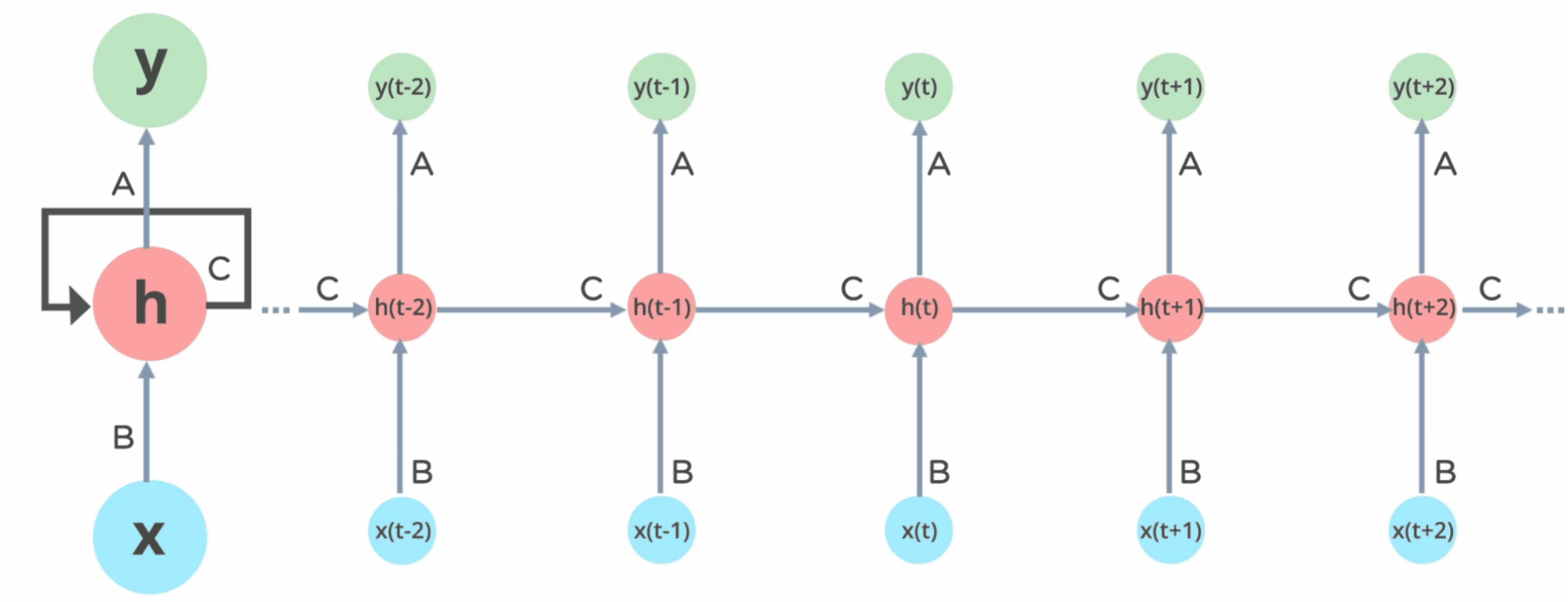
x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}
Paris	*Talks*	*Set*	*Stage*	*for*	*Action*	*as*	*Risks*	*to*	*the*	*Climate*	*Rise*

Time series:

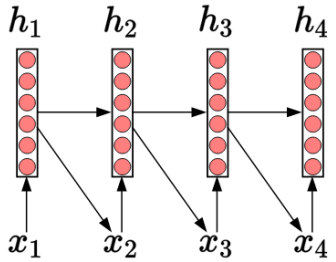


Week 9 – : Deep Learning
Modelling Sequences

Week 9 – : Deep Learning
Recurrent neural networks (RNN)



Week 9 – : Deep Learning
RNN – Prediction



$$h_1 = \text{Encode}(x_1)$$
$$x_2 \sim \text{Decode}(h_1)$$
$$h_2 = \text{Encode}(h_1, x_2)$$
$$x_3 \sim \text{Decode}(h_2)$$
$$h_3 = \text{Encode}(h_2, x_3)$$
$$x_4 \sim \text{Decode}(h_3)$$
$$h_4 = \text{Encode}(h_3, x_4)$$

Update context vector:

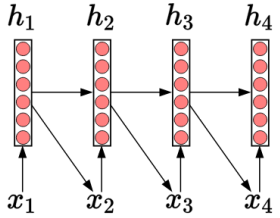
$$h_t = \text{Encode}(h_{t-1}, x_t)$$

Predict next character:

$$x_{t+1} = \text{Decode}(h_t)$$

context h_t compresses $x_1, \ldots x_t$

Week 9 – : Deep Learning
RNN – Prediction



$$h_1 = \text{Encode}(x_1)$$
$$x_2 \sim \text{Decode}(h_1)$$
$$h_2 = \text{Encode}(h_1, x_2)$$
$$x_3 \sim \text{Decode}(h_2)$$
$$h_3 = \text{Encode}(h_2, x_3)$$
$$x_4 \sim \text{Decode}(h_3)$$
$$h_4 = \text{Encode}(h_3, x_4)$$

Update context vector:

$$h_t = \text{Encode}(h_{t-1}, x_t)$$

Predict next character:

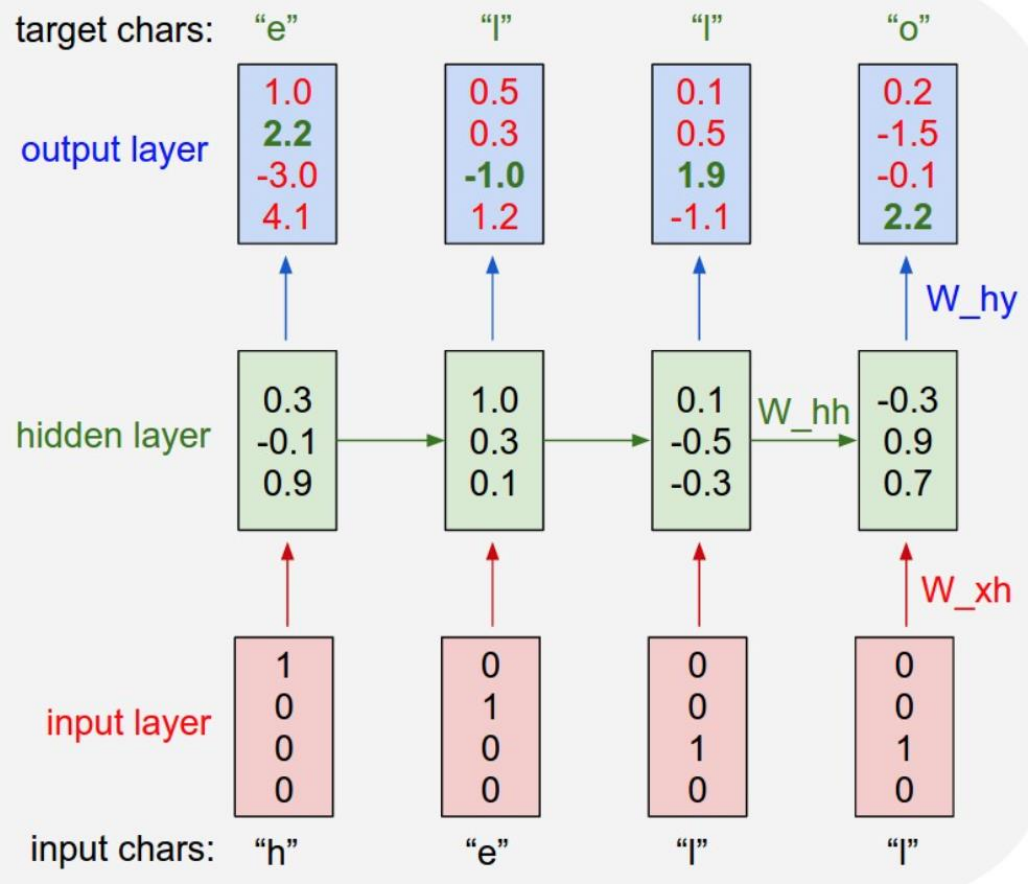
$$x_{t+1} = \text{Decode}(h_t)$$

context h_t compresses $x_1, \ldots x_t$

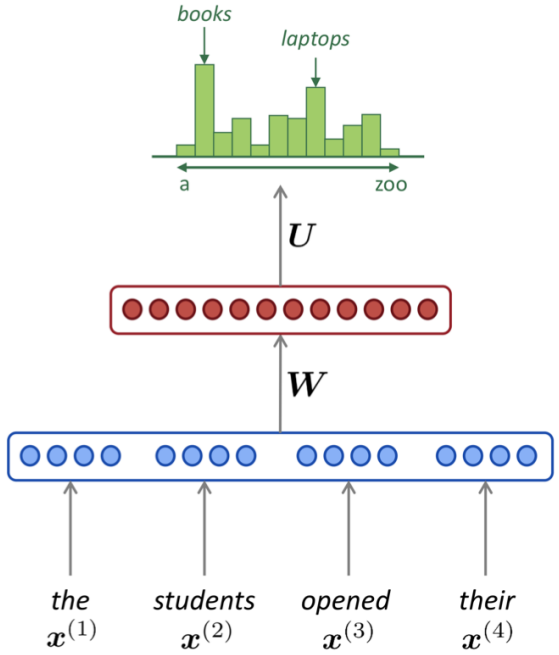
$$\text{Encode}(h_{t-1}, x_t) = \sigma(V\, h_{t-1} + W\, x_t) = h_t$$

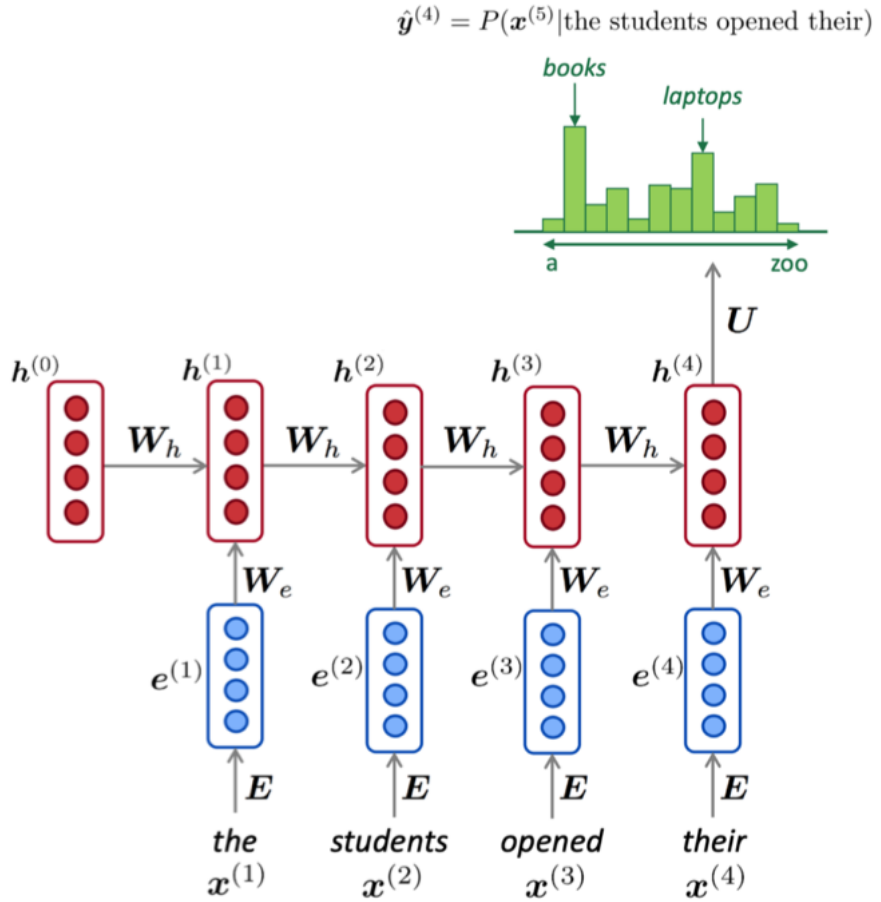
$$\text{Decode}(h_t) \sim \text{softmax}(W'\, h_t) = p(x_{t+1})$$

Week 9 – : Deep Learning
RNN – Prediction – Assignment



Week 9 – : Deep Learning
Example NN VS RNN





Week 9 – : Deep Learning
RNN – Coding

```
simple_rnn = tf.keras.layers.SimpleRNN(4)
```

Week 9 – : Deep Learning
RNN – Coding

University of Salford MANCHESTER

```
simple_rnn = tf.keras.layers.SimpleRNN(
    4, return_sequences=True, return_state=True)
```

return_sequences: Boolean. Whether to return the last output in the output sequence, or the full sequence. Default: False.

return_state: Boolean. Whether to return the last state in addition to the output. Default: False

Week 9 – : Deep Learning
RNN – Coding

University of Salford MANCHESTER

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
model = keras.Sequential()
from tensorflow.keras.layers import SimpleRNN, LSTM, Dense
```

Week 9 – : Deep Learning
RNN – Coding


```
# define model
model = keras.Sequential()
model.add(SimpleRNN(100, input_shape=(5,1)))
model.add(Dense(1))
```

Week 9 – : Deep Learning
RNN – Coding

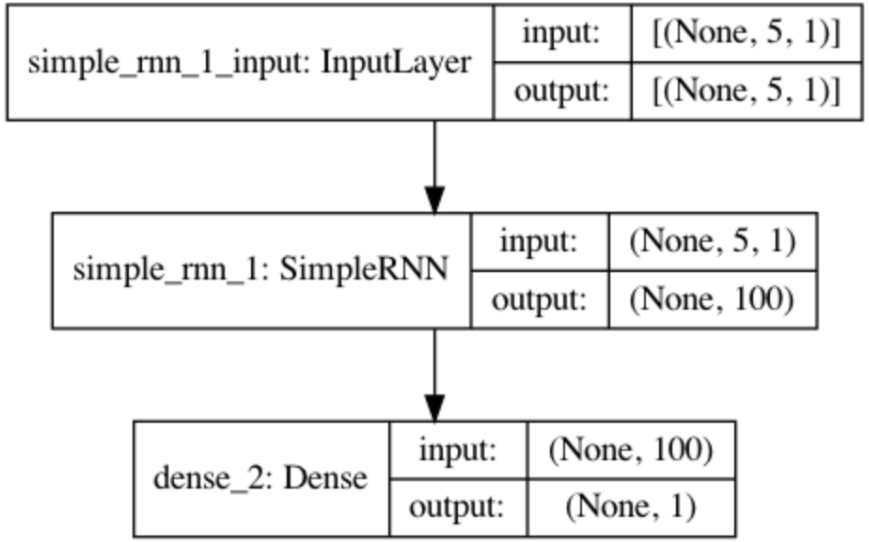
University of Salford MANCHESTER

```
# define model
model = keras.Sequential()
model.add(SimpleRNN(100, input_shape=(5,1)))
model.add(Dense(1))
```

```
model.summary()
```

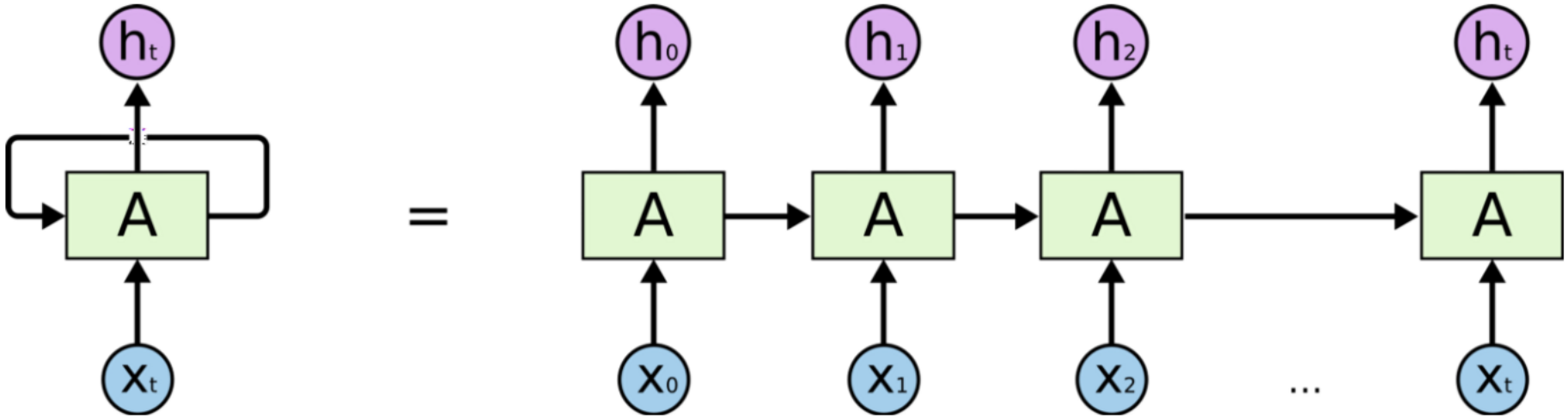
```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
simple_rnn_1 (SimpleRNN)     (None, 100)               10200
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 101
=================================================================
Total params: 10,301
Trainable params: 10,301
Non-trainable params: 0
_________________________________________________________________
```

```
from tensorflow.keras.utils import plot_model
plot_model(model, 'model.png', show_shapes=True)
```

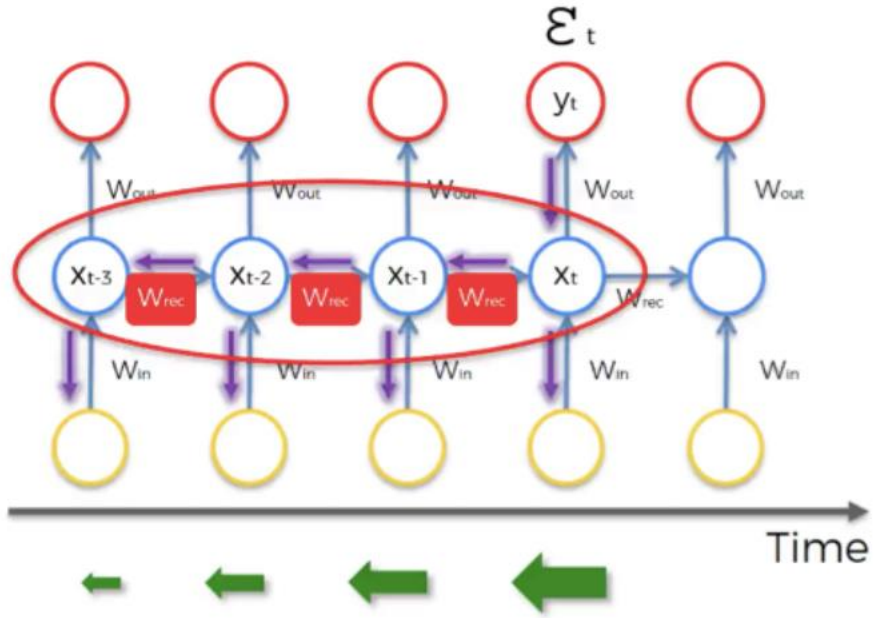



```
# compile the model
model.compile(----)
# fit the model
model.fit(----)
# evaluate the model
```

Week 9 – : Deep Learning
RNN – Problems - Vanishing gradients



Week 9 – : Deep Learning
RNN – Problems - Vanishing gradients



$$\frac{\partial \mathcal{E}}{\partial \theta} = \sum_{1 \le t \le T} \frac{\partial \mathcal{E}_t}{\partial \theta} \quad (3)$$

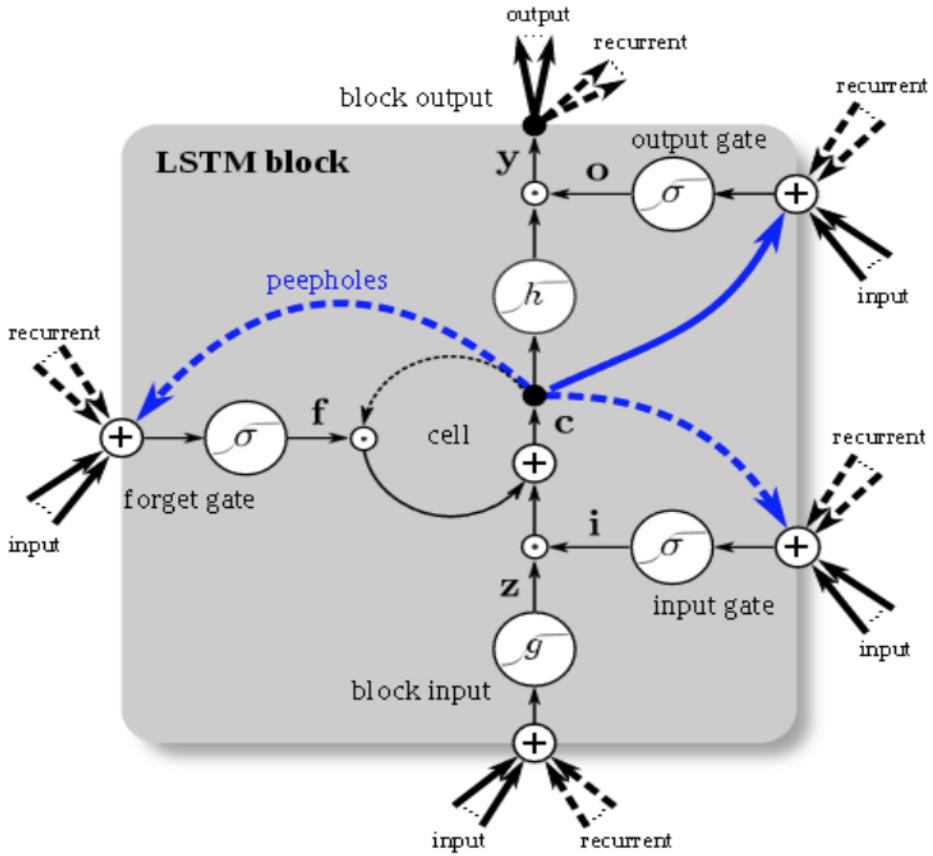
$$\frac{\partial \mathcal{E}_t}{\partial \theta} = \sum_{1 \le k \le t} \left(\frac{\partial \mathcal{E}_t}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} \frac{\partial^+ \mathbf{x}_k}{\partial \theta} \right) \quad (4)$$

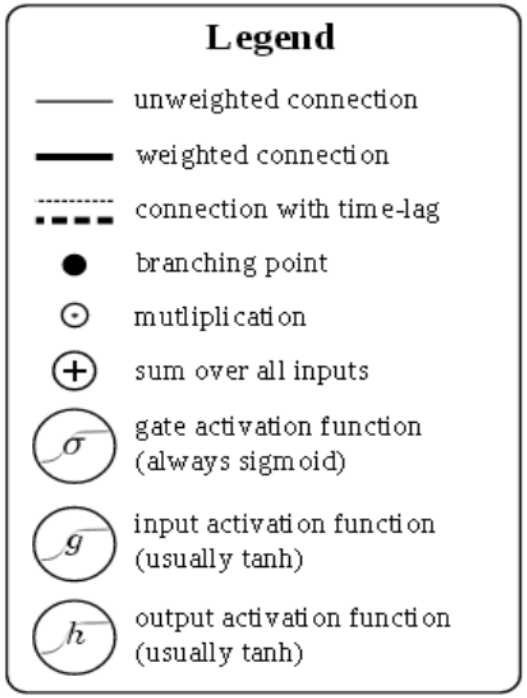
$$\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} = \prod_{t \ge i > k} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_{i-1}} = \prod_{t \ge i > k} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1})) \quad (5)$$

W_{rec} ~ small ⇨ Vanishing

W_{rec} ~ large ⇨ Exploding

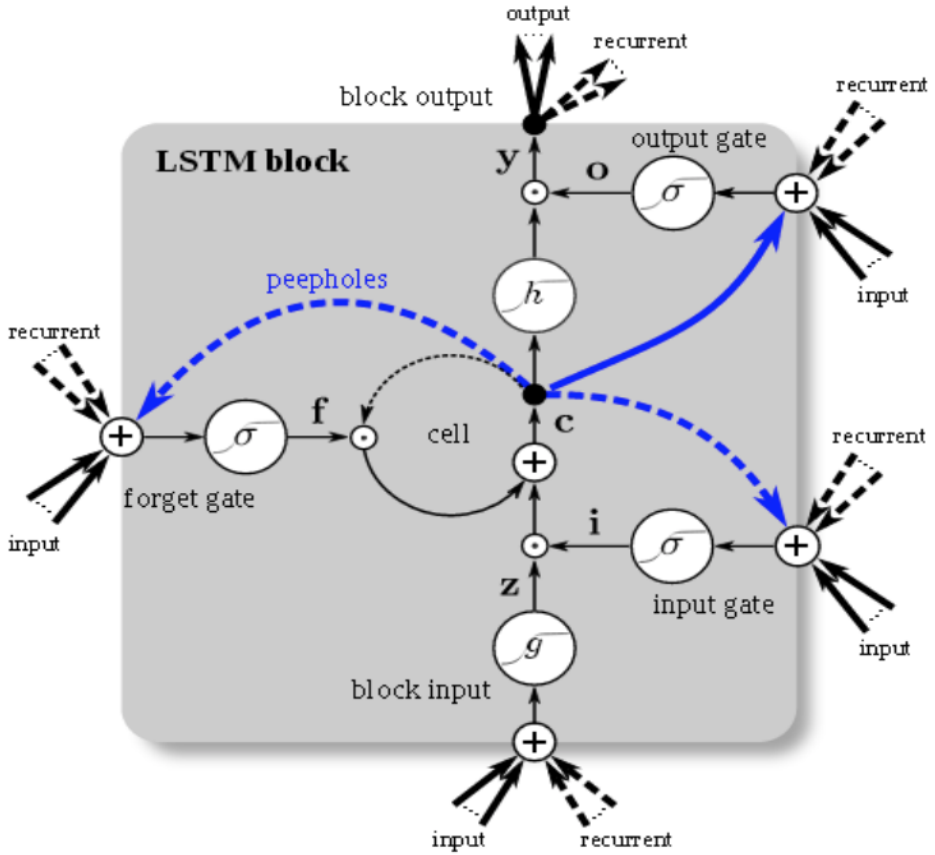
Week 9 – : Deep Learning
Long Short-Term Memory (LSTM)

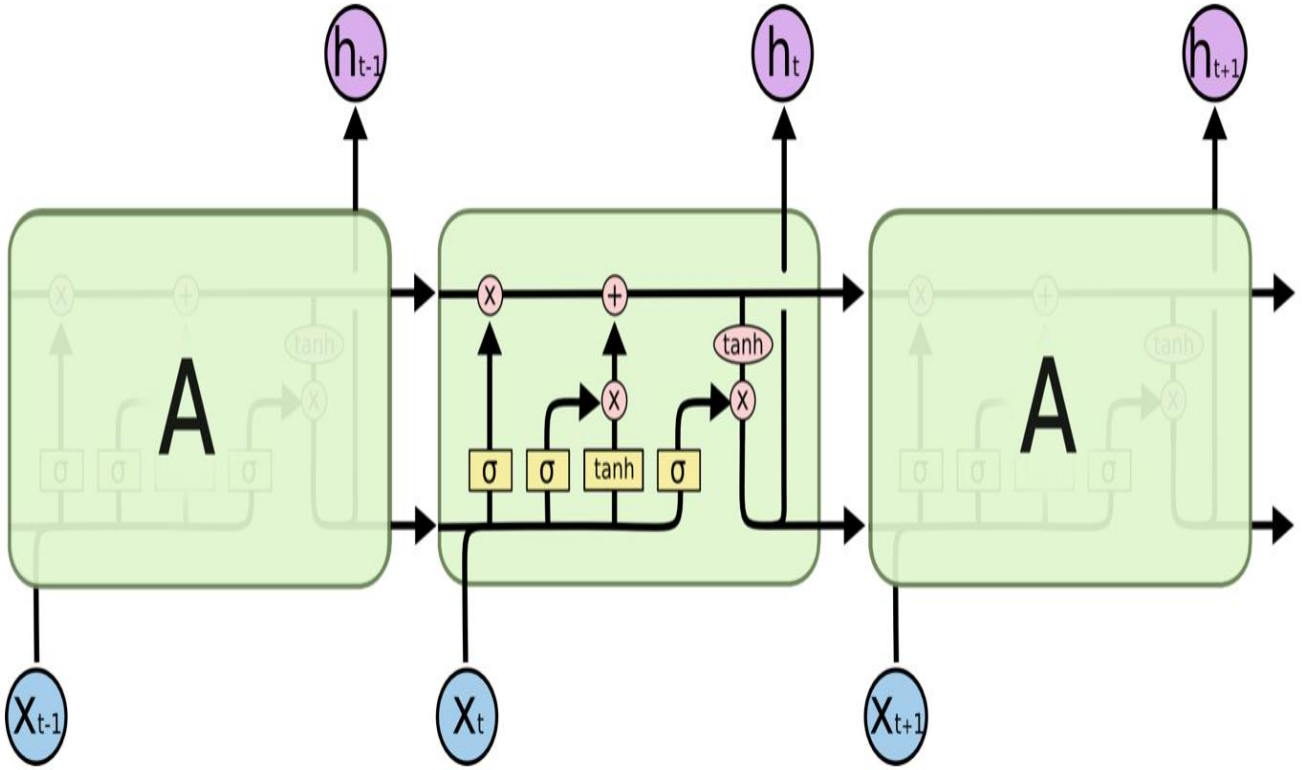




[Jürgen Schmidhuber & Sepp Hochreiter uber]

Week 9 – : Deep Learning
Long Short-Term Memory (LSTM)





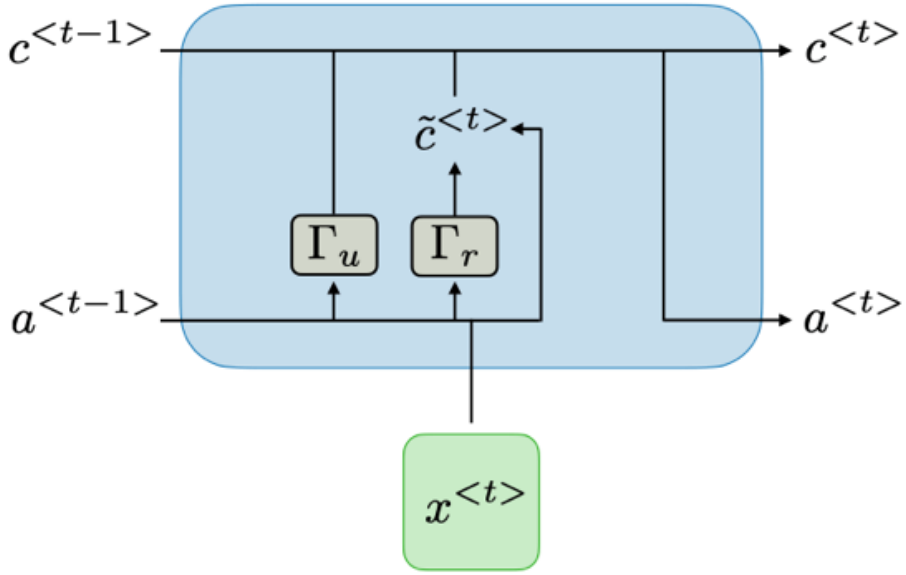
Week 9 – : Deep Learning
LSTM -Coding

University of Salford MANCHESTER

```
# define model
model = keras.Sequential()
model.add(SimpleRNN(100, input_shape=(5,1)))
model.add(Dense(1))
```

```
# define model
model = keras.Sequential()
model.add(LSTM(100, input_shape=(5,1)))
model.add(Dense(1))
```

Week 9 – : Deep Learning
Gated Recurrent Unit (GRU)

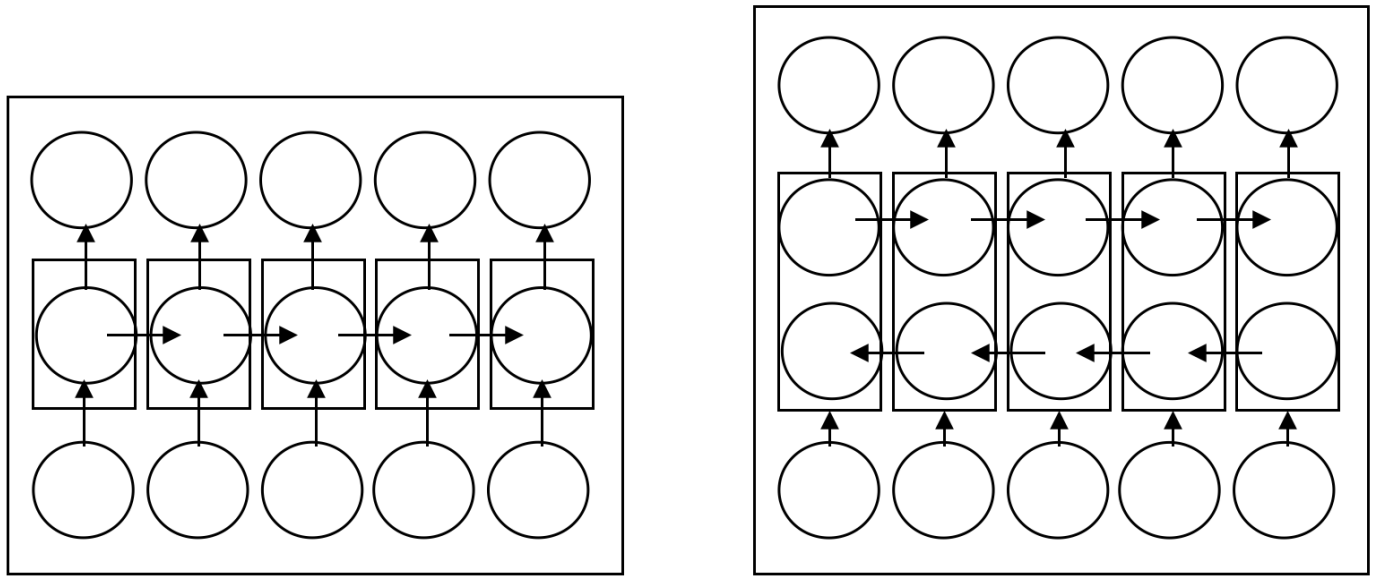


Week 9 – : Deep Learning
Gated Recurrent Unit (GRU)

University of Salford MANCHESTER

```
# define model
model = keras.Sequential()
model.add(GRU(100, input_shape=(5,1)))
model.add(Dense(1))
```

Week 9 – : Deep Learning Bidirectional



Structure overview

(a) unidirectional RNN
(b) bidirectional RNN

Week 9 – : Deep Learning
Bidirectional LSTM - Code

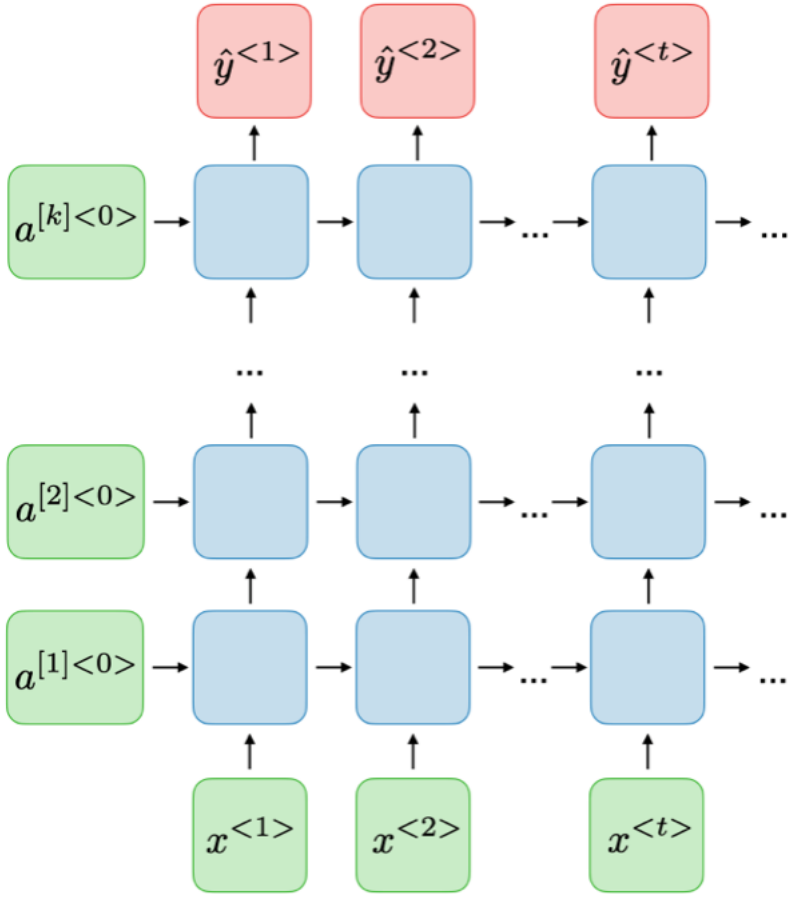
```
model = Sequential()

 model.add(Bidirectional(LSTM(10, return_sequences=True)
, backward_layer=LSTM(10, activation='relu', return_sequences=True, go_backwards=True)
, input_shape=(5, 10)))

 model.add(Dense(5))
```

Week 9 – : Deep Learning
Bidirectional LSTM - Code

University of Salford MANCHESTER

```
model = Sequential()
model.add(Bidirectional(LSTM(10, return_sequences=True), input_shape=(5, 10)))
model.add(Bidirectional(LSTM(10)))
model.add(Dense(5))
model.add(Activation('softmax'))
model.compile(loss='categorical_crossentropy', optimizer='rmsprop')

 # With custom backward layer
 model = Sequential()
 forward_layer = LSTM(10, return_sequences=True)
 backward_layer = LSTM(10, activation='relu', return_sequences=True,
                       go_backwards=True)
 model.add(Bidirectional(forward_layer, backward_layer=backward_layer,
                         input_shape=(5, 10)))
 model.add(Dense(5))
 model.add(Activation('softmax'))
 model.compile(loss='categorical_crossentropy', optimizer='rmsprop')
```

Week 9 – : Deep Learning
Deep RNN



Week 9 – : Deep Learning
Deep GRU (DGRU)

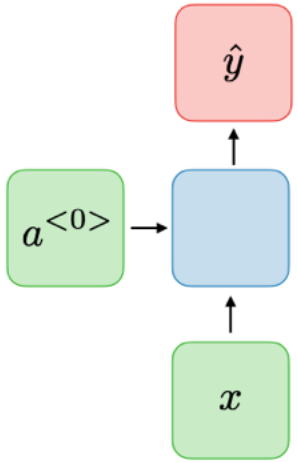
University of Salford MANCHESTER

```
# define model
model = keras.Sequential()
model.add(GRU(100, input_shape=(5,1), return_sequences = True))
model.add(GRU(100, return_sequences = True)),
model.add(GRU(100)),
model.add(Dense(1))
```

Week 9 – : Deep Learning
Applications of RNNs:

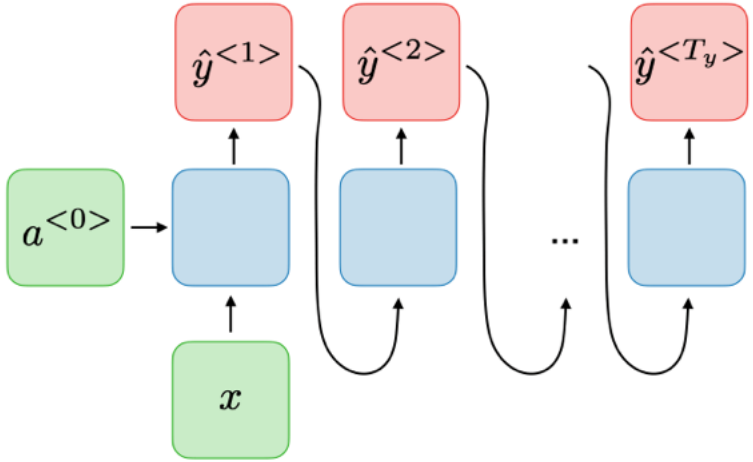
University of Salford MANCHESTER

RNN models are mostly used in the fields of natural language processing and speech recognition.

Type of RNN	Illustration	Example
One-to-one $T_x = T_y = 1$		Traditional neural network

Week 9 – : Deep Learning
Applications of RNNs:

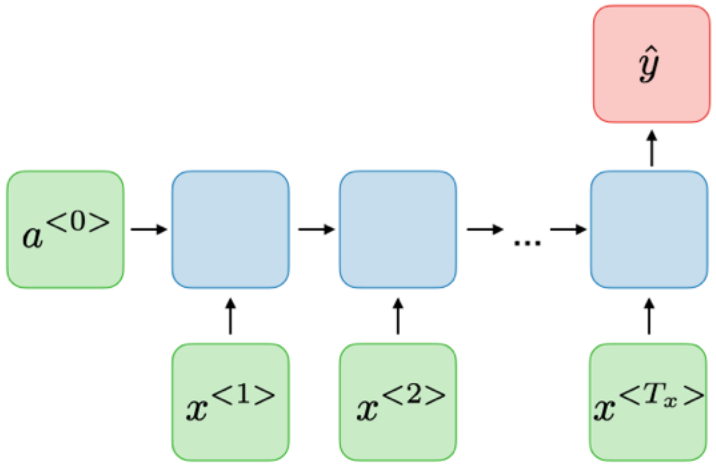
RNN models are mostly used in the fields of natural language processing and speech recognition.

Type of RNN	Illustration	Example
One-to-many $T_x = 1, T_y > 1$		Music generation

Week 9 – : Deep Learning
Applications of RNNs:

University of Salford MANCHESTER

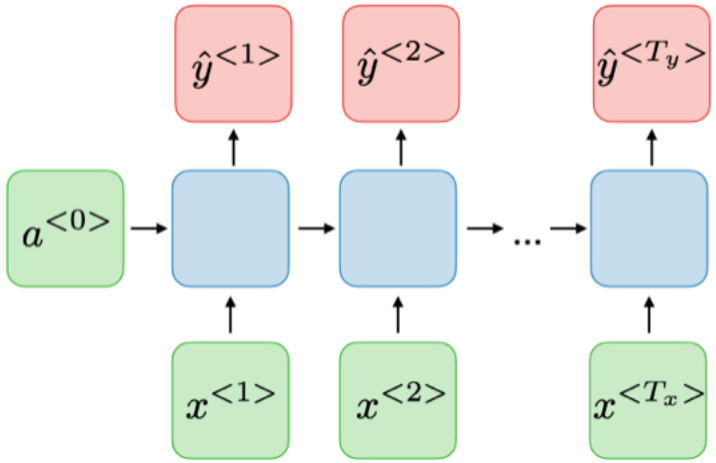
RNN models are mostly used in the fields of natural language processing and speech recognition.

Type of RNN	Illustration	Example
Many-to-one $T_x > 1, T_y = 1$		Sentiment classification

Week 9 – : Deep Learning
Applications of RNNs:

University of Salford MANCHESTER

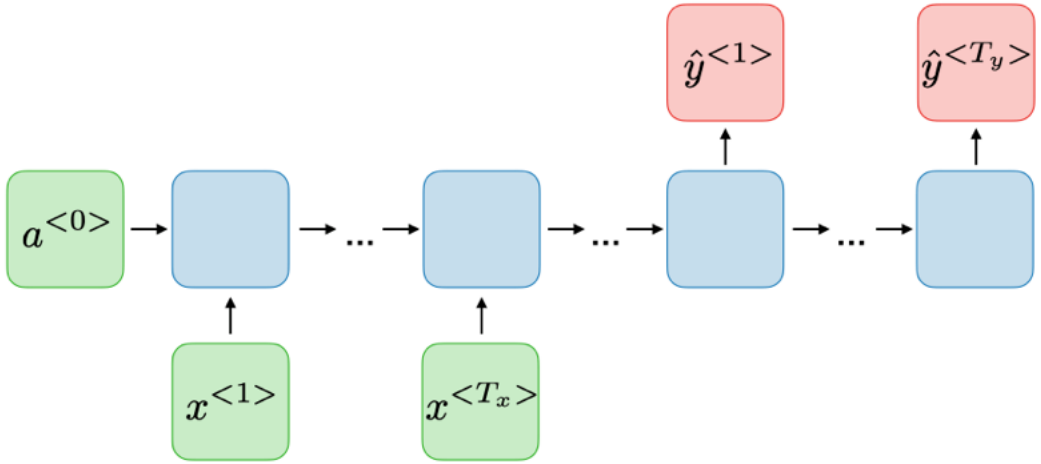
RNN models are mostly used in the fields of natural language processing and speech recognition.

Type of RNN	Illustration	Example
Many-to-many $T_x = T_y$		Name entity recognition

Week 9 – : Deep Learning
Applications of RNNs:

University of Salford MANCHESTER

RNN models are mostly used in the fields of natural language processing and speech recognition.

Type of RNN	Illustration	Example
Many-to-many $T_x \neq T_y$		Machine translation

Week 9 – : Deep Learning
Applications of RNNs: Example

University of Salford MANCHESTER

```
from numpy import sqrt
from numpy import asarray
from tensorflow.keras.layers import LSTM

# split a univariate sequence into samples
def split_sequence(sequence, n_steps):
    X, y = list(), list()
    for i in range(len(sequence)):
        # find the end of this pattern
        end_ix = i + n_steps
        # check if we are beyond the sequence
        if end_ix > len(sequence)-1:
            break
        # gather input and output parts of the pattern
        seq_x, seq_y = sequence[i:end_ix], sequence[end_ix]
        X.append(seq_x)
        y.append(seq_y)
    return asarray(X), asarray(y)
```

```
# load the dataset
path = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/monthly-car-sales.csv'
df = read_csv(path, header=0, index_col=0, squeeze=True)
df.tail()
```

```
Month
1968-08    16722
1968-09    14385
1968-10    21342
1968-11    17180
1968-12    14577
```

Week 9 – : Deep Learning
Applications of RNNs: Example

University of Salford MANCHESTER

```
# retrieve the values
values = df.values.astype('float32')
# specify the window size
n_steps = 5
# split into samples
X, y = split_sequence(values, n_steps)
# reshape into [samples, timesteps, features]
X = X.reshape((X.shape[0], X.shape[1], 1))
# split into train/test
n_test = 12
X_train, X_test, y_train, y_test = X[:-n_test], X[-n_test:], y[:-n_test], y[-n_test:]
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(91, 5, 1) (12, 5, 1) (91,) (12,)
```

Week 9 – : Deep Learning
Applications of RNNs: Example

University of Salford MANCHESTER

```
# define model
model = Sequential()
model.add(LSTM(100, activation='relu', kernel_initializer='he_normal', input_shape=(n_steps,1)))
model.add(Dense(50, activation='relu', kernel_initializer='he_normal'))
model.add(Dense(50, activation='relu', kernel_initializer='he_normal'))
model.add(Dense(1))
# compile the model
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
# fit the model
model.fit(X_train, y_train, epochs=350, batch_size=32, verbose=0, validation_data=(X_test, y_test))
# evaluate the model
mse, mae = model.evaluate(X_test, y_test, verbose=0)
print('MSE: %.3f, RMSE: %.3f, MAE: %.3f' % (mse, sqrt(mse), mae))
# make a prediction
row = asarray([18024.0, 16722.0, 14385.0, 21342.0, 17180.0]).reshape((1, n_steps, 1))
yhat = model.predict(row)
print('Predicted: %.3f' % (yhat))
```

Week 9 – : Deep Learning
Applications of RNNs: Example

University of Salford MANCHESTER

```
model.summary()
```

```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_3 (LSTM)                (None, 100)               40800
_________________________________________________________________
dense_19 (Dense)             (None, 50)                5050
_________________________________________________________________
dense_20 (Dense)             (None, 50)                2550
_________________________________________________________________
dense_21 (Dense)             (None, 1)                 51
=================================================================
Total params: 48,451
Trainable params: 48,451
Non-trainable params: 0
```

Week 9 : Deep Learning with NLP

University of Salford MANCHESTER

weather	weather	play
Sunny	Hot	No
Overcast	Hot	Yes
Sunny	Mild	No

```
[ ]  1 # Assigning features and label variables
     2 weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny',
     3 'Rainy','Sunny','Overcast','Overcast','Rainy']
     4 temp=['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mild']
     5 play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']
```

Week 9 : Deep Learning with NLP - One-hot encodings

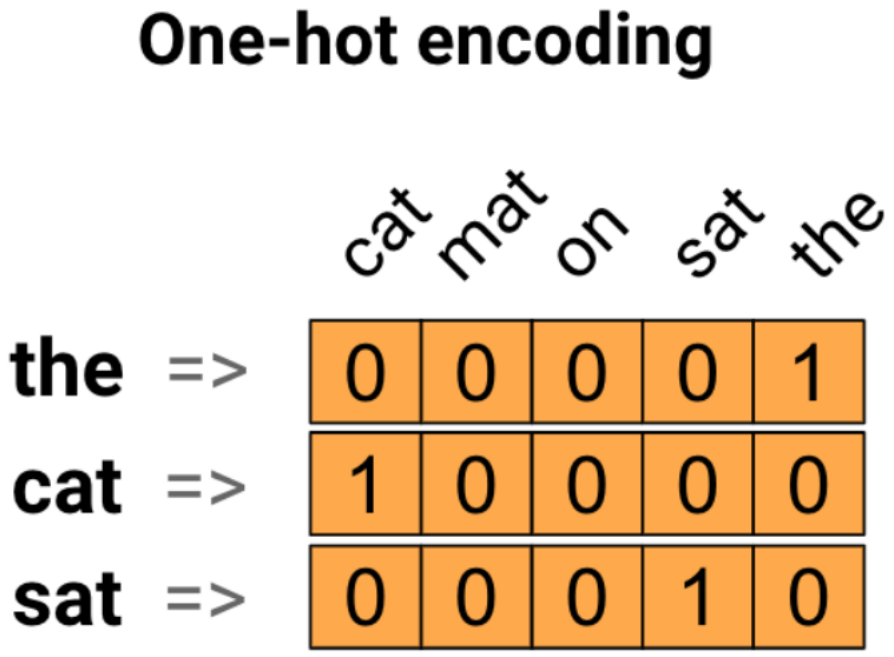
University of Salford MANCHESTER

```
# Converting strings into numbers.
le = preprocessing.LabelEncoder()
weatherEncode = le.fit(weather)
weather_encoded = weatherEncode.transform(weather)
print('weather',weather_encoded)


# Converting string temp into numbers
leT = preprocessing.LabelEncoder()
WorkDayEncode = leT.fit(WorkDay)
WorkDay_encoded = WorkDayEncode.transform(WorkDay)
print ('WorkDay',WorkDay_encoded)

leTT = preprocessing.LabelEncoder()
TimeDayEncode = leTT.fit(WorkDay)
TimeDay_encoded = TimeDayEncode.transform(WorkDay)
print ('WorkDay',TimeDay_encoded)

# Converting string labels into numbers
leL = preprocessing.LabelEncoder()
Lebal_encoded=leL.fit(TrafficJam)
label=Lebal_encoded.transform(TrafficJam)
print ("TrafficJam:",label)
```

```
weather [0 0 0 0 0 0 1 1 1 1 1 1 2 2 2 2 2 2]
WorkDay [1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0]
WorkDay [1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0]
TrafficJam: [1 0 1 0 0 0 1 1 1 0 0 0 1 1 1 1 0 1]
```

Week 9 : Deep Learning with NLP - One-hot encodings

The cat sat on the mat



Week 9 : Deep Learning with NLP - One-hot encodings

University of Salford MANCHESTER

Discrete Representation

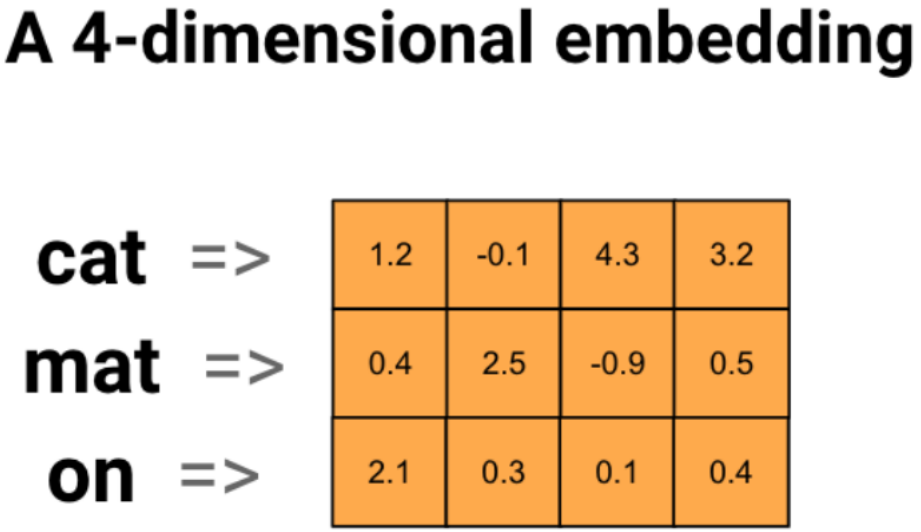
motel[0000000000010000]
AND
hotel [0000000100000000] = 0

Week 9 : Deep Learning with NLP - Word embeddings

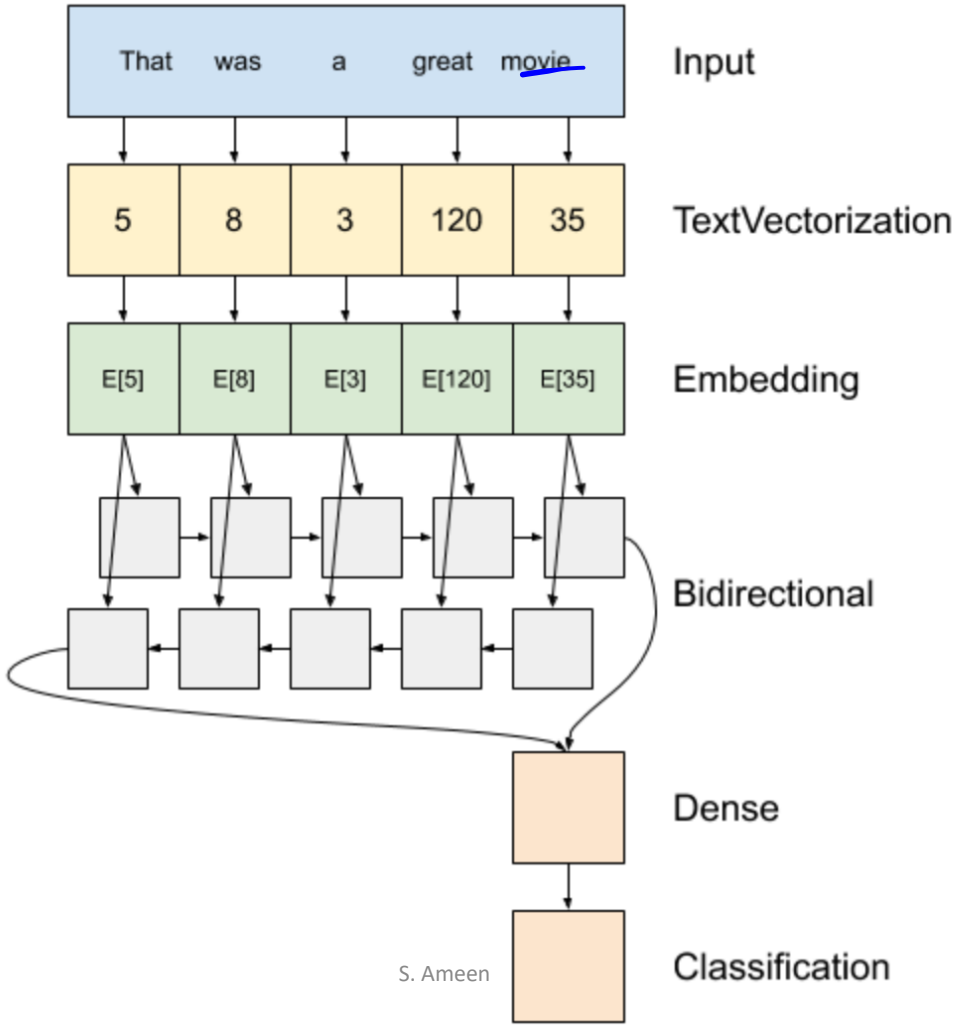
University of Salford MANCHESTER

Word embeddings

Word embeddings give us a way to use an efficient, dense representation in which similar words have a similar encoding.



Week 9 : Deep Learning with NLP - Word embeddings



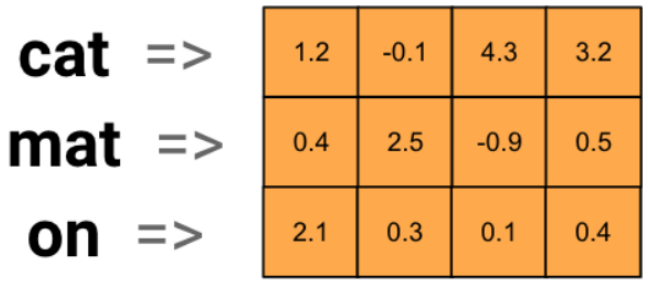
Week 9 : Deep Learning with NLP - Word embeddings

University of Salford MANCHESTER

```
import csv
import tensorflow as tf
import numpy as np
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

```
vocab_size = 1000
embedding_dim = 16
max_length = 120
trunc_type='post'
padding_type='post'
oov_tok = "<OOV>"
training_portion = .8
```

A 4-dimensional embedding




```
>>> sequence = [[1], [2, 3], [4, 5, 6]]
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence)
array([[0, 0, 1],
       [0, 2, 3],
       [4, 5, 6]], dtype=int32)
```

```
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence, value=-1)
array([[-1, -1,  1],
       [-1,  2,  3],
       [ 4,  5,  6]], dtype=int32)
```

```
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence, padding='post')
array([[1, 0, 0],
       [2, 3, 0],
       [4, 5, 6]], dtype=int32)
```

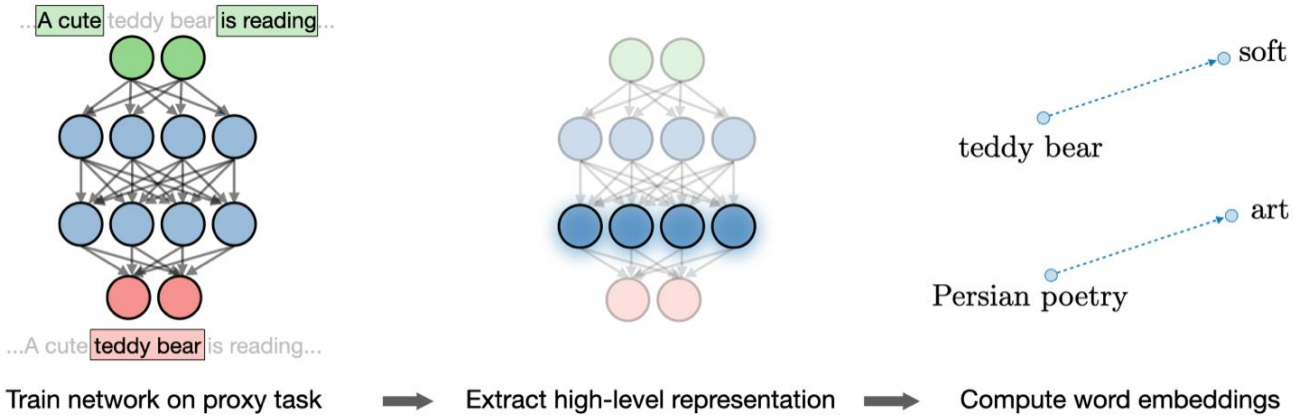
```
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence, maxlen=2)
array([[0, 1],
       [2, 3],
       [5, 6]], dtype=int32)
```

Week 9 : Deep Learning with NLP - word2Vec

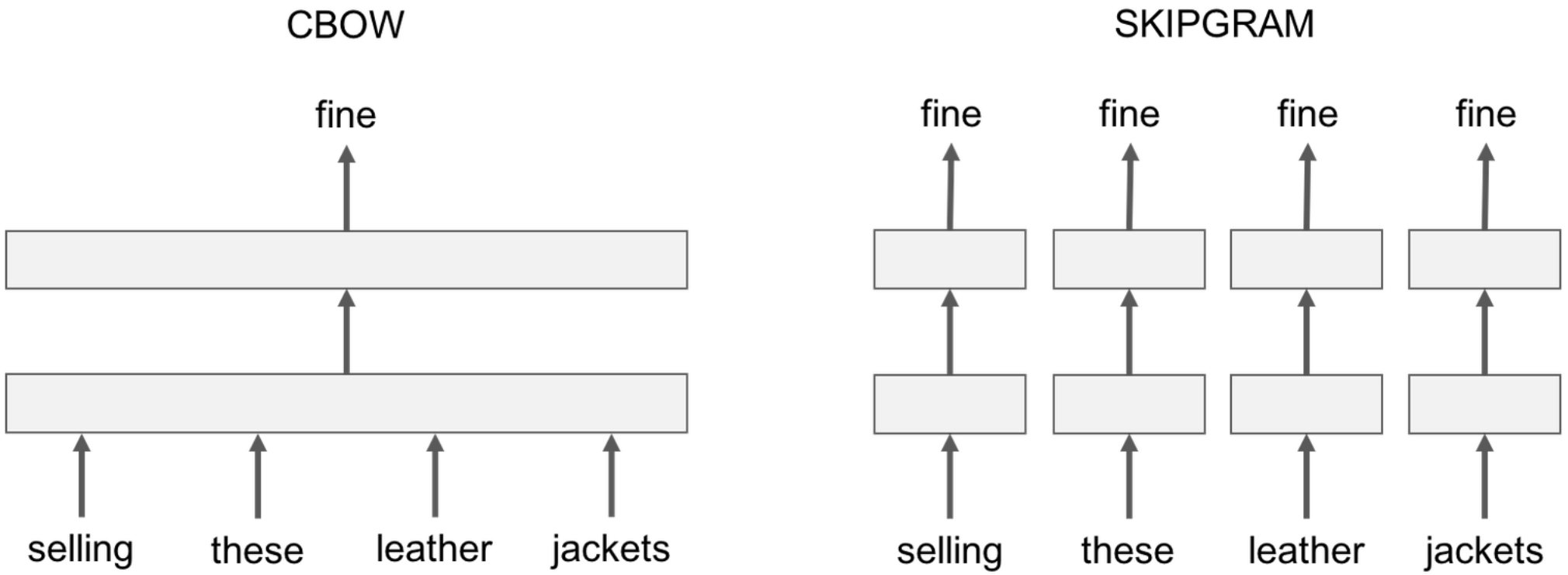
Word2Vec: There are two models that are commonly used to train these embeddings:

The Skip-gram model takes the input as each word in the corpus, sends them to a hidden layer (embedding layer) and from there it predicts the context words. Once trained, the embedding for a particular word is obtained by feeding the word as input and taking the hidden layer value as the final embedding vector.

The CBOW (Continuous Bag of Words) model takes the input the context words for the given word, sends them to a hidden layer (embedding layer) and from there it predicts the original word. Once again, after training, the embedding for a particular word is obtained by feeding the word as input and taking the hidden layer value as the final embedding vector.



Week 9 : Deep Learning with NLP – CBOW vs Skipgram



Week 9 : Deep Learning with NLP - GloVe: Global Vectors for Word Representation

University of Salford MANCHESTER

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

$$J(\theta) = \frac{1}{2} \sum_{i,j=1}^{W} f(P_{ij})(u_i^T v_j - \log P_{ij})^2$$

Week 9 : Deep Learning with NLP - GloVe:

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

leptodactylidae

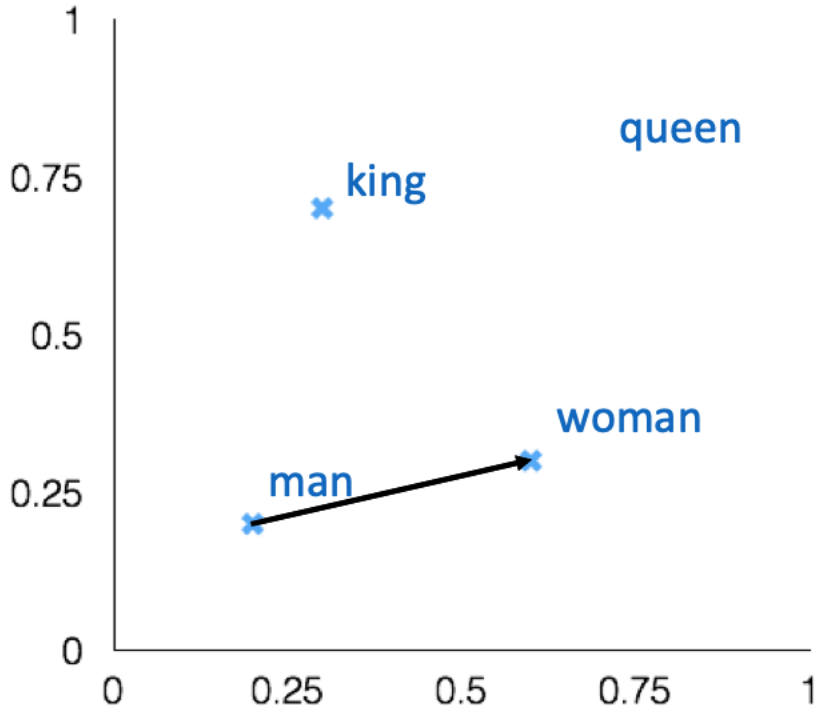
rana

eleutherodactylus

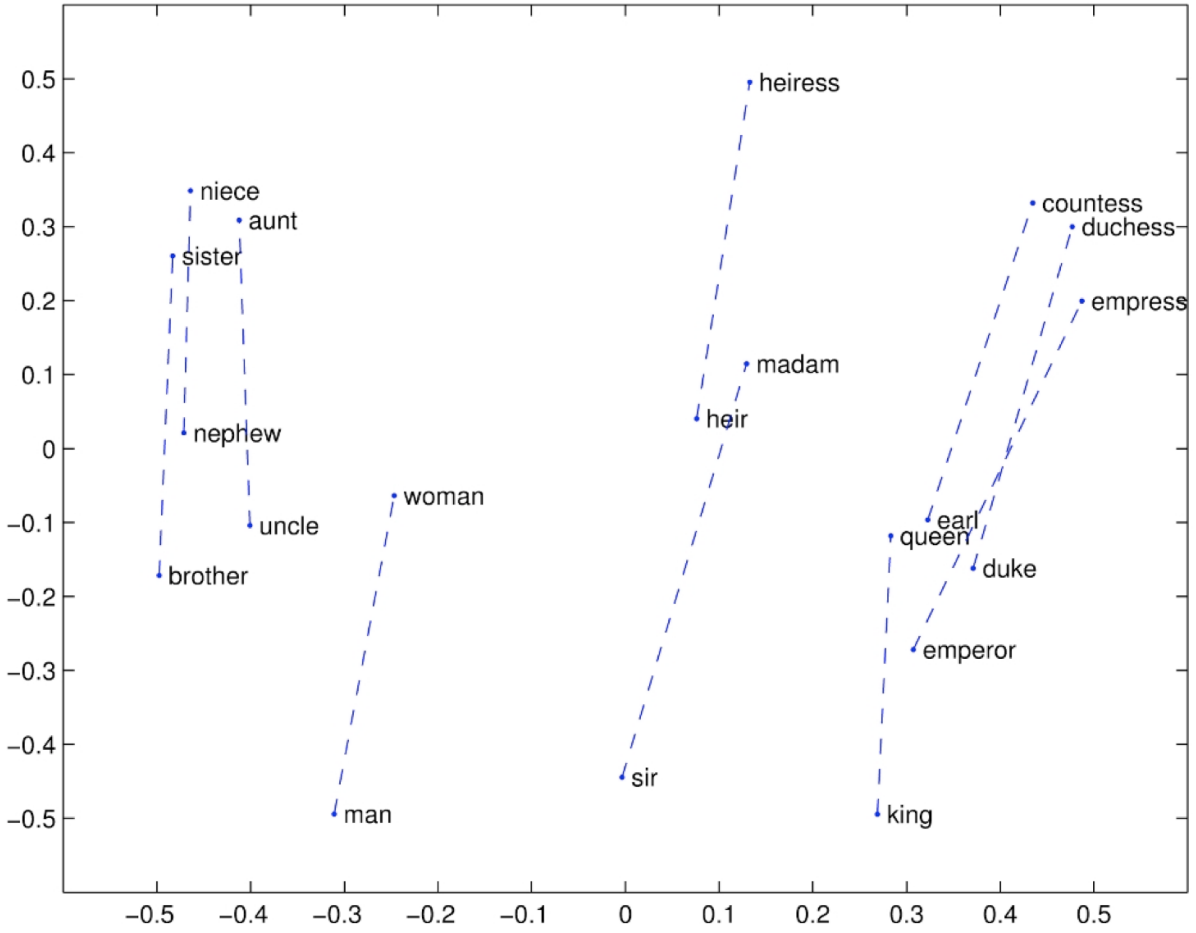
Week 9 : Deep Learning with NLP - GloVe:

man:woman :: king:?

+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]

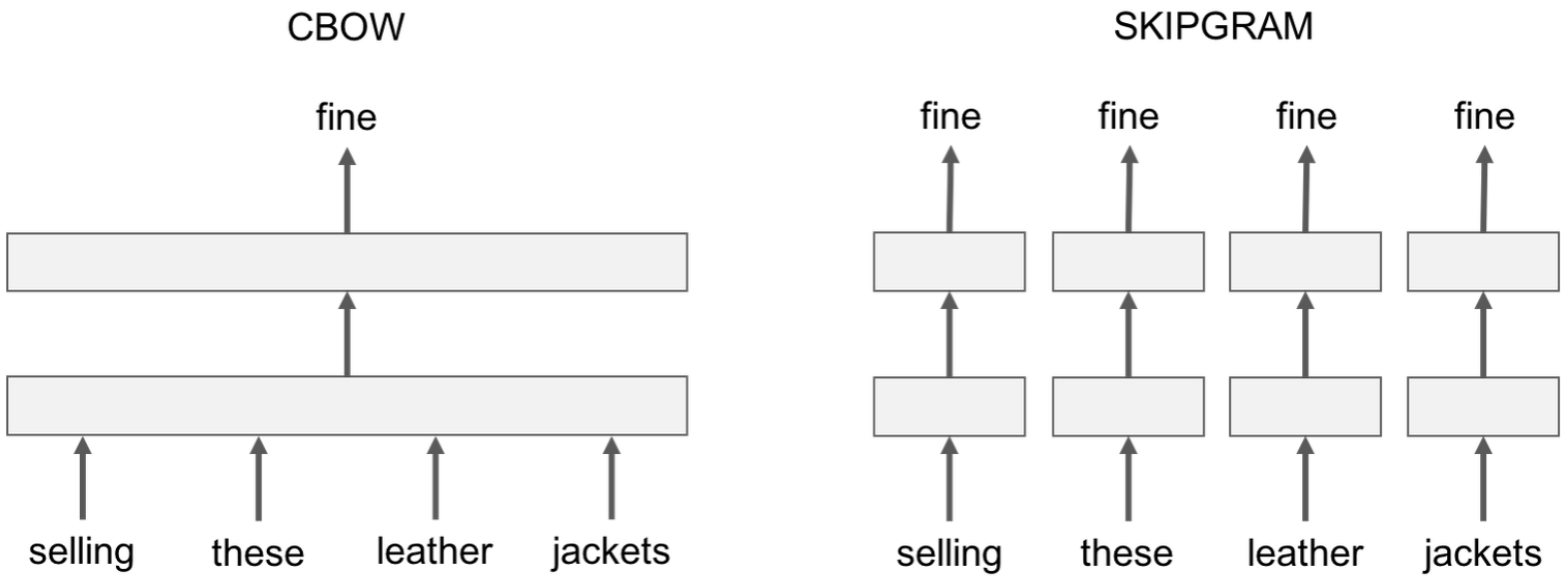


Week 9 : Deep Learning with NLP - GloVe:



Week 9 : Deep Learning with NLP - FastText:

Fasttest: is a library for efficient learning of word representations and sentence classification.



import fastText

https://fasttext.cc/

Week 9 : Deep Learning with NLP :

University of Salford MANCHESTER

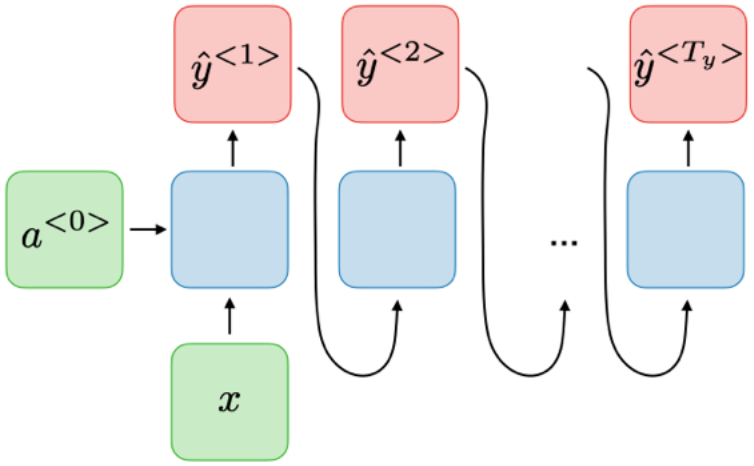
Demo

Week 9 : Deep Learning with NLP : Application :

University of Salford MANCHESTER

One-to-many

$T_x = 1, T_y > 1$



Music generation

Week 9 : Deep Learning with NLP : Application : - Shakespeare

University of Salford MANCHESTER

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

Week 9 : Deep Learning with NLP : Application : - Wikipedia

University of Salford MANCHESTER

```
Naturalism and decision for the majority of Arab countries' capitalide was grounded
by the Irish language by [[John Clair]], [[An Imperial Japanese Revolt]], associated
with Guangzham's sovereignty. His generals were the powerful ruler of the Portugal
in the [[Protestant Immineners]], which could be said to be directly in Cantonese
Communication, which followed a ceremony and set inspired prison, training. The
emperor travelled back to [[Antioch, Perth, October 25|21]] to note, the Kingdom
of Costa Rica, unsuccessful fashioned the [[Thrales]], [[Cynth's Dajoard]], known
in western [[Scotland]], near Italy to the conquest of India with the conflict.
Copyright was the succession of independence in the slop of Syrian influence that
was a famous German movement based on a more popular servicious, non-doctrinal
and sexual power post. Many governments recognize the military housing of the
[[Civil Liberalization and Infantry Resolution 265 National Party in Hungary]],
that is sympathetic to be to the [[Punjab Resolution]]
(PJS)[http://www.humah.yahoo.com/guardian.
cfm/7754800786d17551963s89.htm Official economics Adjoint for the Nazism, Montgomery
was swear to advance to the resources for those Socialism's rule,
was starting to signing a major tripad of aid exile.]]
```

Week 9 : Deep Learning with NLP : Application : - Algebraic Geometry (Latex)

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

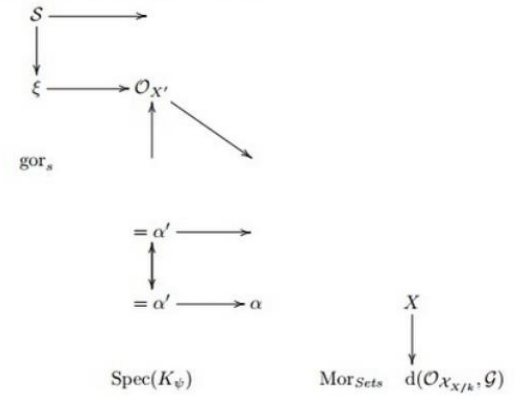
be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_t}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

is an isomorphism of covering of $\mathcal{O}_{X_t}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

Week 9 : Deep Learning with NLP : Application : - Linux Source Code

University of Salford MANCHESTER

```
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
  rw->name = "Getjbbregs";
  bprm_self_clearl(&iv->version);
  regs->new = blocks[(BPF_STATS << info->historidac)] | PFMR_CLOBATHINC_SECONDS << 12;
  return segtable;
}
```

Week 9 : Deep Learning with NLP : Application : - Generating Baby Names

University of Salford MANCHESTER

Rudi Levette Berice Lussa Hany Mareanne Chrestina Carissy Marylen Hammine Janye Marlise Jacacrie Hendred Romand Charienna Nenotto Ette Dorane Wallen Marly Darine Salina Elvyn Ersia Maralena Minoria Ellia Charmin Antley Nerille Chelon Walmor Evena Jeryly Stachon Charisa Allisa Anatha Cathanie Geetra Alexie Jerin Cassen Herbett Cossie Velen Daurenge Robester Shermond Terisa Licia Roselen Ferine Jayn Lusine Charyanne Sales Sanny Resa Wallon Martine Merus Jelen Candica Wallin Tel Rachene Tarine Ozila Ketia Shanne Arnande Karella Roselina Alessia Chasty Deland Berther Geamar Jackein Mellisand Sagdy Nenc Lessie Rasemy Guen Gavi Milea Anneda Margoris Janin Rodelin Zeanna Elyne Janah Ferzina Susta Pey Castina

Week 9 : Deep Learning with NLP : Application : - Poem

University of Salford MANCHESTER

These are some random sampling generated by the network

ما أنا من أبيوالشَّوقُ يُسْرَجُ بالأمورِ الأخيرولقد أَرَدْتُ به العَرَاقَ ومِنْ أَنْيَتَنَادَمَ الأَرْضُ وَبَيْنَ المُحَالِوَالْمَعَالِي
وَالأَمَانِي وَالْعُلُومُوَالْحَيَا فِي الْقَلْبِ أَنْ لَمْ تَسْتَرِيحِ

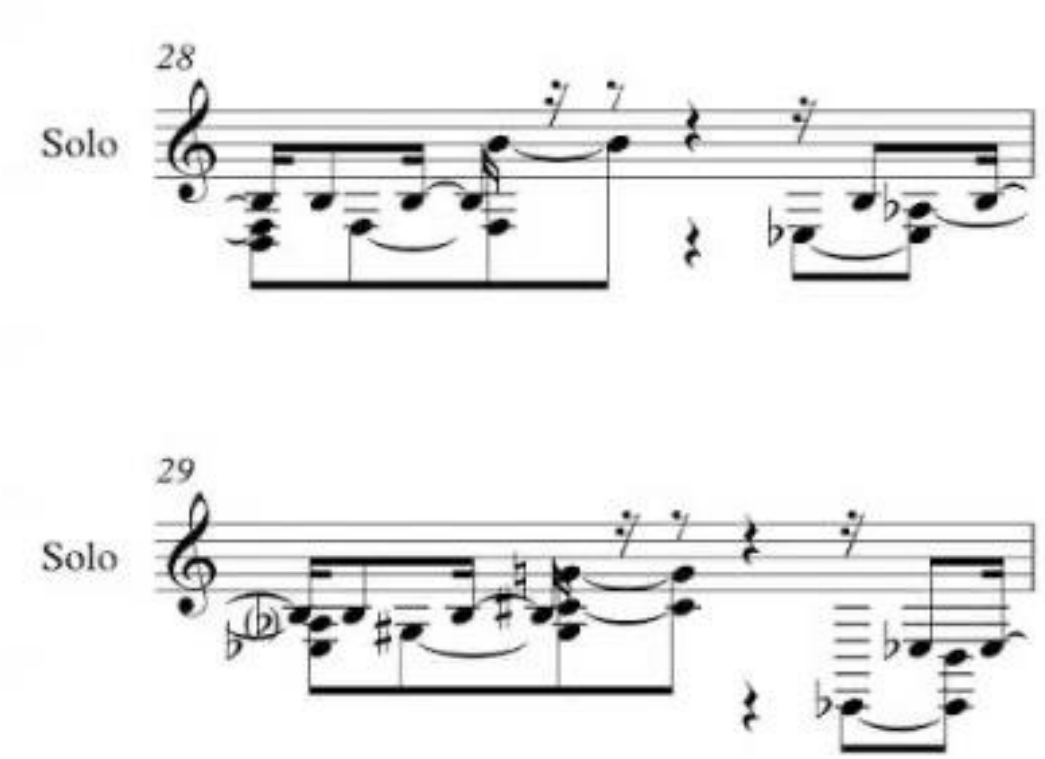
تَمَنَّيْتُ الْحَرِيَّ وَلَمْ أَبْقَ بِهَابِالْمَاءِ مِنْ مَاءِ الْمُحَالِ الْمُسْتَرِيبِأَنْتَ ==========
مَا أَرْدَاكَ أَنْ يَتَقَرَّبَاوَالْمَاءُ يَحْسَبُ بَعْدَ العَدَلِ الأَنْفُسِوَالْعَالِمُ الْحُبُّ عَلَى الْمَاءِ الْغُرَابِوَالْعَيْنُ يَعْلُو لَهُ عَلَى الْعَدْلِ
مُسْتَنْهَلُوَالْحُبُّ يَسْتَبْقِي بِهِ الْعَدْلُ فَالْهَوَىوَالْعُلَى وَالْخَيْرُ يَعْرِفُ الأَحْدَادُوَالْعَيْنُ يُسْمِعُ مِنْ سَمَاءِ الْمَجْدِ مَا أَوْدَعَتْمِنْ
حَرْبِهِ فِي الْمُسْتَبَاحِ وَجَادَهُوَمَا بِكَ الْمُسْتَسْلِمُ الْمُتَسَاقِيوَالْغَرَامُ وَالْحَيْثُ يَدْرُو مَاءَ الْوَرَىوَالْمَرْءُ يُرْدَى بِالْمُهَاجِرِ
وَالْمَعَالِيوَالْعُمْرُ أَوْ لِلْمَاءِ مُحْتَمِلُوَالرَّمْسُ يَسْمُو بِهِ الْغَدَاةُوَتَسْرِي بِالْقَرَائِنِ وَالْمَعَالِيوَأَجَارُ الْحَسْبِ مَا
بَاتَ==========

والصَّمتُ الذييُرْجِي الرَّجاءَ به الجَوادُ الرَّاقيوالشَّمسُ يَعْرِفُ أَنَّهُ الأرضُ الذييَحْرُقُ الأَمْرَ الذي أَخْرَجَ
الخُلْدُوَأَعْطَى الأَرْضَ مِنْ بَعْضِ الأَرَاكِ وَلَمْ تَرُدْإلى الأَرْضِ مِنْ أَقْبَالِهِ وَتَرَامَىوَلَمْ أَدْرِ مِنْ أَنْجَائِهِ وَالْعَدَائِمُوَأَشْرَقَ مِنْ
عَيْنٍ وَلَمْ يَعْرِفِ الْعَدَابِأَحْزَانِ مَا يَشْرِي الْكُرُومَ الْمَوَاعِدُوَلَمْ تَرْجِعِ الأَرْضُ الْمَنَايَا وَأَعْتَرَتْبِهِ مِنْ لَوَاءِ الْخَيْرِ وَالْمَاءُ
مَائِدُوَتَبْلُغُ فِي الأَرْضِ الْحُبَائِبِ وَالْحَبْرُوَأَصْبَحْتُ أَحْيَا لَيْسَ يَعْرِفُ أَنَّنِيلَعَمْرُكَ أَرْضَى مِنْ بَنِي مَاءِ مَائِدِوَإِنْ كُنْتُ مِنْ
أَعْنَاقِهِ وَبَلِيَّةً تَرَى كَيْفَ يَكْرِمُ لَوْ تَرَى الْعَارِ مِنْ عَدَاوَأَعْرَضَ مِنْ حَالٍ وَأَرْضِي بِأَبْرَقِوَلَمْ أَكُ مِنْ مَاءٍ وَلاَ أَوْ تَرَى
بِهِوَأَعْطِي بِهِ الأَرْضَ الْحُبَابُ بِأَبْصَرِالعصر الإسلامي >> مروان ابن أبي حفصة >> أَتَانِي كَأَنَّ العَدَاءَ الْحَبِيبُرقم
القصيدة : 27920----------يا رَبَّ أَسْرَى لَيْسَ يَحْسُبُهُ الأَمْرُرقم القصيدة : 17967-----
أَصَابَ الْحَيَاةُ وَالْخَلْقُ مُرْتَجِعٌ----------

Salem Ameen
Sessional Lecturer at The University of Salford
1 article

Inspired by the work of **Andrej Karpathy blog** for training multi-layer Recurrent Neural Network (RNN, LSTM, and GRU) and then sampling form the model. I trained **char-rnn** to many of Arabic poems and surprisingly the network can match Arabic diacritics which is difficult to match to many people but the network learns to generate them. I trained the file with size of nearly 93MB. LSTM with 3 layers, the size of the network 700, 0.5 dropout while number of parameters in the model: 10341950

Week 9 : Deep Learning with NLP : Application : - Poem



Week 9 : AI

- Thank You