University College Dublin
An Coláiste Ollscoile, Baile Átha Cliath

SPRING, 21/22 TRIMESTER EXAMINATIONS

COMP47590

Advanced Machine Learning

Module Coordinator: Assoc Professor Brian Mac Namee

Student Number

Seat Number

Time Allowed: 120 minutes

Materials Permitted in the Exam Venue:
Non-programmable or scientific calculator
Programmable calculator

Materials to be Supplied to Students:
8 Page Answer Booklets
New Cambridge Statistical Tables

Instructions to Students:
Answer any three out of four questions. All questions carry equal marks. Total marks available 90. The value of each part of each question is shown in brackets next to it.

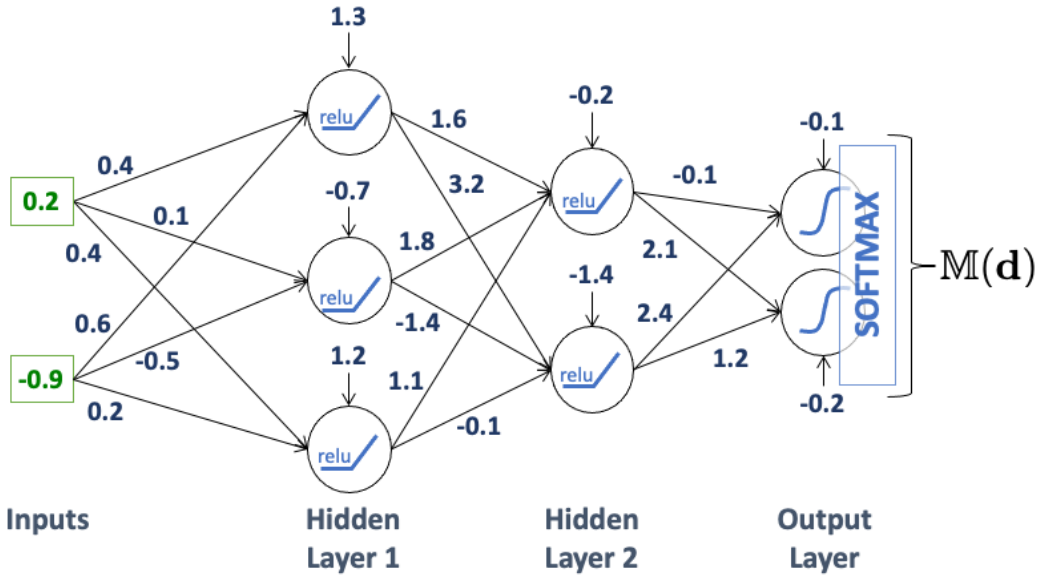
SOLUTIONS

SOLUTIONS

SOLUTIONS

SOLUTIONS

1. **(a)** The image below shows a *feed forward artificial network*. The computational units in the two hidden layers use *rectified linear* (*relu*) activation functions and the output layer unit uses a *softmax* activation function. The *weights* and *biases* are shown along the links in the network.



(i) Perform a **forward propagation** through the network using an input feature vector of [0.2, -0.9]. Show your workings.

[12 marks]

The network inputs:

$$\mathbf{d} = \begin{bmatrix} 0.2 \\ -0.9 \end{bmatrix}$$

Weights and biases for Layer 1:

$$\mathbf{W}^{[1]} = \begin{bmatrix} 0.4 & 0.1 \\ 0.1 & -0.5 \\ 0.4 & 0.2 \end{bmatrix}$$

$$\mathbf{b}^{[1]} = \begin{bmatrix} 1.3 \\ -0.7 \\ 1.2 \end{bmatrix}$$

Weights and biases for Layer 2:

$$\mathbf{W}^{[2]} = \begin{bmatrix} 1.6 & 1.8 & 1.1 \\ 3.2 & -1.4 & -0.1 \end{bmatrix}$$

$$\mathbf{b}^{[2]} = \begin{bmatrix} -0.2 \\ -1.4 \end{bmatrix}$$

Weights and biases for Layer 3:

$$\mathbf{W}^{[3]} = \begin{bmatrix} -0.1 & 2.4 \\ 2.1 & 1.2 \end{bmatrix}$$

$$\mathbf{b}^{[3]} = \begin{bmatrix} -0.1 \\ -0.2 \end{bmatrix}$$

To perform a forward propagation for the first layer in the network, first calculate $ extbf{z}^{[1]}$:

$$\begin{aligned}\mathbf{z}^{[1]} &= \mathbf{W}^{[1]}\,\mathbf{d} + \mathbf{b}^{[1]} \\ &= \begin{bmatrix} 0.4 & 0.1 \\ 0.1 & -0.5 \\ 0.4 & 0.2 \end{bmatrix} \begin{bmatrix} 0.2 \\ -0.9 \end{bmatrix} + \begin{bmatrix} 1.3 \\ -0.7 \\ 1.2 \end{bmatrix} \\ &= \begin{bmatrix} 1.29 \\ -0.23 \\ 1.1 \end{bmatrix}\end{aligned}$$

then apply the activation function, in this case a relu function, to calculate the activation of the nodes at Layer 1:

$$\begin{aligned}\mathbf{a}^{[1]} &= \mathrm{g}(\mathbf{z}^{[1]}) \\ &= \mathrm{g}\left(\begin{bmatrix} 1.29 \\ -0.23 \\ 1.1 \end{bmatrix}\right) \\ &= \begin{bmatrix} 1.29 \\ 0.0 \\ 1.1 \end{bmatrix}\end{aligned}$$

To perform a forward propagation for the second layer in the network, first calculate $ extbf{z}^{[2]}$:

$$\begin{aligned}\mathbf{z}^{[2]} &= \mathbf{W}^{[2]}\,\mathbf{a}^{[1]} + \mathbf{b}^{[2]} \\ &= \begin{bmatrix} 1.6 & 1.8 & 1.1 \\ 3.2 & -1.4 & -0.1 \end{bmatrix} \begin{bmatrix} 1.29 \\ 0.0 \\ 1.1 \end{bmatrix} + \begin{bmatrix} -0.2 \\ -1.4 \end{bmatrix} \\ &= \begin{bmatrix} 3.074 \\ 2.618 \end{bmatrix}\end{aligned}$$

then apply the activation function, in this case a extbf{relu}, to calculate the activation of the output nodes of the network:

$$\begin{aligned}\mathbf{a}^{[2]} &= \mathrm{g}(\mathbf{z}^{[2]}) \\ &= \mathrm{g}\left(\begin{bmatrix} 3.074 \\ 2.618 \end{bmatrix}\right) \\ &= \begin{bmatrix} 3.074 \\ 2.618 \end{bmatrix}\end{aligned}$$

To perform a forward propagation for the third layer in the network, first calculate $ extbf{z}^{[3]}$:

$$\begin{aligned}\mathbf{z}^{[3]} &= \mathbf{W}^{[3]}\,\mathbf{a}^{[2]} + \mathbf{b}^{[3]} \\ &= \begin{bmatrix} -0.1 & 2.4 \\ 2.1 & 1.2 \end{bmatrix} \begin{bmatrix} 3.074 \\ 2.618 \end{bmatrix} + \begin{bmatrix} -0.1 \\ -0.2 \end{bmatrix} \\ &= \begin{bmatrix} 5.876 \\ 9.397 \end{bmatrix}\end{aligned}$$

then apply the activation function, in this case a extbf{softmax function}, to calculate the activation of the output nodes of the network:

$$\begin{aligned}\mathbf{a}^{[3]} &= \mathrm{g}(\mathbf{z}^{[3]}) \\ &= \mathrm{g}\left(\begin{bmatrix} 5.876 \\ 9.397 \end{bmatrix}\right) \\ &= \begin{bmatrix} 0.029 \\ 0.971 \end{bmatrix}\end{aligned}$$

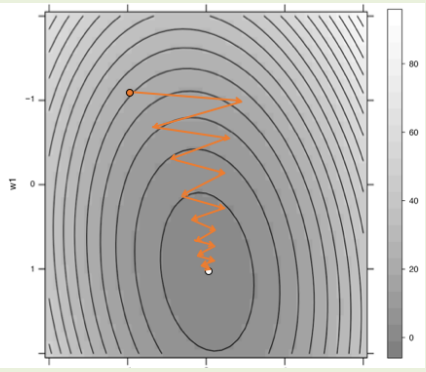
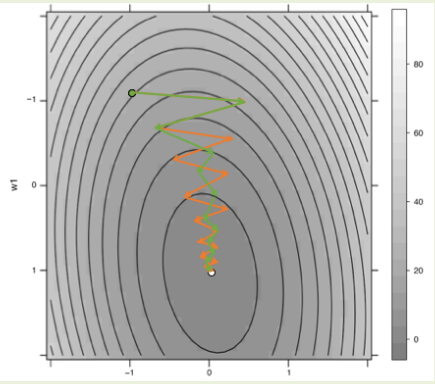
(ii) If the target feature vector for the current input vector is [1.0, 0.0], calculate the **loss** associated with this training instance using **cross entropy loss**.

[2 marks]

Calculate cross entropy loss

Loss = -(1*log(0.0287) + 0*log(0. 0.9713))
= 3.5503

(b) **Gradient descent with momentum**, **RMSprop**, and **adam** are three common adaptations to the basic gradient descent algorithm used to train

		neural networks. Explain how these approaches improve upon basic gradient descent and how they differ from each other.
		[8 marks]
		Sample Answer (Diagrams such as those included in this answer would be useful.) The gradient descent algorithm optimizes network weight values during a journey across an error (or loss) surface.  All of the adaptations listed improve this process by taking a more direct route across the error surface. This is achieved by adding different types of momentum terms. **Gradient descent with momentum** adds a momentum term to the gradient descent process to avoid sudden changes in exploration of the error surface. A diagram such as the following would help.  This is achieved by using an exponentially weighted moving average applied to the gradient terms over subsequent iterations. $$v_{d\mathbf{W}} = \beta v_{d\mathbf{W}} + (1-\beta)\, d\mathbf{W}$$ $$v_{d\mathbf{b}} = \beta v_{d\mathbf{b}} + (1-\beta)\, d\mathbf{b}$$ Weight and bias terms are then updated with these averaged gradients rather than raw gradient values.

$$\mathbf{W} = \mathbf{W} - \alpha v_{d\mathbf{W}}$$
$$\mathbf{b} = \mathbf{b} - \alpha v_{d\mathbf{b}}$$

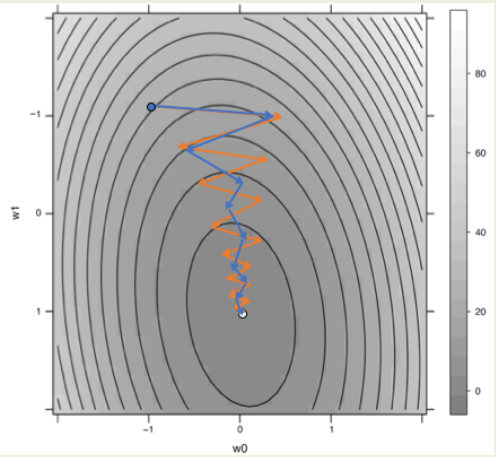
RMSprop builds on top of the same idea as gradient descent with momentum but uses the square of the gradients in the exponentially weighted moving average.

$$v_{d\mathbf{W}} = \beta\, v_{d\mathbf{W}} + (1-\beta)\, d\mathbf{W}^{*2}$$
$$v_{d\mathbf{b}} = \beta v_{d\mathbf{b}} + (1-\beta)\, d\mathbf{b}^{*2}$$

These values are then used in the weight and bias update rules.

$$\mathbf{W} = \mathbf{W} - \alpha \frac{d\mathbf{W}}{\sqrt{v_{d\mathbf{W}}} + \epsilon}$$
$$\mathbf{b} = \mathbf{b} - \alpha \frac{d\mathbf{b}}{\sqrt{v_{d\mathbf{b}}} + \epsilon}$$

This gives a more direct route across the error surface.



Adam mixes the gradient descent with momentum and RMSprop approaches in a weighted sum:

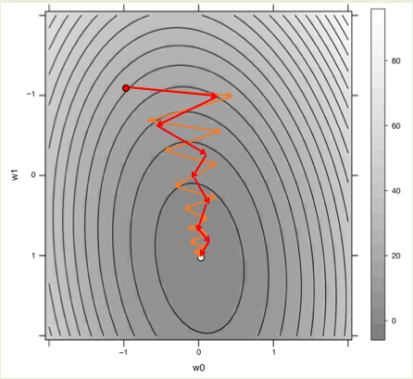
$$v_{d\mathbf{W}} = \beta_1 v_{d\mathbf{W}} + (1-\beta_1)\, d\mathbf{W}$$
$$v_{d\mathbf{b}} = \beta_1 v_{d\mathbf{b}} + (1-\beta_1)\, d\mathbf{b}$$

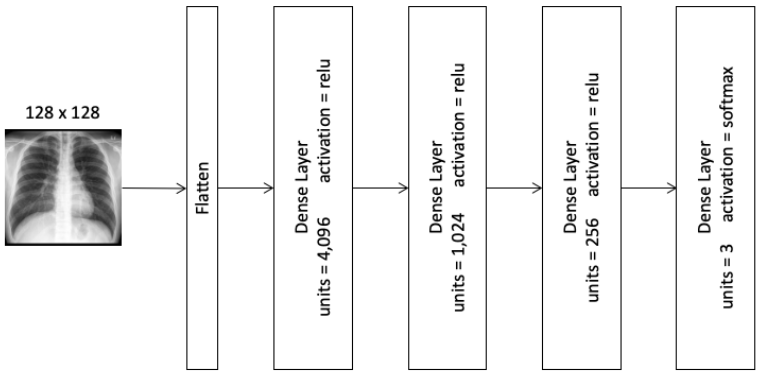
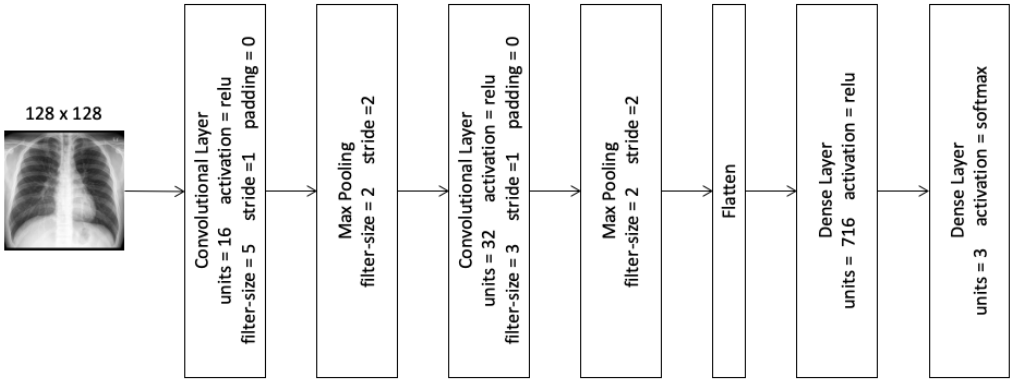
Gradient Descent with Momentum

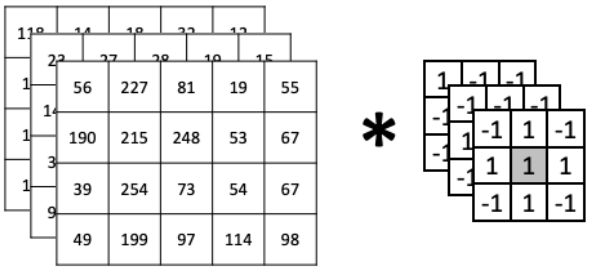
$$s_{d\mathbf{W}} = \beta_2\, s_{d\mathbf{W}} + (1-\beta_2)\, d\mathbf{W}^{*2}$$
$$s_{d\mathbf{b}} = \beta_2 s_{d\mathbf{b}} + (1-\beta_2)\, d\mathbf{b}^{*2}$$

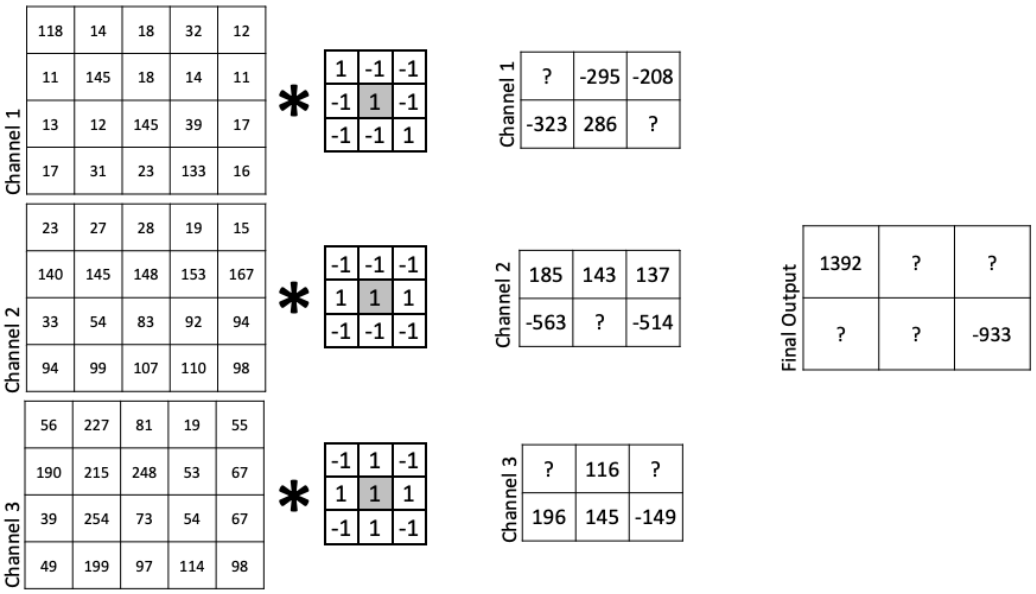
RMSprop

$$\mathbf{W} = \mathbf{W} - \alpha \frac{v_{d\mathbf{W}}}{\sqrt{s_{d\mathbf{W}}} + \epsilon}$$
$$\mathbf{b} = \mathbf{b} - \alpha \frac{v_{d\mathbf{b}}}{\sqrt{s_{d\mathbf{b}}} + \epsilon}$$

		This gives a very efficient route across an error surface. 
	(c)	Regularisation can be defined as "*any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error*" (Goodfellow et al, 2016). Describe **two** regularisation approaches that can be used when training deep neural networks and explain who they help to reduce generalisation error.
		[8 marks]
		Students can describe any regularisation approach, but these might include: • L2 regularisation • Dropout • Data augmentation 4 marks awarded for each.

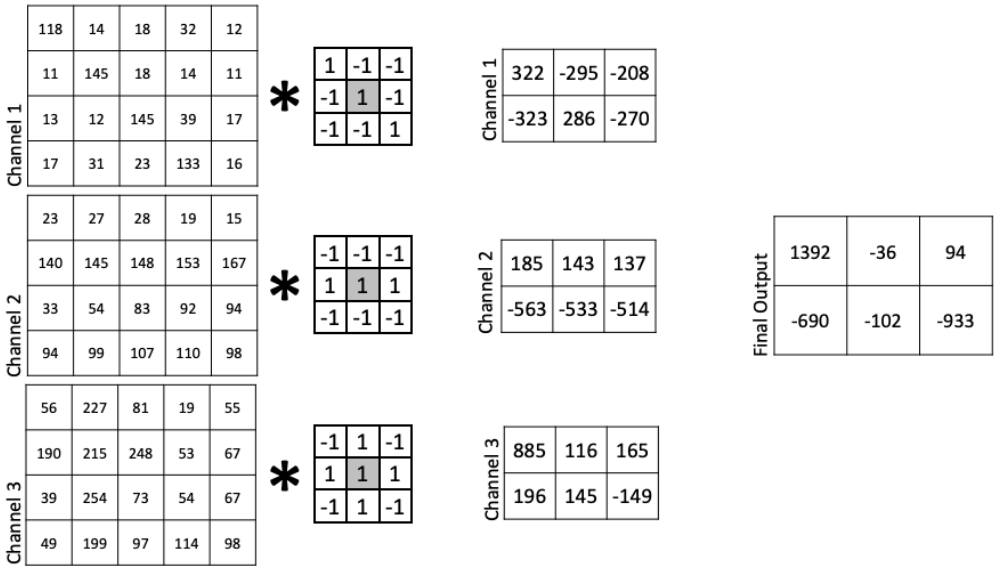
2.	**(a)**	You have been tasked with training a neural network to diagnose pneumonia from chest x-rays. The model should produce diagnoses on a three-point scale: *grade-0*= no pneumonia, *grade-1*= mild pneumonia, *grade-2*= severe pneumonia. The only input to the model is a 128 pixel by 128 pixel greyscale image of a chest x-ray. Image (a) shows the architecture of a multi-layer perceptron neural network designed for this problem. Image (b) shows the architecture of a convolutional neural network designed for this problem. Both architectures are composed of four layers.  (a) A multi-layer perceptron network for diabetes diagnosis  (b) A convolutional neural network for diabetes diagnosis Calculate the number of parameters (weights and biases) that need to be learned for each network architecture.
		[12 marks]
		<u>Sample Answer</u> **Multi-layer perceptron:** This is pretty straight-forward as it involves multiplying through the sizes of the network layers. Input: 128 * 128 = 16,384 Layer 1: 16,384 * 4,096 + 4,096 = 67,112,960 Layer 2: 4,096 * 1,024 + 1,024 = 4,195,328 Layer 3: 1,024 * 256 + 256 = 262,400

		Layer 4: 256 * 3 + 3 = 771 Total parameters: 71,571,459 **Convolutional neural network:** Students need determine the number of weights based on the size of each filter and the size of each layer. To calculate the number of activations at the flattening layer they also need to keep track of the number of activations flowing through the network. Layer 1 Dim: 128 x 128 x 1 Layer 1: 5 * 5 * 1 * 16 + 16 = **416** Layer 1 Output Dim: 124 x 124 x 16 Layer 1 Output Dim After Pooling: 62 x 62 x 16 Layer 2: 3 * 3 * 16 * 32 + 32 = **4,640** Layer 2 Output Dim: 60 x 60 x 32 Layer 2 Output Dim After Pooling: 30 x 30 x 32 Layer 2 Dimensionality after flattening: 30 x 30 x 32 = **28,800** Layer 3: 28,800 * 716 + 716 = **20,621,516** Layer 4: 716 * 3 + 3 = **2,151** Total parameters: **20,628,723**
	(b)	The image below shows a 3 channel input that is being convolved (cross correlated) with a 3 channel kernel.  The image below expands the three-channel input and three-channel kernel so that all values can be seen and shows the intermediate convolution result for each channel as well as the final output. Calculate the values marked with a **?** in the intermediate convolution results and the final output.



[7 marks]

Simple convolutions are calculated for each channel and then summed for the final output.

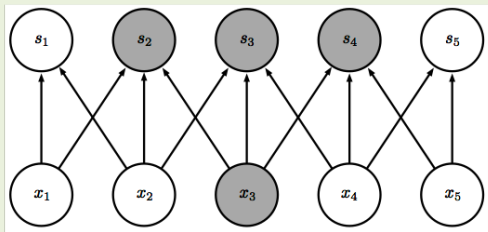
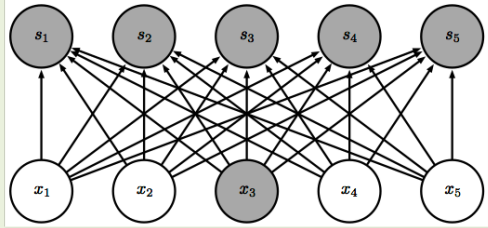
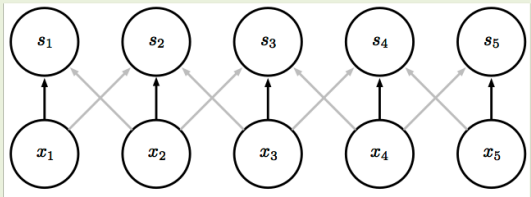
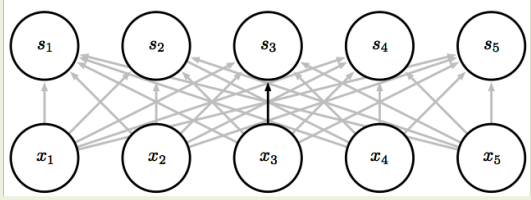


(c) The ability of convolutional networks to learn accurate models with many fewer weights than multi-layer perceptrons of similar depth is often attributed to **shared weights** and **sparse connections**. Explain the meaning of these two terms.

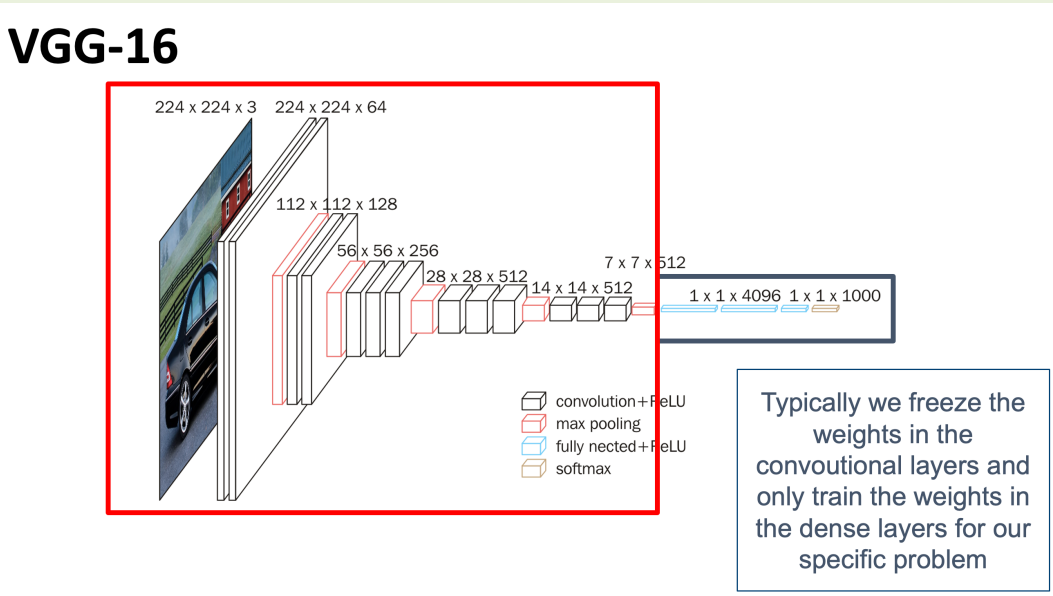
[5 marks]

Sample Answer

The connections between layers in a convolutional neural network are much less dense than the connections within simpler feed-forward networks (e.g. multi-layer perceptrons). The image below illustrates this.

		This arises from the fact that small convolutional kernels are used and so only a small number of the inputs to a network are actually connected to any hidden layer unit in the network. Sparse connections due to small convolution kernel  Dense connections  Convolutions are similarly responsible for the fact that weights in a CNN are shared. The bottom part of the image below shows the connections between two dense layers. In this scenario the weight on each link only affects one pair of computational units. In the CNN scenario above, however, the weights in the convolutional filter are applied at multiple positions within a network and so are shared across hidden units. Convolution shares the same parameters across all spatial locations  Traditional matrix multiplication does not share any parameters 
	(d)	**Transfer learning** using pre-trained weights has become a standard approach to training large convolutional neural networks for image classification. Explain what this means, and how it is helpful for training convolutional neural networks with small datasets.
		[6 marks]
		Students should provide an explanation similar to the following. Large network models (e.g. VGG-16), pre-trained on broad coverage datasets (e.g. ImageNet), can be loaded into most machine learning packages now. By doing this we can take advantage of what has been learned before. Typically all but the layers after the flattening layer in a network are frozen so that weights don't change during training and only the final layers are

trained. Something like the image below (albeit with much less detail) would be useful.



Students should then explain that by using pre-trained weights early layers of the network will be able to identify broadly useful features within images. These ought to be useful for any image classification problem. This is particularly useful when we have a small dataset for an image classification problem as we get the benefit of a large neural network, without needing the large dataset that is typically required to train such a network.

3.	**(a)**	Describe the concept of **discounted return** that is frequently used in reinforcement learning.
		[4 marks]
		Sample Answer Calculating the expected return from a sequence of actions is key in developing reinforcement learning systems. In a basic formulation of expected return that simply sums rewards, e.g. $$G = r_t + r_{t+1} + r_{t+2} + r_{t+3} + \ldots + r_e$$ expected future rewards are considered to be as valuable as the immediate reward that the agent will receive from taking the next immediate action. Just like we might be more excited about receiving a gift of $100 today than a promise to receive a gift of $100 in a year's time, it is reasonable when calculating expected return to pay more attention to the immediate reward we expect to receive from taking the next action, than to the rewards that we expect to receive in 10 or even 100 action's time. This is known as discounted return. We can define discounted return as: $$G_\gamma = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots + \gamma^{e-t} r_e$$ where γ is a discount factor that can take a value between 0 and 1.
	(b)	An intelligent agent trained to play a video game completes an episode and receives the following sequence of rewards over six timesteps: {r_0 = -33, r_1 = -11, r_2 = -12, r_3 = 27, r_4 = 87, r_5 = 156} Compare the discounted returns calculated at time t = 0 based on this reward sequence when discounting factors of 0.72 and 0.22 are used.
		[6 marks]
		Sample Answer Students should begin by calculating the discounted returns using: $$G_\gamma = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots + \gamma^{e-t} r_e$$ For discounting factor of 0.72: $$\begin{aligned} G = & (0.72^0 \times -33) + (0.72^1 \times -11) + (0.72^2 \times -12) \\ & + (0.72^3 \times 27) + (0.72^4 \times 87) + (0.72^5 \times 156) \\ = & (-33) + (0.72 \times -11) + (0.5184 \times -12) \\ & + (0.3732 \times 27) + (0.2687 \times 87) + (0.1935 \times 156) \\ = & \ 16.4985 \end{aligned}$$ For discounting factor of 0.22: $$\begin{aligned} G = & (0.22^0 \times -33) + (0.22^1 \times -11) + (0.22^2 \times -12) \\ & + (0.22^3 \times 27) + (0.22^4 \times 87) + (0.22^5 \times 156) \\ = & (-33) + (0.22 \times -11) + (0.0484 \times -12) \\ & + (0.0106 \times 27) + (0.0023 \times 87) + (0.0005 \times 156) \\ = & \ -35.4365 \end{aligned}$$

Students should then discuss that with the lower discount factor much less attention is paid to later rewards and so the overall return is much lower.

(c) As part of a project to develop a farm simulator, the behaviour of a cow has been observed over a day. During the day the cow can REST, GRAZE grass, PLAY with other cows, or enter the milking parlour to MILK. The behaviour of the cow over the course of a day is captured in the table below (with time flowing down through the columns and left to right).

1	REST	13	PLAY	25	PLAY	37	MILK	49	REST	61	REST
2	GRAZE	14	REST	26	REST	38	MILK	50	GRAZE	62	REST
3	REST	15	REST	27	REST	39	MILK	51	GRAZE	63	PLAY
4	REST	16	GRAZE	28	REST	40	REST	52	GRAZE	64	PLAY
5	REST	17	GRAZE	29	REST	41	REST	53	REST	65	PLAY
6	GRAZE	18	REST	30	MILK	42	REST	54	REST	66	GRAZE
7	GRAZE	19	PLAY	31	MILK	43	REST	55	GRAZE	67	GRAZE
8	GRAZE	20	GRAZE	32	MILK	44	REST	56	GRAZE	68	REST
9	REST	21	REST	33	MILK	45	REST	57	GRAZE	69	REST
10	REST	22	GRAZE	34	MILK	46	GRAZE	58	REST	70	REST
11	PLAY	23	GRAZE	35	MILK	47	GRAZE	59	GRAZE	71	REST
12	PLAY	24	PLAY	36	MILK	48	GRAZE	60	REST	72	REST

(i) Based on this behaviour sequence, calculate a **transition matrix** that gives the probability of moving between the four states.

[8 marks]

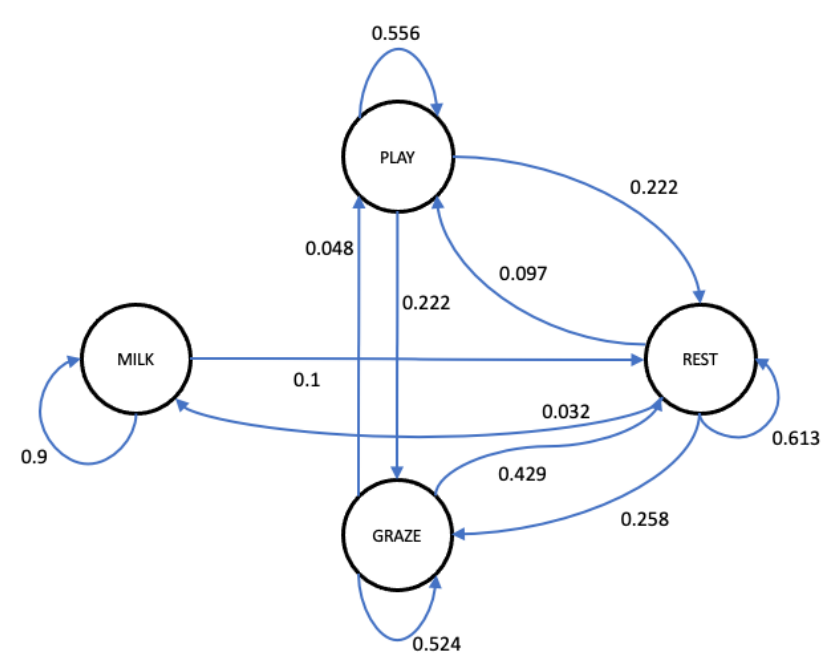
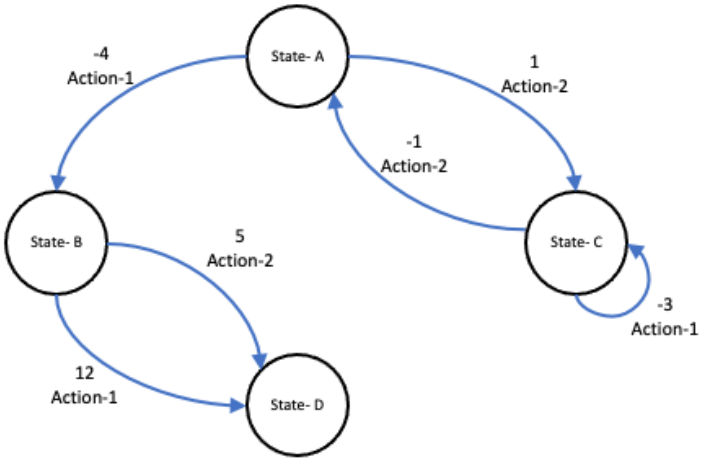
Sample Answer

The first step to building the transition matrix is to count the frequency of each possible state transition. Working down through the list of states we can count the number of times we move from one state to the next. This gives a transition frequency table:

	PLAY	MILK	REST	GRAZE
PLAY	5.0	0.0	2.0	2.0
MILK	0.0	9.0	1.0	0.0
REST	3.0	1.0	19.0	8.0
GRAZE	1.0	0.0	9.0	11.0

By normalising each row in the table (dividing each value by the sum of values in the row) we can calculate the final transition matrix:

	PLAY	MILK	REST	GRAZE
PLAY	0.556	0.000	0.222	0.222
MILK	0.000	0.900	0.100	0.000
REST	0.097	0.032	0.613	0.258
GRAZE	0.048	0.000	0.429	0.524

		(ii)	Draw a **Markov process diagram** to capture the behaviour of the cow as described by the transition matrix.
			[3 marks]
			Sample Answer A diagram like this one is appropriate for this answer. 
	(d)	The following image shows a simple state transition diagram for a domain in which an agent can occupy one of four states and has two actions available to it.  Actions that connect states are shown above the arrows in the diagram. This environment is fully deterministic (actions always lead to the states shown in the diagram above). The rewards associated with each state	

transition are shown in the diagram above the name of the action taken to complete the state transition.

The action-value-function-table for this environment is shown in the table below.

State	Action	Value
State-A	Action-1	3.24
State-A	Action-2	-1.24
State-B	Action-1	6.78
State-B	Action-2	4.56
State-C	Action-1	-12.56
State-C	Action-2	-2.87
State-D	Action-1	0
State-D	Action-2	0

(i) If the agent begins in State-C which action will it select following a **greedy action selection strategy**?

[3 marks]

Action-2 returns the highest reward so would be selected.

(ii) What state would the agent occupy after taking the action selected in Part (a) of this question and what reward would the agent receive after taking the action.

[3 marks]

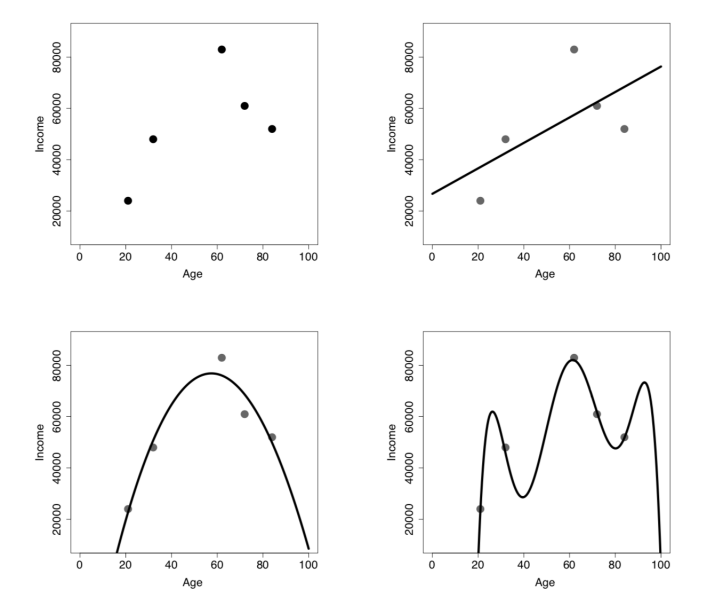
State-A would be the next state visited.

(iii) Assuming that **Q-learning** is being used, update the entry in the action value table above for State-C and the action selected in Part (a) of this question. In your calculations assume that $\alpha = 0.1$ and that $\gamma = 0.9$.

[3 marks]

The Q value can be calculated as follows.

Revised Q(State-C, Action-2)	=	-2.87	+	0.1	*(	-1	+	0.9	*	3.24	-	-2.87	
	=	-2.87	+	0.1	*(	4.786							
	=	-2.39											

4.	(a)	Machine learning algorithms face a constant struggle between **over-fitting** and **under-fitting**. Explain what this means.
		[10 marks]
		Sample Answer *1) What is meant by an ill-posed problem and what are the implications of this for machine learning.* • *Inductive machine learning algorithms essentially search through a hypothesis space to find the best hypothesis that is consistent with the training data used. It is possible to find multiple hypotheses that are consistent with a given training set (i.e. agrees with all training examples). It is for this reason that inductive machine learning is referred to as an ill-posed problem as there is typically not enough information in the training data used to build a model to choose a single best hypothesis. Inductive machine learning algorithms must somehow choose one of the available hypotheses as the **best**. An example like that shown in the figure below would be useful at this point*  *2) How do machine learning algorithms deal with the fact that machine learning is ill posed.* • *Because inductive learning is ill-posed, we have to make some extra assumptions to have a unique solution with the data we have. The set of assumptions we make to have learning possible is called the **inductive bias** of the learning algorithm - this is the main implication of inductive machine learning being ill-posed.* *3) Define what is meant by inductive bias:* • *The inductive bias of a learning algorithm:*

		1. is a set of assumption about what the true function we are trying to model looks like. *2. defines the set of hypotheses that a learning algorithm considers when it is learning.* *3. guides the learning algorithm to prefer one hypothesis (i.e. the hypothesis that best fits with the assumptions) over the others.* *4. is a necessary prerequisite for learning to happen because inductive learning is an ill posed problem.* • *An example of the specific inductive bias introduced by particular machine learning algorithms would be good here. E.g.:* o *Maximum margin: when drawing a boundary between two classes, attempt to maximize the width of the boundary. This is the bias used in Support Vector Machines. The assumption is that distinct classes tend to be separated by wide boundaries.* o *Minimum cross-validation error: when trying to choose among hypotheses, select the hypothesis with the lowest cross-validation error.* *4) The importance and difficulty of selecting the right inductive bias:* • *As learning is not possible without inductive bias the question becomes* ***how to choose the right bias?*** *This, however, is important and difficult because:* o *If the inductive bias of the learning algorithm constrains the search to only consider simple hypotheses, we may have excluded the real function from the hypothesis space. In other words, the true function is* ***unrealizable*** *in the chosen hypothesis space, (i.e., we are* ***underfitting****).* o *If the inductive bias of the learning algorithm allows the search to consider complex hypotheses, the model may hone in on irrelevant factors in the training set. In other words, the model with* ***overfit*** *the training data.*
	(b)	When developing a machine learning model that will be deployed to perform a task for a user, we can describe three different goals of evaluation: 1. to determine which model is the most suitable for a task 2. to estimate how the model will perform after deployment 3. to convince users that the model will meet their needs Describe the differences between these goals, and how the evaluation methods used to achieve each of them can be different.
		[10 marks]
		<u>Sample Answer</u> *Students should outline that when performing an evaluation to prepare a model for deployment the first goal is to determine which approach might work best. Decision s to be made based on this evaluation include which modelling approach will be used, which data pre-processing techniques*

		might be used, and the optimal algorithm hyper-parameters to be used. This evaluation is very much driven by the machine learning practitioner. Typically, k-fold cross validation can be used as the main evaluation method for this kind of evaluation. *Once all of the decisions about what modelling approach to take have been made the next goal attempts to evaluate the likely performance of a model after deployment. This is largely concerned with estimating generalisation error of the model. The only sensible way to evaluate this is to use a hold out test set that has been involved at all in training the model. This is the only reasonable way to evaluate generalisation error.* *Once practitioners are convinced that a modelling approach will achieve acceptable performance after deployment the last step is to convince the people for whom the model is being built that it will meet their needs. This can be done with the same kinds of experiments used to evaluate likely generalisation error, but often different performance measures need to be used so as to speak to a different audience (not machine learning practitioners). It is also important to consider a deployment experiment at this stage too.*
	(c)	The following is a definition of machine learning: *The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience.* - Tom Mitchell Do you believe that this definition accurately defines machine learning? In your answer discuss the appropriateness of the definition, the scope of the definition, and any recommendations for improvements you would suggest.
		[10 marks]
		<u>Sample Answer</u> This is an open ended question. To score well students should discuss the following points: • The GDPR refers to automated processing which will include machine learning • The GDPR refers to "*automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly effects him or her"*. This does not cover all applications of machine learning but only those that have a legal or significant effect on a data subject. For example, image classification for improving search (a large application area for deep learning) is unlikely to fall under this definition. • Many refer to a "*right to explanation*" within the GDPR but there is some debate over how much that is present.

		• If a right to explanation is required sophisticated machine learning models like deep neural networks will certainly be affected.