Reg. No.															

B.Tech. DEGREE EXAMINATION, DECEMBER 2022
Fifth Semester

18AIC301J – DEEP LEARNING TECHNIQUES
(For the candidates admitted from the academic year 2020-2021 and 2021 -2022)

Note:
(i) **Part - A** should be answered in OMR sheet within first 40 minutes and OMR sheet should be handed over to hall invigilator at the end of 40th minute.
(ii) **Part - B** should be answered in answer booklet.

Time: 2½ Hours | Max. Marks: 75

PART – A (25 × 1 = 25 Marks)
Answer **ALL** Questions

	Marks	BL	CO	PO
1. Where does the chemical reaction take place in neuron? (A) Dendrites (B) Axon (C) Synapses (D) Nucleus	1	1	1	1
2. Which of the following is correct for the neural network? (i) The training time is dependent on the size of the network (ii) Neural networks can be simulated on the conventional computers (iii) Artificial neurons are identical in operation to a biological one (A) All of the above (B) (ii) is true (C) (i) and (ii) are true (D) None of the above	1	2	1	2
3. A four input has weights 4, 3, 2 and 1. The transfer function is linear with the constant of proportionality = 2. The inputs are 4, 10, 5 and 20 respectively. What is the output? (A) 119 (B) 238 (C) 152 (D) 76	1	2	1	2
4. Why is the XOR problem is exceptionally interesting to neural network researchers? (A) It can be expressed in a way that allows you to use a neural network (B) It is complex binary operation that can not be solved using neural networks (C) It can be solved by single layer perceptron (D) It is the simplest linearly inseperable problem that exists	1	1	1	1
5. What is true regarding backpropagation rule? (A) It is also called generalized delta rule (B) Error in output is propagated backwards only to determine weight updates (C) There is no feedback of signal at any stage (D) All of the mentioned	1	1	1	1
6. _____ is a recommended model for pattern recognition in unlabeled data (A) CNN (B) Shallow neural network (C) Autoencoders (D) RNN	1	1	2	1

7. If your neural network model seems to have high bias, what is the promising thing to try? 1 2 2 1

(A) Add regularization (B) Get more test data
(C) Get more training data (D) Make the neural network deeper

8. What is weight delay? 1 1 2 1

(A) The process of gradually decreasing the learning rate during training (B) A technique to avoid vanishing gradient by imposing a ceiling on the values of the weights
(C) A regularization technique that weights results in gradient descent shrinking the weights on every iteration (D) Gradual corruption of the weights in the neural network if it is trained on noisy data

9. Which one is false a dropout? 1 2 2 2

(A) Dropout is implemented per layer in a network (B) Dropout is a hyperparameter
(C) Dropout can be used in input hidden and output layers (D) none of the above

10. Which of the following methods does not prevent a model from overfitting to the training set? 1 1 2 1

(A) Dropout (B) Pooling
(C) Early stopping (D) Data augmentation

11. Why is the pooling layer used in a convolutional neural network? 1 1 3 1

(A) They are of no use (B) Dimension reduction
(C) Object recognition (D) Image sensing

12. _____ computes the output volume by computing dot product between all filters and image patch. 1 1 3 1

(A) Input layers (B) Convolutional layer
(C) Activation function layer (D) Pool layer

13. _____ reduces each channel in the features map to a single value. 1 1 3 1

(A) Max pooling (B) Average pooling
(C) Global pooling (D) None of the above

14. What is the right order for a text classification model component? 1 2 3 2

(1) Text cleaning (2) text annotation (3) Gradient descent
(4) Model tuning (5) text-to-predictors

(A) 12345 (B) 13425
(C) 12534 (D) 13452

15. Convolution layers are powerful but also has big computational cost. Which of the following can make it cheaper? 1 2 3 2

(A) Bigger convolutions (B) Smaller convolutions
(C) Faster convolutions (D) Wider convolutions

16. What is a receptive filed? 1 1 4 1

(A) It is the name given to the connectivity of neurons only to a local region of the input volume
(B) It is the name given to the relationship of neurons

(C) It is the name given to the connectivity of neurons only to a local region of the output volume
(D) None of the above

17. _____ occurs when the gradients becomes very small and tend towards zero. 1 1 4 1
(A) Exploding gradient
(B) Vanishing gradient
(C) LSTM networks
(D) GRU networks

18. Which one of the following is false about LSTM? 1 1 4 1
(A) LSTM is an extension of RNN which extends its memory
(B) LSTM enables RNN to learn long-term dependencies
(C) LSTM solves the exploding gradients issue in RNN
(D) None of the above

19. Which kind of activation function is typical for a convolution layer in an RNN? 1 1 4 1
(A) Gaussian
(B) Sigmoid
(C) TANH
(D) RELU

20. Which of the following model contains internal memory? 1 1 4 1
(A) Convents
(B) Capsnets
(C) RNN
(D) Autoencoders

21. Consider a GAN which successfully produces images of apples. Which of the following propositions is false? 1 2 5 2
(A) The generator aims to learn the distribution of apple images
(B) The generator can produce unseen apple images
(C) The discriminator can be used to classify images as apples Vs non-apple
(D) After training the GAN, the discriminator loss eventually reaches a constant value

22. Which among the following is not an attention model? 1 1 5 1
(A) Bert
(B) Transformer
(C) Autoencoder
(D) Self-attention

23. For which task can Boltzman machine be used? 1 1 5 1
(A) Pattern mapping
(B) Feature mapping
(C) Classification
(D) Pattern association

24. GAN was developed and introduced by? 1 1 5 1
(A) Allan turning
(B) J. Goodfellow
(C) Rutherford
(D) None of the above

25. Select the correct option? 1 2 5 2
(i) Boltzmann machine are non-deterministic generative DL models with 3 types of nodes: visible, hidden and output.
(ii) Boltzman machines fall into the class of unsupervised learning.
(A) Both statements are true
(B) Statement (i) is true statement (ii) is false
(C) Statement (i) is false.statement (ii) is true
(D) Both statements are false

PART – B (5 × 10 = 50 Marks)
Answer **ALL** Questions

		Marks	BL	CO	PO
26. a.i.	What is perceptron convergence theorem? Explain the steps involved in perceptron learning algorithm.	5	2	1	1
ii.	Find the derivatives of the activation function of 1) Sigmoid 2) TanH 3) RcLU 4) Leaky RcLU	5	3	1	2
	(OR)				
b.	Consider inputs = [2, 3] and output = [1]. Let the initial weights be w_1=0.11, w_2=0.21, w_3=0.12, w_4=0.08, w_5=0.14, w_6=0.15. Compute the updated weights and predicted value after the first round of backpropagation? Explain the forward pass and backward propagation steps neatly.	10	3	1	3
27. a.i.	You have to solve a classification task. You first train your network of samples, training converger, but the training loss is very high. You then decide to train this network on 10, 000 examples. (This is more number of samples than what was considered earlier). Is your approach to fixing the problem correct? If yes, explain the most likely result of training with 10, 000 examples. If not, give a solution to their problem.	5	4	2	3
ii.	Explain how the following technique help in regularization in neural network? 1) Dropout 2) Data augmentation 3) Early stopping	5	2	2	1
	(OR)				
b.i.	Name any four optimization techniques used in deep learning. Clearly mention the technique behind each approach and its advantages/ disadvantages over other techniques.	5	2	2	1
ii.	Explain how to use denoising auto encoders for image denoising. Explain with algorithmic steps and mathematical representation of last functions.	5	3	2	2
28.a.i.	You come up with a CNN classifier. For each layer, calculate the number of weights, number of biases and the size of the associated feature maps. • CONV-K-N denotes a convolutional layer with N filters, each of size K×K, padding = 0, stride = 1. • POOL-K indicates K×K pooling layer with stride k and padding 0. • FC-N stands for fully connected layer with N neurons.	10	4	3	3

Layer	Activation map dimension	Number of weights	No of biases
Input	128×128×3	0	0
Conv-9-32			
Pool-2			
Conv-5-64			
Pool-2			
Conv-5-64			
Pool-2			
FC-3			

(OR)

b.i. Write the formula for calculating the outputs size in a convolution layer? Consider a 2A convolution layer that takes a 3×128×128 input and has 40 filters of size 5×5, what is the size of the output layer? 5 3 3 2

Case a) stride = 1
Case b) stride = 2
Case c) padding = 0
Case d) padding = 1

ii. You are solving the binary classification task of classifying images as cat Vs non cat. You design a CNN with a single output neuron. Let the output of this neuron be Z. the final output of your network $\hat{y} = r(\mathrm{Re}LU(z))$. You classify all inputs with a final value $\hat{y} \geq 0.5$ as cat images. What is the problem you are going to encounter? 5 4 3 3

29. a.i. You have a dataset D_1 with 1 million labelled training examples for classification and dataset D_2 with 1000 labelled training examples. Your friend trains a model from scratch on dataset D_2. You decide to train on D_1, and then apply transfer learning to train on D_2. State one problem your friend is likely to find with his approach. How does your approach address their problem? 5 4 4 3

ii. What do you mean by transfer learning in neural networks? How it is helpful in improving the learning? 5 2 4 1

(OR)

b.i. What are the problems encountered in a simple recurrent neural network? Mention some techniques to overcome the same. 5 2 4 1

ii. What is the difference between Vanilla RNNS and LSTMS? Explain the architecture of an LSTM network with a neat diagram and mathematical formulations. 5 2 4 1

30. a.i. Explain generative adversarial networks with a neat diagram. Also, mention any three applications of GAN? 5 2 5 1

ii. How do you train a GAN? Explain with proper algorithm and mathematical formulation. 5 2 5 1

(OR)

b.i. Design an image captioning model with visual attention. Write the steps involved in it. 5 2 5 3

ii. Visual Question Answering (VQA) is an algorithm that takes as input an image and a natural language question about the image and generates a natural language answer as the output. Design the proposed VQA model. 5 3 5 3

* * * * *

SRM Institute of Science and Technology

College of Engineering and Technology, School of Computing

DEPARTMENT OF COMPUTING TECHNOLOGIES

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamilnadu

Academic Year: AY-2022-23 (ODD)

Test: University theory, Date: 13/12/22

Course Code & Title: 18AIC301J, Deep Learning Techniques

Year & Sem: 3rd year, Vth sem

Duration: 150 mins, Max marks:75

Answer key of Part B

(Apply and analyse) -Descriptive (Either or)- 5X10 marks=50 marks (110 min)

26 (a) (i) What is perceptron convergence theorem? Explain the steps involved in perceptron learning algorithm ?

[Marks split: Theorem-2 marks; Steps-3 marks. Total=5 marks]

Perceptron Convergence Theorem: For any finite set of linearly separable labeled examples, the Perceptron Learning Algorithm will halt after a finite number of iterations. In other words, after a finite number of iterations, the algorithm yields a vector w that classifies perfectly all the examples.

Steps involved in perceptron learning algorithm:

1. Feed the features of the model that is required to be trained as input in the first layer.
2. All weights and inputs will be multiplied – the multiplied result of each weight and input will be added up
3. The Bias value will be added to shift the output function
4. This value will be presented to the activation function (the type of activation function will depend on the need)
5. The value received after the last step is the output value.

26 (a)(ii) Find the derivatives of the activation functions of (i) Sigmoid, (ii) tanh, (iii) ReLU and (iv) Leaky ReLU

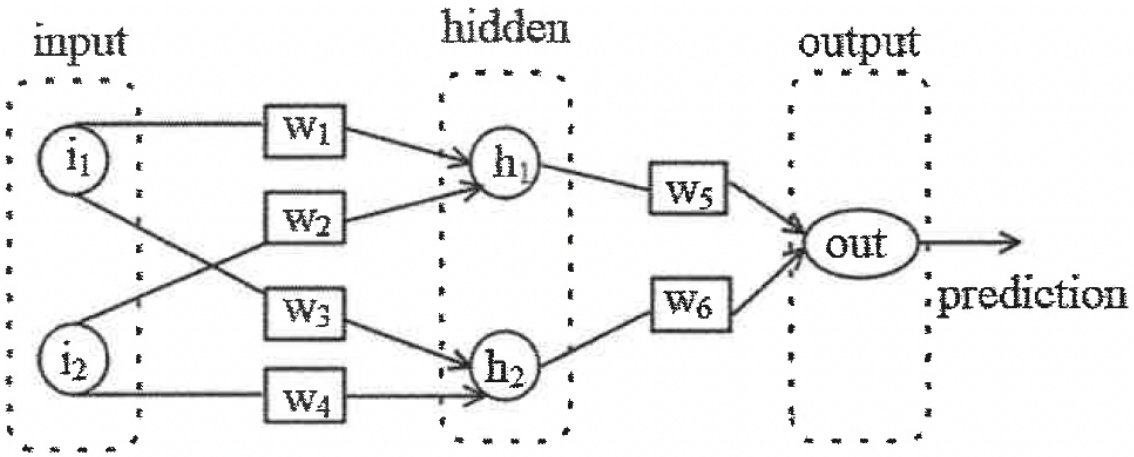
[Marks split: Activation functions and its derivarives 4*1.25 marks; Total=5 marks]

Function Type	Equation	Derivative
Linear	$f(x) = ax + c$	$f'(x) = a$
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)\,(1-f(x))$
TanH	$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ReLU	$f(x) = \begin{cases} 0 \text{ for } x<0 \\ x \text{ for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 \text{ for } x<0 \\ 1 \text{ for } x \geq 0 \end{cases}$
Parametric ReLU	$f(x) = \begin{cases} \alpha x \text{ for } x<0 \\ x \text{ for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha \text{ for } x<0 \\ 1 \text{ for } x \geq 0 \end{cases}$
ELU	$f(x) = \begin{cases} \alpha(e^x - 1) \text{ for } x<0 \\ x \text{ for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha \text{ for } x<0 \\ 1 \text{ for } x \geq 0 \end{cases}$

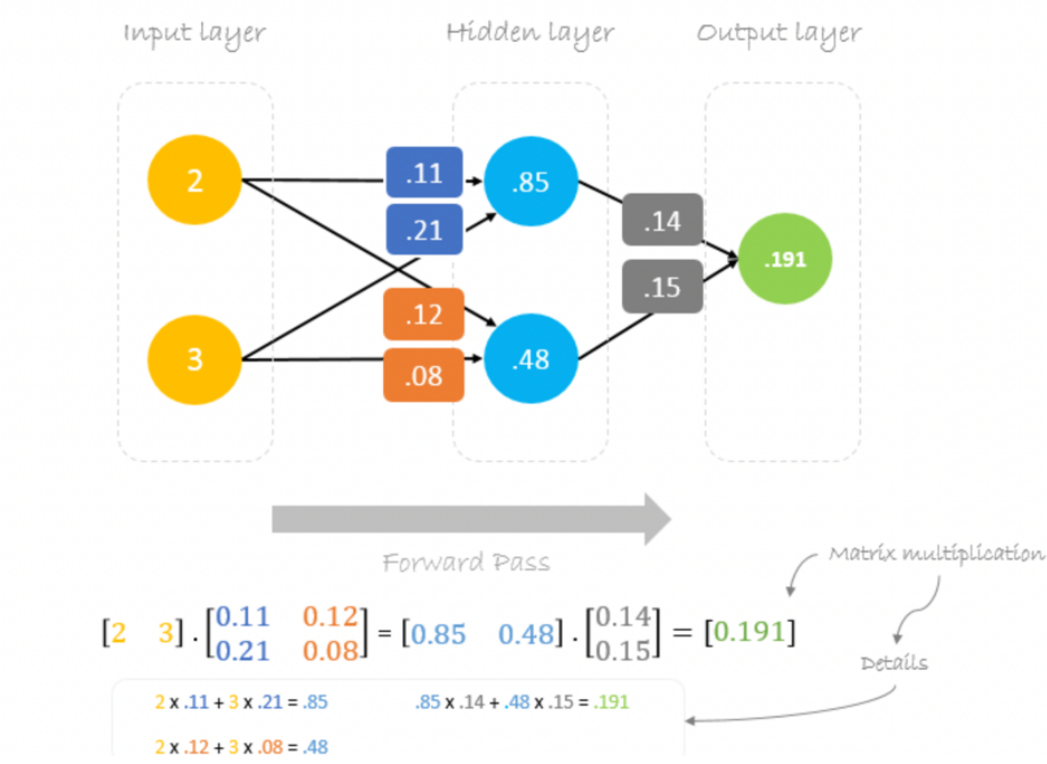
---OR---

26 (b) Consider inputs=[2, 3] and output=[1]. Let the initial weights be: w1 = 0.11, w2 = 0.21, w3 = 0.12, w4 = 0.08, w5 = 0.14 and w6 = 0.15. Compute the updated weights and new predicted value after the first round of backpropagation. Explain the forward pass and backpropagation steps neatly.

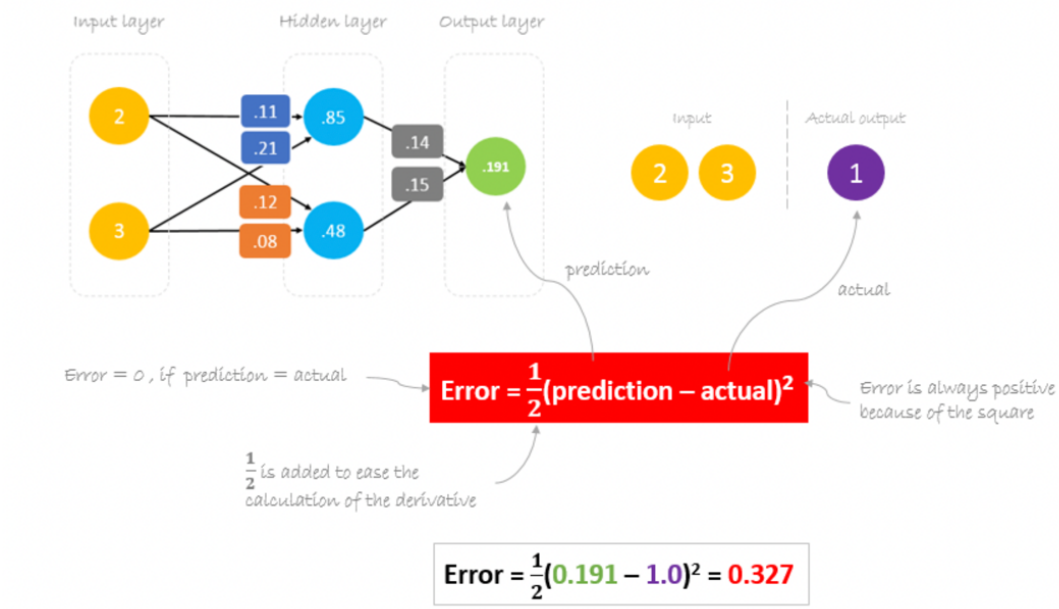
[Marks split: Mathematical formulation and weight computation-5 marks; forward pass and backpropagation steps-5 marks. Total=10 marks]



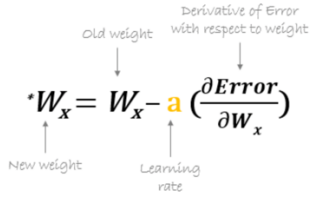
Forward pass:



Error computation:

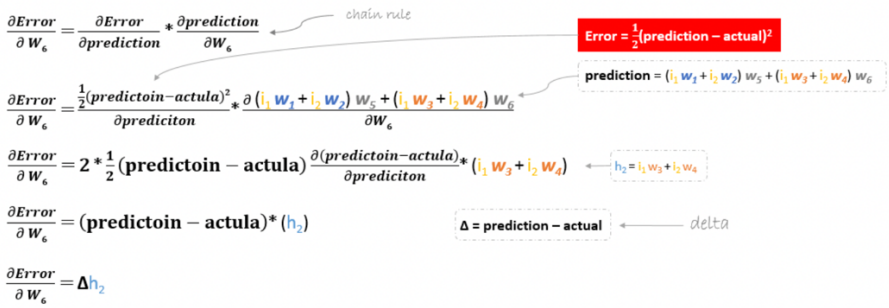


Reducing Error via backpropagation:



$$^{*}W_6 = W_6 - a\left(\frac{\partial Error}{\partial W_6}\right)$$

The derivation of the error function is evaluated by applying the chain rule as following



$$^{*}W_6 = W_6 - a\,\Delta h_2$$

$$^{*}W_5 = W_5 - a\,\Delta h_1$$

$$\frac{\partial Error}{\partial W_1} = \frac{\partial Error}{\partial prediction} * \frac{\partial prediction}{\partial h_1} * \frac{\partial h_1}{\partial W_1}$$ ← chain rule

Error = $\frac{1}{2}$(prediction – actual)^2

prediction = $(h_1)\,w_5 + (h_2)\,w_6$

$h_1 = i_1 w_1 + i_2 w_2$

$$\frac{\partial Error}{\partial W_1} = \frac{\partial \frac{1}{2}(predictoin - actula)^2}{\partial prediciton} * \frac{\partial\,(h_1)\,w_5 + (h_2)\,w_6}{\partial h_1} * \frac{\partial\, i_1 w_1 + i_2 w_2}{\partial w_1}$$

$$\frac{\partial Error}{\partial W_1} = 2 * \frac{1}{2}(\mathbf{predictoin - actula}) \frac{\partial (predictoin - actula)}{\partial prediciton} * (w_5) * (i_1)$$

$$\frac{\partial Error}{\partial W_1} = (\mathbf{predictoin - actula}) * (w_5 i_1)$$

Δ = prediction – actual ← delta

$$\frac{\partial Error}{\partial W_1} = \Delta\, w_5 i_1$$

Updated weights

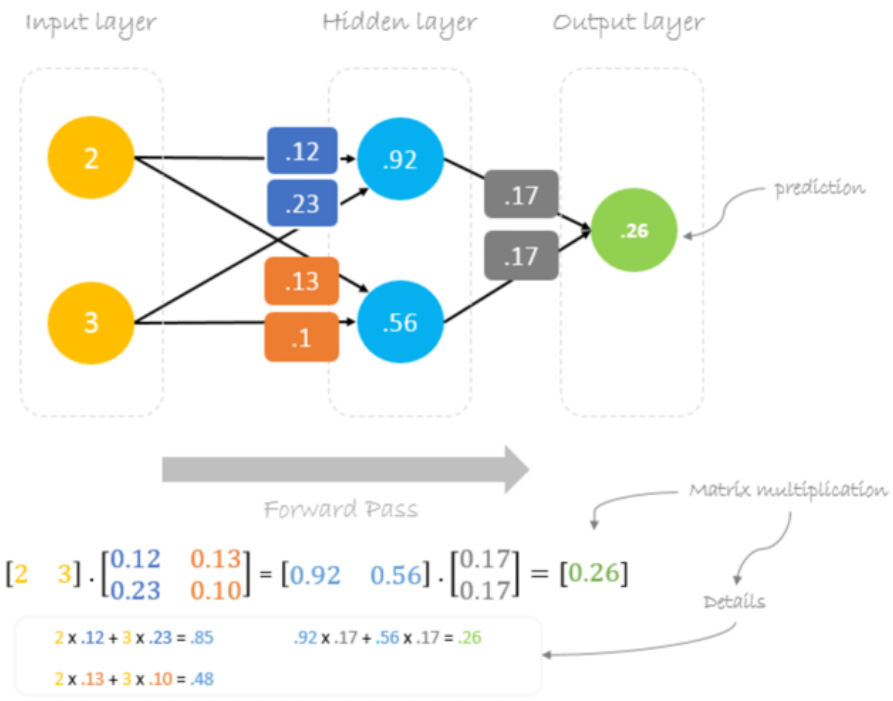
$$^{*}w_6 = w_6 - a\,(h_2 \,.\, \Delta)$$
$$^{*}w_5 = w_5 - a\,(h_1 \,.\, \Delta)$$
$$^{*}w_4 = w_4 - a\,(i_2 \,.\, \Delta w_6)$$
$$^{*}w_3 = w_3 - a\,(i_1 \,.\, \Delta w_6)$$
$$^{*}w_2 = w_2 - a\,(i_2 \,.\, \Delta w_5)$$
$$^{*}w_1 = w_1 - a\,(i_1 \,.\, \Delta w_5)$$

Δ = 0.191 – 1 = -0.809 ← **Delta** = prediction - actual

a = 0.05 ← **Learning rate**, we smartly guess this number

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - 0.05(-0.809) \begin{bmatrix} 0.85 \\ 0.48 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - \begin{bmatrix} -0.034 \\ -0.019 \end{bmatrix} = \begin{bmatrix} 0.17 \\ 0.17 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - 0.05(-0.809) \begin{bmatrix} 2 \\ 3 \end{bmatrix} . [0.14 \quad 0.15] = \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - \begin{bmatrix} -0.011 & -0.012 \\ -0.017 & -0.018 \end{bmatrix} = \begin{bmatrix} .12 & .13 \\ .23 & .10 \end{bmatrix}$$



---

27. a.i. You have to solve a classification task. You first train your network of samples, training converger, but the training loss is very high. You then decide to train this network on 10, 000 examples. (This is more number of samples than what was considered earlier). Is your approach to fixing the problem correct? If yes, explain the most likely result of training with 10, 000 examples. If not, give a solution to their problem.

[Marks split: Problem identification-2 marks; Solution-3 marks. Total=5 marks]

Answer key:

The model is suffering from bias problem. Increasing the amount of data reduces the variance, and is not likely to solve the problem. A better approach would be to decrease the bias of the model by adding more layers/ learnable parameters. It is possible that training converged to a local optimum. Training longer/ using a better optimizer/ restarting from a different initialization also could work.

ii. Explain how the following technique help in regularization in neural network?

1) Dropout
2) Data augmentation
3) Early stopping

[Marks split: Regularization -0.5 marks; Each technique-1.5*3=4.5 marks. Total=5 marks]

Dropout: In machine learning, "dropout" refers to the practice of disregarding certain nodes in a layer at random during training. Dropout is a regularization technique for reducing overfitting in neural networks by preventing complex co-adaptations on training data. The term "dropout" refers to dropping out units (both hidden and visible) in a neural network. It is a very efficient way of performing model averaging with neural networks. A dropout is a regularization approach that prevents overfitting by ensuring that no units are codependent with one another.

Data augmentation: deep learning models are data hungry, if we are lacking data then by using data augmentation transformations of the image we can generate data. Data augmentation is a preprocessing technique because we only work on the data to train our model. In this technique, we generate new instances of images by cropping, flipping, zooming, shearing an original image. So, whenever the training lacks the image dataset, using augmentation, we can create thousands of images to train the model perfectly.

Early stopping: In machine learning, early stopping is a form of regularization used to avoid overfitting when training a learner with an iterative method, such as gradient descent. Such methods update the learner so as to make it better fit the training data with each iteration.In early stopping, the algorithm is trained using the training set, and the point at which to stop training is determined from the validation set. Training error and validation error are analyzed. The training error steadily decreases while the validation error decreases until a point, after which it increases. So a model with better validation set error can be obtained if the parameters that give the least validation set error are used.

(OR)

b.i. Name any four optimization techniques used in deep learning. Clearly mention the technique behind each approach and its advantages/ disadvantages over other techniques.

[Marks split: 4 techniques and its pros and cons: 4*1.25=5; Total=5 marks]

1) Stochastic Gradient Descent

It's a variant of Gradient Descent. It tries to update the model's parameters more frequently. In this, the model parameters are altered after computation of loss on each training example. So, if the dataset contains 1000 rows SGD will update the model parameters 1000 times in one cycle of dataset instead of one time as in Gradient Descent.

$\theta=\theta-\alpha\cdot\nabla J(\theta;x(i);y(i))$, where {x(i) ,y(i)} are the training examples.
As the model parameters are frequently updated parameters have high variance and fluctuations in loss functions at different intensities.

Advantages:

1. Frequent updates of model parameters hence, converges in less time.
2. Requires less memory as no need to store values of loss functions.
3. May get new minima's.

Disadvantages:

1. High variance in model parameters.
2. May shoot even after achieving global minima.
3. To get the same convergence as gradient descent needs to slowly reduce the value of learning rate.

---

2) Mini-Batch Gradient Descent

It's best among all the variations of gradient descent algorithms. It is an improvement on both SGD and standard gradient descent. It updates the model parameters after every batch. So, the dataset is divided into various batches and after every batch, the parameters are updated.
$\theta=\theta-\alpha\cdot\nabla J(\theta;$ B(i)), where {B(i)} are the batches of training examples.

Advantages:

1. Frequently updates the model parameters and also has less variance.
2. Requires medium amount of memory.

All types of Gradient Descent have some challenges:

1. Choosing an optimum value of the learning rate. If the learning rate is too small than gradient descent may take ages to converge.
2. Have a constant learning rate for all the parameters. There may be some parameters which we may not want to change at the same rate.
3. May get trapped at local minima.

---

3) Momentum

Momentum was invented for reducing high variance in SGD and softens the convergence. It accelerates the convergence towards the relevant direction and reduces the fluctuation to the irrelevant direction. One more hyperparameter is used in this method known as momentum symbolized by 'γ'.

$V(t)=\gamma V(t-1)+\alpha.\nabla J(\theta)$
Now, the weights are updated by **$\theta=\theta-V(t)$.**
The momentum term γ is usually set to 0.9 or a similar value.

Advantages:
Reduces the oscillations and high variance of the parameters.
Converges faster than gradient descent.

Disadvantages:
One more hyper-parameter is added which needs to be selected manually and accurately.

4) Nesterov Accelerated Gradient

Momentum may be a good method but if the momentum is too high the algorithm may miss the local minima and may continue to rise up. So, to resolve this issue the NAG algorithm was developed. It is a look ahead method. We know we'll be using **γV(t−1)** for modifying the weights so, **θ−γV(t−1)** approximately tells us the future location. Now, we'll calculate the cost based on this future parameter rather than the current one.
V(t)=γV(t−1)+α. ∇J(θ−γV(t−1)) and then update the parameters using **θ=θ−V(t).**

NAG vs momentum at local minima
Advantages:
Does not miss the local minima.
Slows if minima's are occurring.
Disadvantages:
Still, the hyperparameter needs to be selected manually.

5) Adagrad

One of the disadvantages of all the optimizers explained is that the learning rate is constant for all parameters and for each cycle. This optimizer changes the learning rate. It changes the learning rate **'η'** for each parameter and at every time step **'t'.** It's a type second order optimization algorithm. It works on the derivative of an error function.

$$g_{t,i} = \nabla_\theta J(\theta_{t,i}),$$

A derivative of loss function for given parameters at a given time t.

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \cdot g_{t,i}.$$

Update parameters for given input i and at time/iteration t
η is a learning rate which is modified for given parameter **θ(i)** at a given time based on previous gradients calculated for given parameter **θ(i).**
We store the sum of the squares of the gradients w.r.t. **θ(i)** up to time step **t**, while **ϵ** is a smoothing term that avoids division by zero (usually on the order of 1e−8). Interestingly, without the square root operation, the algorithm performs much worse.
It makes big updates for less frequent parameters and a small step for frequent parameters.
Advantages:
Learning rate changes for each training parameter.
Don't need to manually tune the learning rate.
Able to train on sparse data.
Disadvantages:
Computationally expensive as a need to calculate the second order derivative.
The learning rate is always decreasing results in slow training.

6) Adam

Adam (Adaptive Moment Estimation) works with momentums of first and second order. The intuition behind the Adam is that we don't want to roll so fast just because we can jump over the minimum, we want to decrease the velocity a little bit for a careful search. In addition to storing an exponentially decaying average of past squared gradients
like **AdaDelta**, *Adam* also keeps an exponentially decaying average of past gradients **M(t).**

M(t) and V(t) are values of the first moment which is the *Mean* and the second moment which is the *uncentered variance* of the gradientsrespectively.

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}.$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}.$$

First and second order of momentum
Here, we are taking mean of **M(t)** and **V(t)** so that **E[m(t)]** can be equal to **E[g(t)]** where, **E[f(x)]** is an expected value of **f(x)**.
To update the parameter:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t.$$

Update the parameters
The values for β1 is 0.9 , 0.999 for β2, and (10 x exp(-8)) for '**ϵ**'.
Advantages:
The method is too fast and converges rapidly.
Rectifies vanishing learning rate, high variance.
Disadvantages:
Computationally costly.

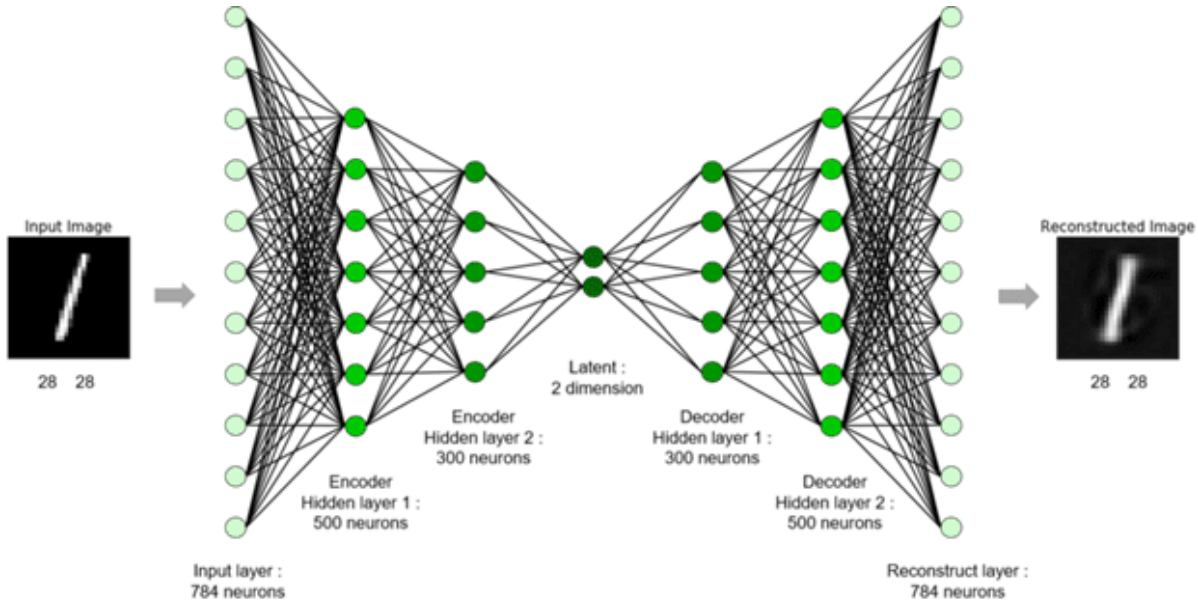
ii. Explain how to use denoising auto encoders for image denoising. Explain with algorithmic steps and mathematical representation of last functions.

[Marks split: Denoising AE-2 marks; Algorithmic steps and mathematical representation-3 marks. Total=5 marks]

An autoencoder contains an encoder and decoder. These two parts function automatically and give rise to the name "autoencoder". Encoder transforms high-dimensional input into lower-dimension (latent state, where the input is more compressed), while a decoder does the reverse encoder job on the encoded outcome and reconstructs the original image.

In denoising, data is corrupted in some manner through the addition of random noise, and the model is trained to predict the original uncorrupted data. Another variation of this is about omitting parts of the input in contrast to adding noise to input so that model can learn to predict the original image. In this case, the idea is storing the output generated by the encoder as a feature vector, which can be used in a supervised model train-prediction approach.

Denoising autoencoders application is very versatile and can be focused on cleaning old stained scanned images or contribute to feature selection efforts in cancer biology. Regarding, old images encoder compression contributes to an output, which helps the model reconstructing the actual image using robust latent representations by the decoder. Regarding cancer biology, the extracted encoder features contribute to the efforts toward the improvement of a cancer diagnosis.



The idea of denoising is based on the intentional addition of noise to the input data before the presentation of data. The major technical specifics for this approach include several aspects as follows.

The denoising autoencoders build corrupted copies of the input images by adding random noise.Next, denoising autoencoders attempt to remove the noise from the noisy input and reconstruct the output that is like the original input. A comparison is made between the original image, and the model prediction using a loss function and the goal is to minimize that loss.

The loss function in denoising autoencoder is

$$L = | \tilde{x} - g(f(x)) |$$

Denoising helps the autoencoder learn the latent representation in data and makes a robust representation of useful data possible hence supporting the recovery of the clean original input.

28.a.i. You come up with a CNN classifier. For each layer, calculate the number of weights, number of biases and the size of the associated feature maps.

- CONV-K-N denotes a convolutional layer with N filters, each of size K×K, padding = 0, stride = 1.
- POOL-K indicates K×K pooling layer with stride k and padding 0.
- FC-N stands for fully connected layer with N neurons.

Layer	Activation map dimension	Number of weights	No of biases
Input	128×128×3	0	0
Conv-9-32			
Pool-2			
Conv-5-64			
Pool-2			
Conv-5-64			
Pool-2			
FC-3			

[Marks split:Feature map size computation formula=2 mark, Computation of weights for each layer-1 mark, so total= 8*1=8 marks. Total=10 marks]

The formula for calculating the output size: $[(W-K+2P)/S]+1$.

- W is the input volume
- K is the Kernel size
- P is the padding
- S is the stride

Solution: Successively:

- $120 \times 120 \times 32$ and $32 \times (9 \times 9 \times 3 + 1)$
- $60 \times 60 \times 32$ and 0
- $56 \times 56 \times 64$ and $64 \times (5 \times 5 \times 32 + 1)$
- $28 \times 28 \times 64$ and 0
- $24 \times 24 \times 64$ and $64 \times (5 \times 5 \times 64 + 1)$
- $12 \times 12 \times 64$ and 0
- 3 and $3 \times (12 \times 12 \times 64 + 1)$

(OR)

b.i. Write the formula for calculating the outputs size in a convolution layer? Consider a 2A convolution layer that takes a 3×128×128 input and has 40 filters of size 5×5, what is the size of the output layer?

Case a) stride = 1
Case b) stride = 2
Case c) padding = 0
Case d) padding = 1

[Marks split:Feature map size computation formula=1 mark, Computation for each case-1*4=4 marks. Total=5 marks]

The formula for calculating the output size: [(W−K+2P)/S]+1.

- W is the input volume - in your case 128
- K is the Kernel size - in your case 5
- P is the padding - in your case 0 i believe
- S is the stride - which you have not provided.

So, we input into the formula:

Case1) : [(W−K+2P)/S]+1.= (128-5+0)/1+1=124. Output_Shape = (124,124,40)

Case2) : [(W−K+2P)/S]+1.= (128-5+0)/2+1=62 Output_Shape = (62,62,40)

Case3) : [(W−K+2P)/S]+1.= (128-5+0)/1+1=124. Output_Shape = (124,124,40)

Case4) : [(W−K+2P)/S]+1.= (128-5+2)/1+1=126 Output_Shape = (126,126,40)

ii. You are solving the binary classification task of classifying images as cat Vs non cat. You design a CNN with a single output neuron. Let the output of this neuron be Z. the final output of your network $\hat{y} = r(\mathrm{Re}LU(z))$. You classify all inputs with a final value $\hat{y} \geq 0.5$ as cat images. What is the problem you are going to encounter?

[Marks split:Problem scenario identification =3 mark, output prediction=2 marks.

Total=5 marks]

Using ReLU then sigmoid will cause all predictions to be positive.

$$(\sigma(ReLU(z)) \geq 0.5 \quad \forall z).$$

---

29. a.i. You have a dataset D_1 with 1 million labelled training examples for classification and dataset D_2 with 1000 labelled training examples. Your friend trains a model from scratch on dataset D_2. You decide to train on D_1, and then apply transfer learning to train on D_2. State one problem your friend is likely to find with his approach. How does your approach address their problem?

[Marks split:Problem scenario identification =3 mark, approach explanation=2 marks.

Total=5 marks]

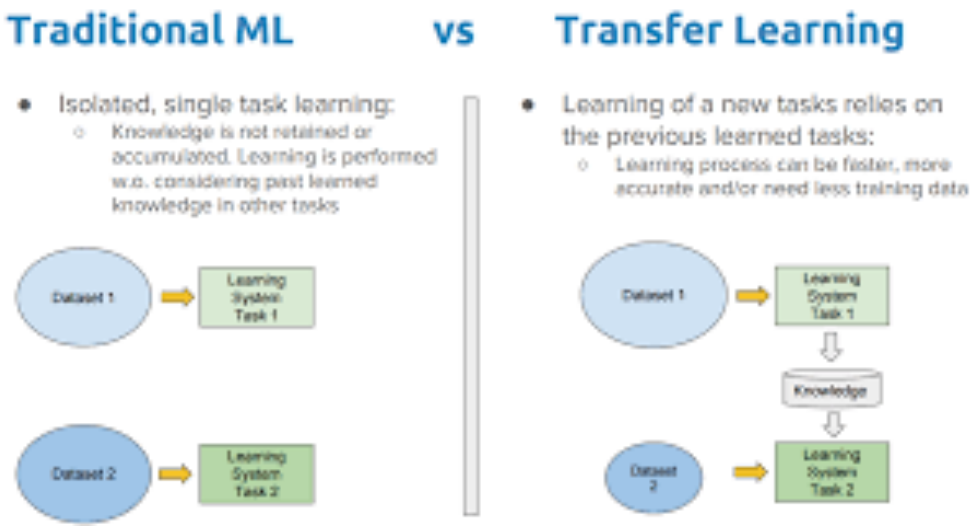
Friend is likely to see overfitting. Model is not going to generalize well to unseen data. By using transfer learningand freezing the weights in the earlier layers, you reduce the no of learnable parameters, while using the weights which have been pretrained on a much larger dataset.

ii. What do you mean by transfer learning in neural networks? How it is helpful in improving the learning?

[Marks split:Transfer learning =3 mark, helpful in learning=2 marks.

Total=5 marks]

The training of neural networks takes a huge amount of resources because of the complexity of the models. **Transfer learning** is used to make the process more efficient and lower the resource demand. Any transferable knowledge or features can be moved between networks to streamline the development of new models. Transfer learning means that training won't need to be restarted from scratch for every new task. Training new machine learning models can be resource-intensive, so transfer learning saves both resources and time. With transfer learning, a model can be trained on an available labelled dataset, then be applied to a similar task that may involve unlabelled data.



(OR)

b.i. What are the problems encountered in a simple recurrent neural network? Mention some techniques to overcome the same.

[Marks split:Problems identification =3 mark, Techniques to overcome =2 marks.

Total=5 marks]

There are two widely known issues with properly training Recurrent Neural Networks, the vanishing and the exploding gradient problems . The influence of a given input on the hidden layer, and therefore on the network output, either decays or blows up exponentially as it cycles around the network's recurrent connections. Schmidhuber, 1997). RNNs suffer from the problem of **vanishing gradients.** The gradients carry information used in the RNN, and when the gradient becomes too small, the parameter updates become insignificant. This makes the learning of long data sequences difficult. The vanishing gradient problem is caused by the derivative of the activation function used to create the neural network. The simplest solution to the problem is to replace the activation function of the network. Instead of sigmoid, use an activation function such as ReLU.

Exploding gradients are a problem when large error gradients accumulate and result in very large updates to neural network model weights during training. Gradients are used during training to update the network weights, but when the typically this process works best when these updates are small and controlled. In general, exploding gradients can be avoided by carefully configuring the network model, such as using **a small learning rate, scaling the target variables, and using a standard loss function.** However, in recurrent networks with a large number of input time steps, exploding gradients may still be an issue. Another popular technique to mitigate the exploding gradients problem is to clip the gradients during backpropagation so that they never exceed some threshold. This is called **Gradient Clipping**.

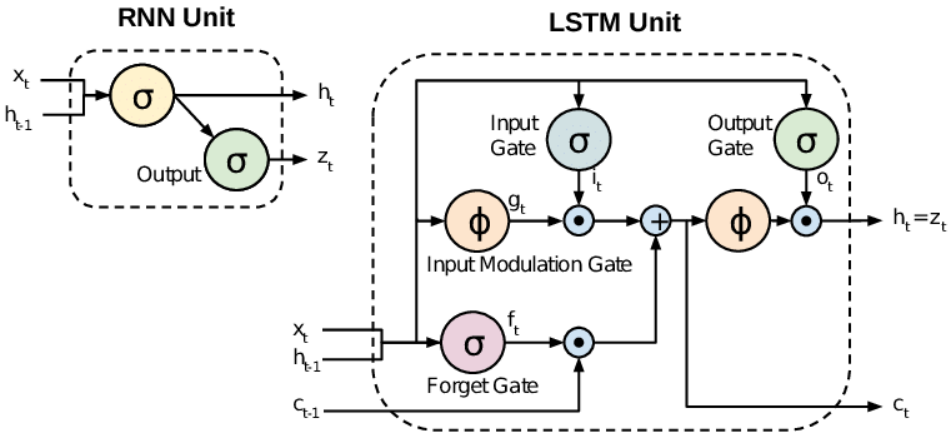
The most effective solution so far is the Long Short Term Memory (LSTM) architecture.

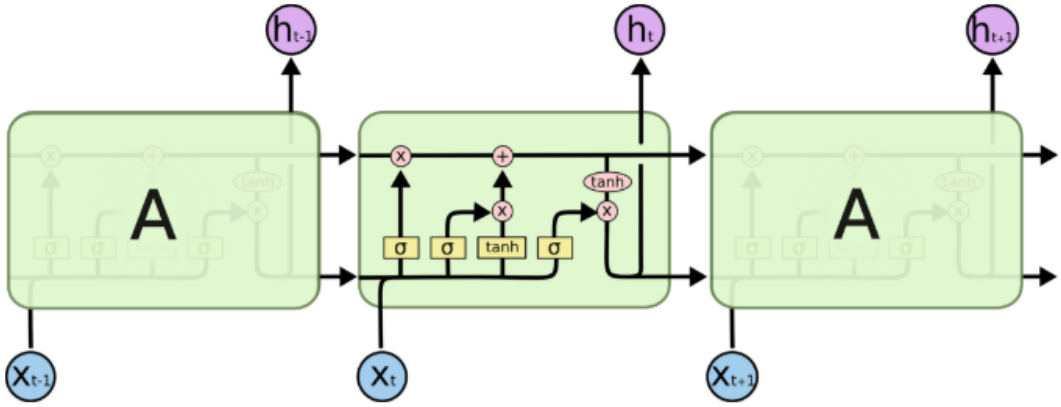
ii. What is the difference between Vanilla RNNS and LSTMS? Explain the architecture of an LSTM network with a neat diagram and mathematical formulations.

[Marks split:Difference =2 mark, LSTM detailing=3 marks.

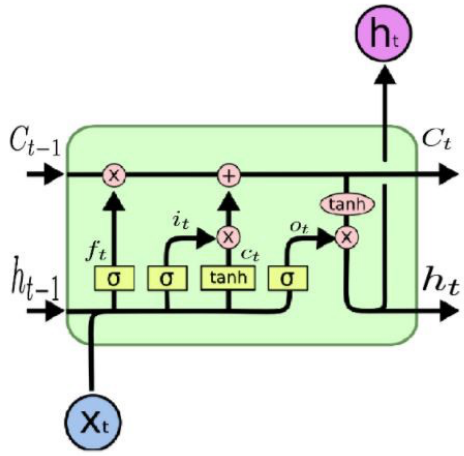
Total=5 marks]

Long short-term memory (LSTM) networks are an extension of RNN that extend the memory. LSTM are used as the building blocks for the layers of a RNN. LSTMs assign data “weights” which helps RNNs to either let new information in, forget information or give it importance enough to impact the output. Vanilla RNNs do not have a cell state. They only have hidden states and those hidden states serve as the memory for RNNs. Meanwhile, LSTM has both cell states and a hidden states. The cell state has the ability to remove or add information to the cell, regulated by "gates".



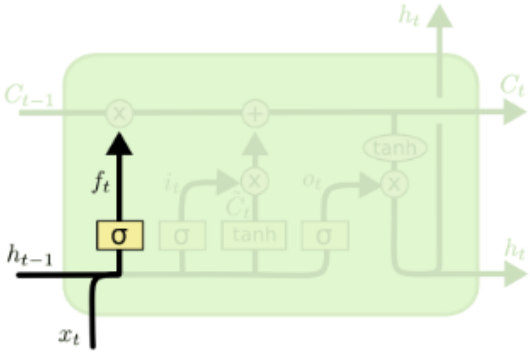


The repeating module in an LSTM contains four interacting layers.

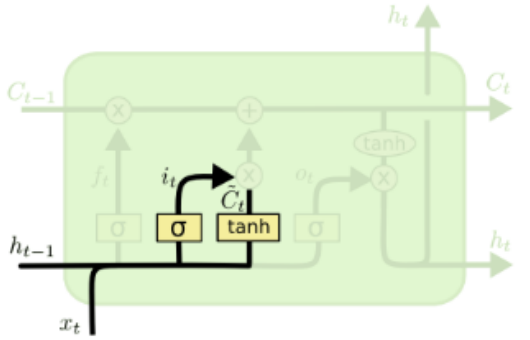


f: Forget gate, Whether to erase cell
i: Input gate, whether to write to cell
g: Gate gate (?), How much to write to cell
o: Output gate, How much to reveal cell

Mathematical formulations:

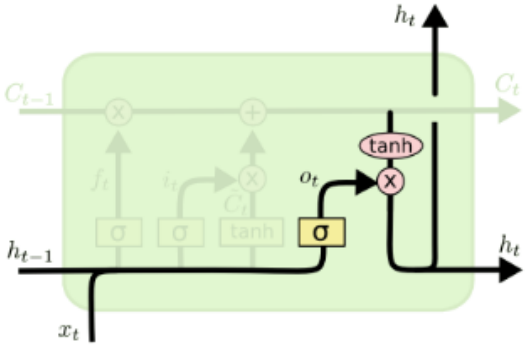


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

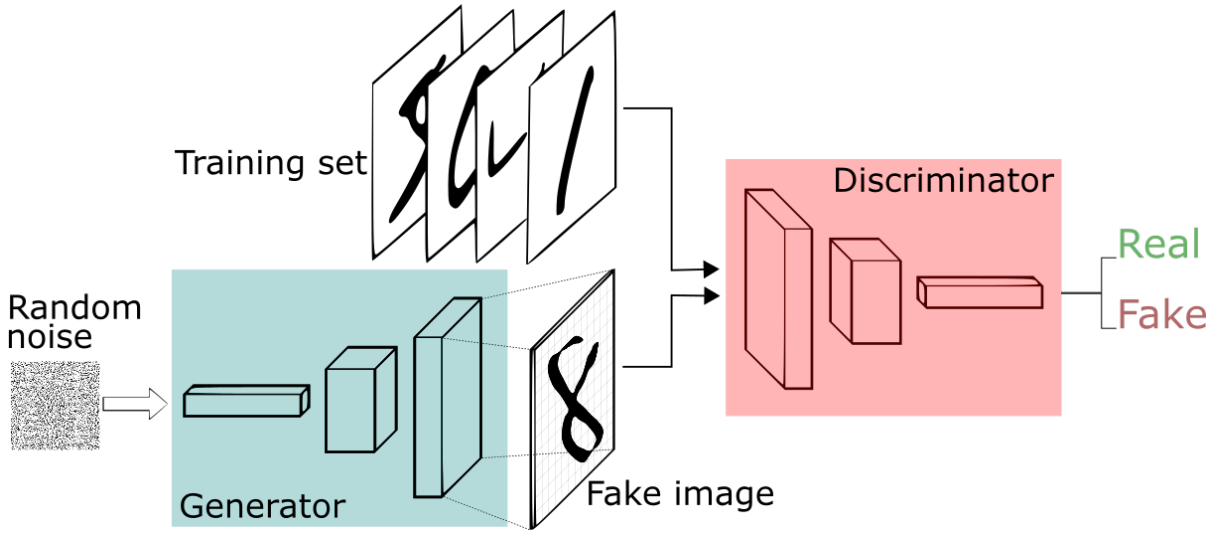
---

30. a.i. Explain generative adversarial networks with a neat diagram. Also, mention any three applications of GAN?

ii. How do you train a GAN? Explain with proper algorithm and mathematical formulation.

[Marks split:GAN diagram =3 mark, applications=2 marks. Total=5 marks]

[Marks split:GAN training formulations =3 marks. Total=5 marks]



Generative Adversarial Networks, or GANs for short, are an approach to generative modeling using deep learning methods, such as convolutional neural networks.Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output new examples that plausibly could have been drawn from the original dataset. GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to classify examples as either real (from the domain) or fake (generated). The two models are trained together in a zero-sum game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples.

Three applications:

- Generate examples for Image Datasets.
- Image-to-Image Translation.
- Text-to-Image Translation.
- Semantic-Image-to-Photo Translation.
- Face Frontal View Generation.
- Generate New Human Poses.
- Photos to Emojis.
- Photograph Editing.

GAN Training formulations:

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

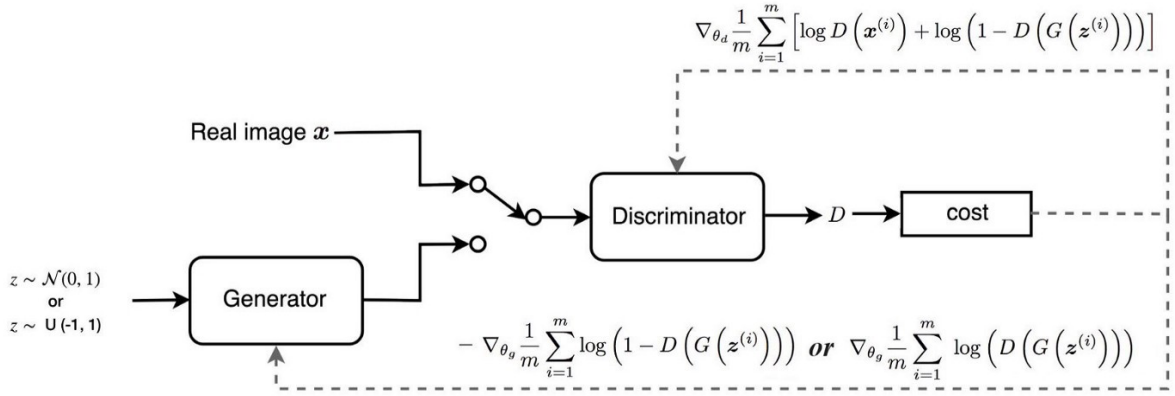
end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.



(OR)

b.i. Design an image captioning model with visual attention. Write the steps involved in it.

ii. Visual Question Answering (VQA) is an algorithm that takes as input an image and a natural language question about the image and generates a natural language answer as the output. Design the proposed VQA model.

* * * * *

[Marks split: Model =2 mark, Steps=3 marks. Total=5 marks]

[Marks split: VQA Model =5 mark. Total=5 marks]

Image captioning model with visual attention:
The encoder-decoder image captioning system would encode the image, using a pre-trained Convolutional Neural Network that would produce a hidden state. Then, it would decode this hidden state by using an LSTM and generate a caption.For each sequence element, outputs from previous elements are used as inputs, in combination with new sequence data. This gives the RNN networks a sort of memory which might make captions more informative and contextaware. But RNNs tend to be computationally expensive to train and evaluate, so in practice, memory is limited to just a few elements. Attention models can help address this problem by selecting the most relevant elements from an input image.

Steps involved: With an Attention mechanism, the image is first divided into n parts, and we compute an image representation of each When the RNN is generating a new word, the attention mechanism is focusing on the relevant part of the image, so the decoder only uses specific parts of the image.

Visual question answering: