Ramakrishna Mission Vivekananda Educational & Research Institute
Belur Math, Howrah, West Bengal
School of Mathematical Sciences, Department of Computer Science

End Semester Examination - **CS411: Applications of Computer Vision and Deep Learning**
M.Sc. Computer Science and Big Data Analytics   Date: 24-May-2025
Time: 11:00 AM to 01:00 PM
Instructor: Jimut Bahan Pal   Max marks: 100

EXAMINATION INSTRUCTIONS

IMPORTANT: Answer all questions completely.

This is a closed-book examination. No reference materials are permitted except for **calculators**.

Time Requirements:

- Minimum time in examination hall: 1 hour
- Maximum examination duration: 3 hours
- Restroom breaks are permitted

Response Guidelines:

- Provide concise, focused answers to all subjective questions
- If clarification is needed, make appropriate assumptions and continue

Prohibited Activities:

- Discussion with other examinees
- Use of unauthorized materials

Submission Process:

- You may make a photocopy of your completed answer script
- Submit the PDF copy by replying to the final instructions email

Remain calm and good luck!

Subjective Questions

1. **Kullback-Leibler Divergence** (Q1. Total = **8**)

   (a) Define the Kullback-Leibler (KL) divergence for two discrete probability distributions P and Q. (2)

   (b) State the three mathematical conditions required for a function to qualify as a distance metric. (3)

   (c) Determine whether KL-divergence satisfies each of these conditions. Justify your conclusions with specific examples or counterexamples. (2)

   (d) Explain why KL-divergence is still widely used in machine learning and statistics despite not being a true distance metric. (1)

2. **From the figure below** (Q2. Total = **20**)

   (a) Derive the change in weight, i.e., Δw_{11}'s equation. Please absorb any constant inside the learning rate. Derive to find $\Delta w_{11} = \eta(t_1 - o_1)o_1(1 - o_1)x_1$. (15)

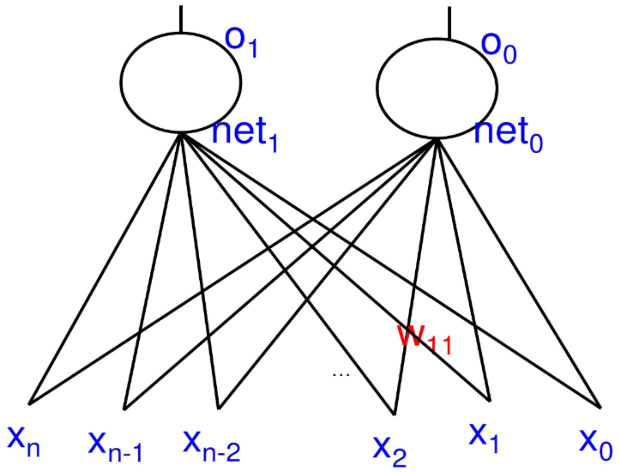

Figure 1: Figure of a neural network having sigmoid as activation.

   Use the same set of procedures followed in the class to produce the final results. Here, the target vector is $<t_1, t_0>$and the observed vector is $<o_1, o_0>$. Use the total sum of squared loss, i.e, TSS = $\frac{1}{2}[(t_1 - o_1)^2 + (t_2 - o_2)^2]$

   (b) What are all the possibilities of vanishing gradient for this problem. (5)

3. **Answer the following questions by referring to the CNN model shown in the next page**: (Q3. Total = **18**)

   1. Explain what this model is doing, i.e., what can this model be used for? Give at least two specific examples of applications. (3)
   2. What is the expected input format of the model? Is it channel-first or channel-last? Justify your answer with evidence from the code. (3)
   3. For an input image of size (3, 160, 160): (3+3+3+3)

      a. Trace the input through each layer of the network

      b. Calculate and present the dimensions of the feature maps after each layer

c. Present your calculations in a clear, organized manner

d. Calculate and report the **total number of trainable parameters in the model**

CNN Model Definition - Pytorch Code

```
import torch
import torch.nn as nn
import torch.nn.functional as F
from torchsummary import summary
from torchviz import make_dot
import torchvision


class CNN(nn.Module):

  def __init__(self, num_classes=10):

    super(CNN, self).__init__()

    self.conv1 = nn.Conv2d(in_channels=3, out_channels=8, kernel_size=3, stride=2)
    self.bn1 = nn.BatchNorm2d(8)
    self.relu1 = nn.ReLU(inplace=True)
    self.pool1 = nn.MaxPool2d(kernel_size=2, stride=2)

    self.conv2 = nn.Conv2d(in_channels=8, out_channels=16, kernel_size=3, stride=3)
    self.bn2 = nn.BatchNorm2d(16)
    self.relu2 = nn.ReLU(inplace=True)
    self.pool2 = nn.MaxPool2d(kernel_size=2, stride=2)

    self.fc1 = nn.Linear(576, 128)
    self.fc_relu1 = nn.ReLU(inplace=True)
    self.fc2 = nn.Linear(128, num_classes)

    self.softmax = nn.Softmax(dim=1)

  def forward(self, x):

    x = self.pool1(self.relu1(self.bn1(self.conv1(x))))
    x = self.pool2(self.relu2(self.bn2(self.conv2(x))))
    x = x.view(x.size(0), -1)
    x = self.fc2(self.fc_relu1(self.fc1(x)))

    return self.softmax(x)

# Initialize the model
model = CNN(num_classes=10)
# Display model summary
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = model.to(device)
print(summary(model, (3, 160, 160)))
```

4. **With reference to the Inverse Transform Sampling, as discussed in the class, please answer the following questions**: (Q4. Total = **20**)

   1. What is Inverse Transform Sampling, and when is it used? (3)
   2. Write a pseudocode for generating samples using Inverse Transform Sampling. (5)
   3. For the following distribution: $\mathrm{P}(X) = \dfrac{1}{\pi\gamma[1 + ((x - x_0)/2)^2]}$

      a. Identify the distribution. (2)

      b. Find the Cumulative Distribution Function (CDF) for the given distribution, i.e., $F_x(X)$. Here x_0 is the location parameter and $\gamma_0 > 0$ is the scale parameter. (6)

      c. Now, take $y = F_x(X)$ and find $\hat{X} = F_x^{-1}(y)$ to produce the target distribution $P(X)$ by sampling from the uniform distribution. (4)

5. **Answer the following questions with reference to Autoencoders and Variational Autoencoders:** (Q5. Total = **20**)

   (a) Justify whether a linear autoencoder with one hidden layer behaves similarly to PCA. State one key difference between their implementations or results. (2+3)

   (b) With respect to a Variational Autoencoder (VAE) with latent variable z and a dataset of samples $\{x_i\}_{i=1}^n$, answer the following questions and provide the corresponding mathematical expressions.

      a. **Prior Distribution:** What is the typical choice of the prior distribution in a VAE? How do we sample z from this distribution? (2)

      b. **Encoder and Decoder:** Define the Encoder and Decoder components of the VAE. Write down their mathematical forms under standard assumptions (use diagram, labeling each component clearly). Explain why a proxy distribution is used for the encoder instead of directly computing the true posterior. (3)

      c. **Derivation of the ELBO:** Starting from $\log p_\theta(x)$, derive the variational lower bound (ELBO). Show how $\log p_\theta(x) = \text{ELBO} + D_{\text{KL}}(\cdot \,||\, \cdot)$ is obtained. Clearly define and derive each term. (6)

      d. **Reparameterization Trick:** Explain the purpose of the reparameterization trick in VAEs. Describe how it is implemented and why it is necessary, with brief mathematical support. (4)

6. **Answer the following questions with reference to the Expectation-Maximization (EM) algorithm with parameters μ, ϕ, and Σ.** (Q6. Total = **14 marks**)

   (a) Write down the Expectation step (E-step) of the EM algorithm, specifying the mathematical expressions involved. (4)

   (b) Write down the Maximization step (M-step) of the EM algorithm, providing the corresponding mathematical formulas. (5)

   (c) Starting from the objective $\sum_{i=1}^{N} \log p(x^{(i)}; \theta)$, derive the Evidence Lower Bound (ELBO) used in the EM algorithm, and explain how it is maximized. (5)