Instructions:

- Write your answers in good legible handwriting. Don't scribble.
- Questions 1,2,3,4,11 are mandatory (**15 Marks**)
- Answer any 3 questions from questions 5,6,7,8,9,10. Don't try to answer all of them. If you answer more than 3, we will only consider the best 3 (**9 Marks**).

1. **(2 marks)** [**As promised**] Let $y = f(x) + \epsilon$, which means that y is related to x by some true function f but there is also some noise ϵ in the relation. Let $\hat{f}(x)$ be our approximation (model) for the relation between y and x. Show that

$$E[(y - \hat{f}(x))^2] = Bias^2 + Variance + \sigma^2$$

2. **(2 marks)** [**As promised**] Convolutional neural networks consist of a series of convolutional layers and max pooling layers. The max pooling layers do not contain any weights. Say the output of one such convolution layer is a 4×4 feature map denoted by h (comprising of $h_{11}, h_{12}, ..., h_{44}$). On top of this we have a 2×2 max pooling layer whose output is denoted by m (comprising of $m_{11}, m_{12}, m_{21}, m_{22}$). Say we have computed the derivative of the loss w.r.t. the output m of the max-pool layers. Express the derivative of the loss w.r.t. the output (h) of the convolution layer.

3. **(2 marks)** [**As promised**] In the 2-dimensional case, why are the contours of the L_1 loss squares and the contours of the L_2 loss circles? (No hand wavy answers please. Give mathematical arguments based on equations)

4. [**As promised**] Suppose $h_t = o_t \odot \sigma(s_t)$ where $h_t, o_t, s_t \in \mathbb{R}^n$
   (a) **(1 mark)** What is the derivative of h_t w.r.t. s_t? Vector or Matrix or Tensor?
   (b) **(1 mark)** How many non-zero entries does the above derivative have?

5. **(3 marks)** Suppose you are experimenting with L_1 and L_2 regularization. Further, imagine that you are running gradient descent and at some point your weight vector is $w = [1, \epsilon] \in \mathbb{R}^2$ where $\epsilon > 0$ is very small. With the help of this example explain why L_2 norm does not encourage sparsity *i.e.*, it will not try to drive ϵ to 0 to produce a sparse weight vector. We are looking for a mathematical explanation. No marks for hand wavy verbose answers.

6. **(3 marks)** Pappu was looking at the use of the Parametric ReLU function ($y = max(\alpha x, x)$) in the context of convolutional neural networks (assume that there is no max pooling layer). He got a brilliant idea and came up with a variant of ReLU called Double Dhamaka ReLU : $y = max(\alpha x, \beta x)$. He discussed this with his best friend Chameli and suggested that she should use this as the activation function in her convolutional neural networks. Alas! Chameli immediately pointed out something which made Pappu realize that adding β was **redundant**. What did Chameli tell Pappu ?

7. Consider a 3 layered neural network with one input layer, one hidden layer and one output layer and sigmoid activations for the neurons in the hidden layer. The dimensions of the input, hidden and output layer are 100, 100 and 20 respectively. You are trying to train this network using mini-batch gradient descent. When you pass the first mini-batch through the network, i.e., when you do a forward propagation for a given mini-batch you observe that all the neurons in the hidden layer output 1. Assume that all your inputs were standardized (zero mean and unit variance). Further, assume there were no biases in the network.
   (a) **(1 mark)** What would be the consequence of this ?
   (b) **(2 marks)** What could be the cause of this ? Remember that this is the first mini-batch and you have not computed or backpropagated any gradients yet. You cannot get away by saying that "there was a bug in the code" ?

8. **(3 marks)** Consider an e-commerce website which has m customers and n products. For every customer you have information about the products that were purchased by that customer. You are interested in finding customers which have very similar interests so that you can recommend the products purchased by one customer to another similar customer. Using the ideas that you have learned in the course can you think of a good way of learning the representation of each customer ?

9. The full set of equations for LSTMs are given below

   Gates:

$$\begin{aligned} o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o) \\ i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i) \\ f_t &= \sigma(W_f h_{t-1} + U_f x_t + b_f) \end{aligned}$$

   States:

$$\begin{aligned} \tilde{s}_t &= \sigma(W h_{t-1} + U x_t + b) \\ s_t &= f_t \odot s_{t-1} + i_t \odot \tilde{s}_t \\ h_t &= o_t \odot s_t \end{aligned}$$

   (a) **(1 mark)** Argue for or against the following statement: "The forget gate should rightfully be called the remember gate!".
   (b) **(2 marks)** Suppose you modify the equation of the "forget" gate such that

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + V_{ff} f_{t-1} + b_f) \tag{1}$$

   Would such a modified LSTM still be able to solve the problem of vanishing gradients ? Explain your answer with clear mathematical arguments.

10. In dropout we drop a node and all its outgoing edges from the network. It would sometimes make more sense to keep a node and drop only some weights outgoing form it. Effectively, this means that every node only participates in the computation of a few nodes in the next layer and not all nodes. Let us call this as drop-connect. Now consider two layers of an MLP $h_i \in \mathbf{R}^d$ and $h_{i+1} \in \mathbf{R}^d$.

    (a) **(1.5 marks)** Write numpy code to apply dropout between h_i and h_{i+1}

    (b) **(1.5 marks)** Write numpy code to apply drop-connect between h_i and h_{i+1}

    Note: The code should be efficient (no for loops). To put this question in context you need to understand that GPUs are very efficient at doing vector-vector, matrix-matrix and matrix-vector operations (multiplications, convolutions, Haddamard products). It is okay if you are a bit hazy with the numpy syntax. In case you are not sure of the syntax, just add a comment to every line of code so that we know what you actually meant to do.

11. Consider the task of Video QA where you want to generate a textual answer. For example,"Q: What is the person in the video doing? A: The person in the video is chopping vegetables.". Assume all videos are of the same length T and all answers are of the same length J. Here $Data = \{video_i, answer_i\}_{i=1}^{n}$ where $video_i = f_{i1}, f_{i2}, ..., f_{iT}$ and $answer_i = w_{i1}, w_{i2}, ..., w_{iT}$

    (a) **(1 mark)** Write down the equations of the encoder and the decoder ?

    (b) **(2 marks)** [**As promised**] How will you incorporate an attention mechanism into the above model ? Write down the exact set of equations that you will use.

    (c) **(2 marks)** Should the attention over the videos also depend on the question ? If so, how will you incorporate this ? In other words, how will you modify the equations that you suggested in part b to ensure that the attention also depends on the question ?

    (d) **(2 marks)** In many cases the number of frames in the video would be very large. For example, a video typically contains 25 frames per second and hence a 20 second video would contain 500 frames. The encoder LSTM thus has to run for many time-steps which is computationally expensive. One option is to uniformly sample only k% of the frames from the given 500 frames and then run the LSTM only over these sampled frames. However, this could lead to some loss of information. Suggest a smarter way of reducing the number of frames processed by the LSTM. Assume that you have infinite training time and you only care about reducing the inference time.