LUMS
A Not-for-Profit University

Department of Electrical Engineering
School of Science and Engineering

EE514/CS535 Machine Learning

ASSIGNMENT 3

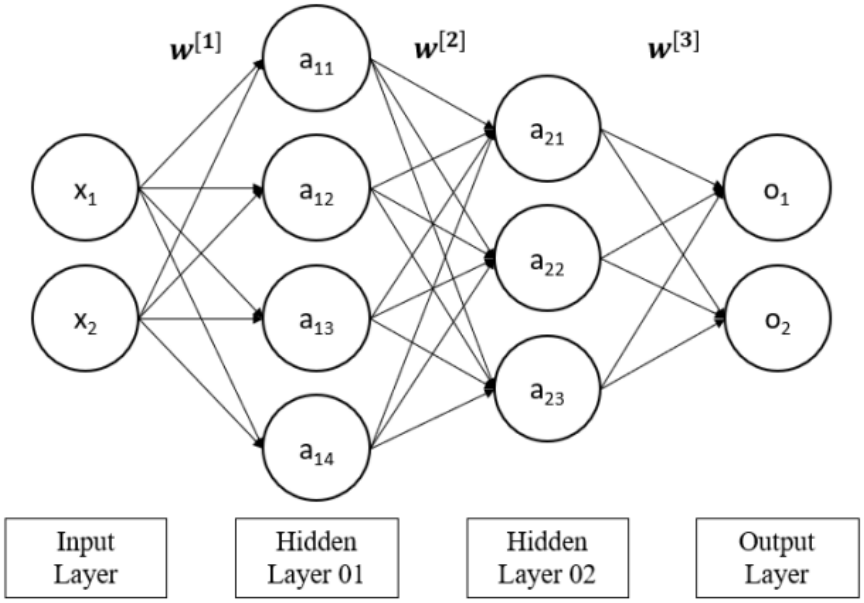
Due Date: 5 pm, Wednesday, May 7, 2025.
Format: 5 problems, for a total of 100 marks
Instructions:

- You are allowed to collaborate with your peers, but copying your colleague's solution is strictly prohibited. This is not a group assignment. Each student must submit his/her own assignment.
- Solve the assignment on blank A4 sheets and staple them before submitting.
- Submit in class or in the dropbox labeled EE-514 outside the instructor's office.
- Write your name and roll no. on the first page.
- Feel free to contact the instructor or the teaching assistants if you have any concerns.

- You represent the most competent individuals in the country; do not let plagiarism come in between your learning. In case any instance of plagiarism is detected, the disciplinary case will be dealt with according to the university's rules and regulations.

Problem 1 (20 marks)

Consider the neural network diagram given below:



Here $\mathbf{w_{ij}^{[k]}}$ represents the value of the weight belonging to the $\mathbf{k^{th}}$ layer (shown in the diagram above), between the $\mathbf{i^{th}}$ node in the input layer and $\mathbf{j^{th}}$ node in the output layer. For example, the weight between the nodes $\mathbf{a_{12}}$ and $\mathbf{a_{23}}$ will be represented as $\mathbf{w_{23}^{[2]}}$. Note that i and j will correspond to the row and column, respectively in the weight matrices given below:

$$\mathbf{w}^{[1]} = \begin{bmatrix} 0.9 & 0.75 & -0.15 & 0.33 \\ -0.79 & -0.03 & 0.1 & 0.02 \end{bmatrix}, \mathbf{w}^{[2]} = \begin{bmatrix} -0.58 & 1.0 & -0.8 \\ 0.23 & -0.45 & 0.05 \\ 0.25 & -0.65 & -0.09 \\ -0.57 & 0.51 & -0.33 \end{bmatrix}, \mathbf{w}^{[3]} = \begin{bmatrix} 0.97 & 0.3 \\ -0.01 & -0.2 \\ 0.85 & 0.76 \end{bmatrix}$$

- The activation function for the hidden layers is tanh, and sigmoid for the output layer.
- The cost function for the neural network is cross entropy.
- The learning rate used for weight updates is 0.01.
- The input values for x_1 and x_2 are 0.5 and 0.3 respectively.
- The actual values (ground truth) for o_1 and o_2 are 0.4 and 0.45 respectively.
- Assume that the biases are 0.

(a) [**4 marks**] Using the appropriate weights and activation function, compute the values of the nodes for the first hidden layer.

(b) [**2 marks**] Using the appropriate weights and activation function, compute the values of the nodes for the second hidden layer.

(c) [**2 marks**] Using the appropriate weights and activation function, compute the values of the nodes for the output layer.

(d) [**12 marks**] Compute the loss, then carry out one iteration of gradient descent using backpropagation.

Problem 2 (15 marks)

Consider multivariate linear regression, where the goal is to predict multiple output variables from multiple input features. Let:

- $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the input matrix, with n samples and d input features, where each row $x_i \in \mathbb{R}^d$ is a sample.
- $\mathbf{Y} \in \mathbb{R}^{n \times m}$ be the output matrix, with m output features, where each row $y_i \in \mathbb{R}^m$ is the corresponding target.
- $\mathbf{W} \in \mathbb{R}^{d \times m}$ be the weight matrix, such that the predicted output is $\hat{\mathbf{Y}} = \mathbf{XW}$
- The loss function is the mean squared error: $\mathbf{E}(\mathbf{W}) = \frac{\mathbf{1}}{\mathbf{2n}}\|\mathbf{Y} - \mathbf{XW}\|_{\mathbf{F}}^{\mathbf{2}}$ where $\|\cdot\|_F$ is the Frobenius norm.

(a) **[3 marks]** Derive the gradient of the L_2-regularized loss function with respect to the weight matrix $\mathbf{W}$ i.e. $\nabla_{\mathbf{W}} E(\mathbf{W})$. The regularized loss is defined as:

$$E_{\text{reg}}(\mathbf{W}) = \frac{1}{2n}\|\mathbf{XW} - \mathbf{Y}\|_F^2 + \frac{\lambda}{2}\|\mathbf{W}\|_F^2$$

where $\lambda > 0$ is the regularization parameter.

(b) **[12 marks]** Now, consider adding white Gaussian noise to the inputs. Define the noisy input matrix as:

$$\mathbf{X}' = \mathbf{X} + \mathbf{E},$$

where $\mathbf{E} \in \mathbb{R}^{n \times d}$ has entries $e_{ij} \sim \mathcal{N}(0, \sigma^2)$, independently sampled, so each sample becomes:

$$\mathbf{x}_i' = \mathbf{x}_i + \epsilon_i,$$

with $\epsilon_i \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d)$. The loss with noisy inputs is:

$$L'(\mathbf{W}; \mathbf{X}', \mathbf{Y}) = \frac{1}{2n}\|(\mathbf{X} + \mathbf{E})\mathbf{W} - \mathbf{Y}\|_F^2.$$

For a single sample i, compute the gradient of the noisy loss:

$$L_i'(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}^T(\mathbf{x}_i + \epsilon_i) - \mathbf{y}_i\|_2^2,$$

with respect to $\mathbf{W}$.

(1) **[4 marks]** Calculate the gradient $\nabla_{\mathbf{W}} L_i'$.

(2) **[5 marks]** Compute the expected gradient $\mathbb{E}_{\epsilon_i}[\nabla_{\mathbf{W}} L_i']$ over the noise distribution.

(3) **[3 marks]** Compare this expected gradient to the gradient from Part (a). Show that adding noise to the inputs is equivalent to L_2 regularization, and determine the relationship between the noise variance σ^2 and the regularization parameter λ.

Problem 3 (30 marks)

Consider a vanilla Recurrent Neural Network (RNN) processing a sequence of length $T = 3$, with no biases for simplicity. The RNN has the following components:

- **Input**: A sequence of vectors $\mathbf{x}_t \in \mathbb{R}^2$ for $t = 1, 2, 3$.
- **Hidden state**: $\mathbf{h}_t \in \mathbb{R}^2$, with initial state $\mathbf{h}_0 = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$.
- **Output**: $\mathbf{y}_t \in \mathbb{R}^1$ (scalar output at each time step).
- **Weights**: $\mathbf{W}_{xh} = \begin{bmatrix} 0.5 & 0.1 \\ 0.2 & 0.3 \end{bmatrix}$ $\mathbf{W}_{hh} = \begin{bmatrix} 0.4 & 0.1 \\ 0.1 & 0.3 \end{bmatrix}$ $\mathbf{W}_{hy} = \begin{bmatrix} 0.6 & 0.4 \end{bmatrix}$
- **Activation**: Tanh for the hidden state, linear (identity) for the output.

The input sequence and true labels are:

$\mathbf{x}_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$, $y_1 = 0.5$, $\mathbf{x}_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}^T$, $y_2 = 0.7$, $\mathbf{x}_3 = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$, $y_3 = 1.0$

The loss function is the mean squared error over the sequence:

$$L = \frac{1}{3} \sum_{t=1}^{3} (y_t - \hat{y}_t)^2$$

(a) **[14 marks]** Derive the general formulation of Backpropagation Through Time (BPTT) for a vanilla RNN with the structure given above, applicable to any sequence length T.

(1) **[2 marks]** Define the total loss $L = \frac{1}{T} \sum_{t=1}^{T} (y_t - \hat{y}_t)^2$. Write the recursive relationships for $\mathbf{h}_t$ and y_t in terms of $\mathbf{x}_t$, $\mathbf{h}_{t-1}$, and the weights.

(2) **[8 marks]** Derive the general expressions for the gradients at each time step using the chain rule:

- $\frac{\partial L}{\partial y_t}$
- $\frac{\partial L}{\partial \mathbf{h}_t}$, showing how it depends on both y_t and future hidden states $\mathbf{h}_{t+1}$
- $\frac{\partial L}{\partial \mathbf{W}_{hy}}$, $\frac{\partial L}{\partial \mathbf{W}_{xh}}$, and $\frac{\partial L}{\partial \mathbf{W}_{hh}}$

(3) **[4 marks]** Find the expression explaining how these gradients are accumulated over time steps to compute the total gradients $\nabla_{\mathbf{W}_{xh}} L$, $\nabla_{\mathbf{W}_{hh}} L$, and $\nabla_{\mathbf{W}_{hy}} L$.

(b) **[6 marks]** Compute the forward pass through the RNN for the given sequence, i.e., the hidden state h_t and the output y_t for each timestep ($t = 1, 2, 3$).

(c) **[7 marks]** Using the true labels, perform BPTT to compute the gradients of the loss L with respect to the weights $\mathbf{W}_{xh}, \mathbf{W}_{hh}, \mathbf{W}_{hy}$

(1) **[2 marks]** Compute the loss L using the outputs from Part (a).

(2) **[5 marks]** Using your derivation from (b) compute the following for t = 3:

- $\frac{\partial L}{\partial y_t}$.
- $\frac{\partial L}{\partial \mathbf{h}_t}$ accounting for contributions from both y_t and $\mathbf{h}_{t+1}$ (for $t < 3$).
- $\frac{\partial L}{\partial \mathbf{W}_{hy}}$, $\frac{\partial L}{\partial \mathbf{W}_{xh}}$, $\frac{\partial L}{\partial \mathbf{W}_{hh}}$

(d) **[3 marks]** Vanilla Recurrent Neural Networks (RNNs) often encounter vanishing or exploding gradient issues during backpropagation through time, limiting their ability to learn long-term dependencies. Explain how the Long Short-Term Memory (LSTM) architecture addresses these challenges. Specifically, describe the key components of the LSTM architecture and how they regulate gradient flow to mitigate these problems, providing insight into the mathematical or structural mechanisms involved.

Problem 4 (20 marks)

Consider a dataset $\mathbf{X} \in \mathbb{R}^{n \times d}$ with n samples and d features, where each row $\mathbf{x}_i \in \mathbb{R}^d$ represents a data point. Assume the data is centered, i.e., $\mathbf{X}_c = \mathbf{X} - \mathbf{1}\mu^T$, where $\mu = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ is the mean vector. In this question, you will explore a linear autoencoder as a dimensionality reduction technique and examine how it relates to Principal Component Analysis (PCA).

For reference, PCA reduces the dimensionality of the dataset to $k < d$ dimensions by projecting the data onto the directions of maximum variance. The covariance matrix of the centered data is defined as:

$$\mathbf{C} = \frac{1}{n}\mathbf{X}_c^T\mathbf{X}_c$$

PCA finds the top k principal components as the eigenvectors of $\mathbf{C}$ corresponding to the k largest eigenvalues. Let $\mathbf{U} \in \mathbb{R}^{d \times k}$ be the matrix whose columns are these eigenvectors, satisfying $\mathbf{U}^T\mathbf{U} = \mathbf{I}_k$. The reconstruction of the centered data is:

$$\hat{\mathbf{X}}_c = \mathbf{X}_c\mathbf{U}\mathbf{U}^T$$

The reconstruction error is:

$$E = \frac{1}{n}\|\mathbf{X}_c - \mathbf{X}_c\mathbf{U}\mathbf{U}^T\|_F^2$$

Minimizing this error corresponds to maximizing the variance captured by the k-dimensional subspace spanned by $\mathbf{U}$.

(a) **[4 marks]** Briefly explain how PCA uses the eigenvectors of the covariance matrix $\mathbf{C}$ to find the principal components, and why the reconstruction error E is minimized when $\mathbf{U}$ contains the top k eigenvectors.

(b) **[8 marks]** A linear autoencoder is a neural network with no non-linear activation functions, designed to compress and reconstruct the data. It consists of:

- **Encoder**: $\mathbf{z}_i = \mathbf{W}^T\mathbf{x}_i$, where $\mathbf{W} \in \mathbb{R}^{d \times k}$ and $\mathbf{z}_i \in \mathbb{R}^k$ is the latent representation ($k < d$).
- **Decoder**: $\hat{\mathbf{x}}_i = \mathbf{W}\mathbf{z}_i = \mathbf{W}\mathbf{W}^T\mathbf{x}_i$, assuming tied weights (the decoder uses $\mathbf{W}$).

The objective is to minimize the reconstruction error over the centered dataset:

$$L(\mathbf{W}) = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \mathbf{W}\mathbf{W}^T\mathbf{x}_i\|_2^2 = \frac{1}{n}\|\mathbf{X}_c - \mathbf{X}_c\mathbf{W}\mathbf{W}^T\|_F^2$$

(1) **[4 marks]** Compute the gradient $\nabla_{\mathbf{W}} L(\mathbf{W})$ with respect to $\mathbf{W}$.

(2) **[4 marks]** Set the gradient to zero and solve for the condition that $\mathbf{W}$ must satisfy at the optimum. Interpret this condition in terms of the covariance matrix $\mathbf{C}$.

(c) **[8 marks]** Investigate how the linear autoencoder resembles PCA based on your results and the PCA formulation provided.

(1) **[2 marks]** Compare the reconstruction error expressions E from PCA and $L(\mathbf{W})$ from the autoencoder. What similarities do you observe between $\mathbf{U}\mathbf{U}^T$ in PCA and $\mathbf{W}\mathbf{W}^T$ in the autoencoder?

(2) **[4 marks]** Using the condition from Part (b)(2), show that the optimal $\mathbf{W}$ in the linear autoencoder spans the same subspace as the principal components from PCA. Discuss whether $\mathbf{W}$ must be orthonormal like $\mathbf{U}$.

(3) **[2 marks]** Conclude whether a linear autoencoder with tied weights is equivalent to PCA, and explain any conditions or assumptions required for this equivalence to hold.

Problem 5 (15 marks)

Consider a Bayesian linear regression model with 1D inputs x and scalar weights w:

$$y = wx + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2),$$

with Gaussian prior $w \sim \mathcal{N}(0, \lambda^{-1})$. You observe three data points:

$$\mathcal{D} = \{(1, 2), (2, 3), (3, 4)\}, \quad \sigma^2 = 0.1, \quad \lambda = 1.$$

(a) **[7 marks]** Derive the closed-form expression for the MAP estimate w_{MAP}. Compute w_{MAP} numerically using the given data.

(b) **[3 marks]** Compute the Hessian H of the negative log-posterior at w_{MAP}. Write the Laplace-approximated posterior $p(w|\mathcal{D})$.

(c) **[3 marks]** For a new input $x^* = 2$:

1. Compute the predictive distribution $p(y^*|x^*, \mathcal{D})$
2. Report the predictive mean $\mathbb{E}[y^*|x^*, \mathcal{D}]$ and variance $\text{Var}(y^*|x^*, \mathcal{D})$

(d) **[2 marks]** Explain why the predictive variance increases as $|x^*|$ grows

— End of Assignment —