1 Neural Network (15 points)

Consider the following neural network with one hidden layer and two prediction tasks: one is reconstruction of x and the other is regression of target y. Each hidden layer node is defined as $Z_k = \sigma(h_k) = \sigma(\sum_{i=1}^{3} w_{ki}\tilde{x}_i)$ for $k = 1, \ldots, 4$. The reconstruction is defined as $\hat{x}_i = \sum_{k=1}^{4} w_{ik} Z_k$ for $i = 1, \cdots 3$ and regression output is defined as $y_j = \sum_{k=1}^{4} v_{jk} Z_k$ for $j = 1, 2$.

Suppose we choose the squared loss function for every pair, i.e.

$$L(\mathbf{x}, \widehat{\mathbf{x}}) = \frac{1}{2}\left((x_1 - \widehat{x}_1)^2 + (x_2 - \widehat{x}_2)^2 + (x_3 - \widehat{x}_3)^2\right)$$

and

$$L(\mathbf{y}, \widehat{\mathbf{y}}) = \frac{1}{2}\left((y_1 - \widehat{y}_1)^2 + (y_2 - \widehat{y}_2)^2\right)$$

where y_j and $\widehat{y}_j$ represent the true outputs and our estimations, respectively; x_i, $\tilde{x}_i$ and $\widehat{x}_i$ represent the true variable, noised input variable and predicted reconstruction (Note: we have used $\tilde{x}_i$ instead of x_i as input to improve the generalization capability of model).

Write down the backpropagation updates for estimation of w_{ki} and v_{jk}. For simplicity, only consider single data sample.

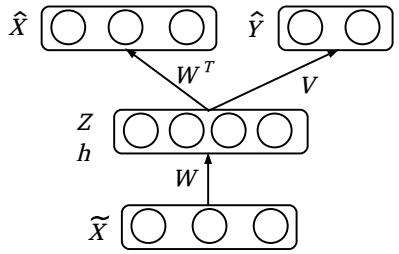

Figure 1: The structure of the neural network (Multi-tasking Auto-Encoder). The input variable $\tilde{x}$ and top layer reconstruction x are both 3-dimensional vector; middle layer hidden representation h is 4-dimensional vector; the prediction y is 2-dimensional vector.

2 Mixture Model and EM Algorithm (15 points)

Suppose $X_1, \ldots, X_n$ are i.i.d distribution random variables with the density function $f_X(x, \lambda) = \lambda e^{-\lambda x}$, for $x \geq 0$ and 0 otherwise. We observe $Y_i = \min\{X_i, c_i\}$ for some fixed known c_i. Assume (for simplicity) that this cut-off happens only for the last $n - r$ variables; i.e. $y_i = c_i$ for $i = r+1, \ldots, n$. The goal is to estimate the value of λ using EM algorithm.

(a) Write the log-likelihood in terms of unobserved variables X_i.

(b) Write down the E-Step and take the expectation.

(c) Write down the M-Step and find the new value of λ.

3 K-Means (Bonus: 10 points)

Consider the K Means algorithm with following assumptions:
1. K clusters $\{C_k\}_{k=1}^K$ with centers $\mathcal{U}_K = \{\mu_k\}_{k=1}^K \subseteq \mathbb{R}^d$
2. Mapping function $f : \mathcal{X} \to \{1, \cdots, K\}$ will assign the proper cluster to a data points by minimizing the following function:
$\phi = \sum_{k=1}^K \frac{1}{n_k} \sum_{i=1}^{n_k} \sum_{j=1}^{n_k} \|x_{ki} - x_{kj}\|^2$

(a) Prove that for any point s if we have a set of points $\mathcal{X} \subseteq \mathbb{R}^d$ with centers $\bar{x}$. Then we have:

$\sum_{x \in \mathcal{X}} \|x - s\|^2 - \sum_{x \in \mathcal{X}} \|x - \bar{x}\|^2 = |\mathcal{X}| \cdot \|\bar{x} - s\|^2$

(b) K-Medoids is an algorithm for clustering which was proposed for working with other distance metrics such as l_1 norm. In those applications that we want the cluster centers to be from one of the points in the dataset itself, we use K-Medoids. So this algorithm starts by initializing the cluster centers(i.e. called Medoids here) randomly from the existing points and continue by assigning the cluster of the closest Medoid to each point and finally, find the Medoid within each cluster by not averaging but going through all the points existing in that cluster. This means each Medoid is the most centrally located but not exactly centrally located point within each cluster.

Considering the performance of K-Medoids and its difference with K-Means, compare K-Means and K-Medoids algorithms in terms of efficiency, sensitivity to outliers and objective functions.

4 Programming

4.1 K-means (15 points)

In this problem, you will implement K-means algorithm. You will evaluate the performance of your method on two synthetic datasets - `hw4_blob.mat` and `hw4_circle.mat`. Both datasets have two dimensions.

K-means tries to minimize the following distortion measure (or objective function):

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x}_n - \boldsymbol{\mu}_k||_2^2$$

where r_{nk} is an indicator variable:

$$r_{nk} = 1 \quad \text{if and only if } \mathbf{x}_n \text{ belongs to cluster } k$$

and $(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)$ are the cluster centers with the same dimension of data points.

(a) Implement k-means using random initialization for cluster centers. The algorithm should run until none of the cluster assignments are changed. Run the algorithm for different values of $K \in \{2, 3, 5\}$, and plot the clustering assignments by different colors and markers. (you need to report 6 plots, 3 for each dataset.)

(b) Does the k-means algorithm separate the two circles in the `hw4_circle.mat` dataset? Explain.

4.2 Mixture of Gaussian Distributions with Unknown Component Numbers (30 points)

In this problem we will experiment with EM algorithm on mixture of Gaussian distribution, with unknown number of components. We will learn the following parameters $\{K, \{\mu_k, \sigma_k, \lambda_k\}_{k=1\cdots K}\}$: K is the number of Gaussian components, λ_k is the prior probability of k-th component, (μ_k, σ_k) are parameters of k-th component.

(a) Load the data from emGMM.mat, and you will get a training data matrix, *dataTr*, and a validation data matrix, *dataVal*.

(b) Please **write your own code** of EM algorithm to train GMM models using each of the following numbers of Gaussian components: $\{3, 5, 7, 9, 11\}$. In this experiment, please

- assume a *full covaraince matrix* for every component (i.e. not diagonal);
- try *5 random initialization* of parameters: in initialization of each component's $\{\lambda_k, \mu_k\}$ (of course you need to make sure λ is normalized) use random values, while in initialization of covariance matrix, use the *same random diagonal matrix* for all components (note that we tie covariance matrix of different components only in initialization, and they will be updated independently in EM learning). In practice, with some initialization, the EM algorithm may suffer from 'Singular Matrix'. In this case, discard this initialization and try different random initialization.

(c) At the end of above experiments, you will have $5 \times 5 = 25$ GMM models, i.e. 5 possible K values, and 5 models corresponding to each K value. For each value of components number, from the 5 models, please select the one with best log-likelihood on heldout data and ***report*** (1) the number of iterations till convergence, (2) marginal log-likelihood of training data, (3) marginal log-likelihood of heldout data.

(d) How many components are you going to select, why?

4.3 Vector Quantization Using k-means (Bonus: 15 points)

One important application of k-means is vector quantization. This is the process of replacing our data points with the prototypes μ_k from the clusters they are assigned to. This technique can be used for image compression.

Download the image `hw4.jpg`, and perform k-means clustering to vector-quantize pixels according their RGB color. Reconstruct the image using the colors in the centers for values of $K \in \{4, 8, 24\}$. Show the reconstructed images. (you need to show three reconstructed images.) (hw4.jpg source: https://en.wikipedia.org/wiki/Louisbourg_Lighthouse)