CSE7850/CX4803 - Spring 2024 - Homework 3

1 Protein structure and sequence co-evolution [20 pts]

In our lecture, we have seen that sequence co-evolution relationships can be used to assist the protein structure prediction. Given a protein sequence $\mathbf{x} = (x_1, x_2, \ldots, x_L)$ for which we want to predict the structure, we can search its homologous (evolutionarily related) sequences in the database and form a multiple sequence alignment (MSA). Next, we can use a generative model called Markov Random Field (MRF, also known as Potts model, Ising model, or undirected graphical model) to calculate the probability distribution for sequences in the MSA. As introduced in the lecture, the probability of a sequence x of length L is defined as

$$p(\mathbf{x}) = \frac{1}{Z} \exp[E(\mathbf{x})], \tag{1}$$

where Z is a normalization constant that ensures $p(\mathbf{x})$ is a probability in $[0, 1]$. In MRF, $E(\mathbf{x})$ is defined using two groups of parameters as following

$$E(\mathbf{x}) = \sum_i e_i(x_i) + \sum_{i<j} e_{ij}(x_i, x_j), \tag{2}$$

where $x_i \in \Sigma$ is the amino acid of sequence $\mathbf{x}$ at the i-th position and Σ is the alphabet of MSA (e.g., 20 possible amino acids and the gap). The parameters $e_i(x_i)$ are called single-site potentials and parameters $e_{ij}(x_i, x_j)$ are called pairwise potentials[1]. Now let us understand this model in more detail by considering the following questions.

(1) [2pts] To make $p(\mathbf{x})$ a valid probability distribution, the normalization constant Z can take the form $Z = \Sigma_{x \in \mathcal{S}} \exp[E(\mathbf{x})]$ for some set of sequences $\mathcal{S}$. What sequences should be included in $\mathcal{S}$ such that $p(\mathbf{x})$ gives the probability of a sequence x of length L? How large is $|\mathcal{S}|$?

(2) [4pts] If we build an MRF model described in Eq, 2 for an MSA of sequences with length L, and assume the alphabet is $|\Sigma| = q$, how many parameters does this MRF have?

(3) [4pts] In our lecture, we mentioned that the single-site potentials e_i can reflect the preference of an amino acid appearing at position i. For example, if $e_i(x_i)$ has a larger value compared to other $e_i(x_j)$ $(j \neq i)$, then it means the amino acid x_i will have a higher likelihood showing at position i. We also mentioned that the pairwise potentials e_{ij} can reflect co-evolving residue pairs. Now, if we have built an MRF model and learned all its parameters, and suppose that residues i and j are co-evolving pairs, while residues u and v are non co-evolving pairs. Please describe, in your own words, how would the values in e_{ij} (note that e_{ij} is a matrix with size $|\Sigma| \times |\Sigma|$) be different from those in e_{uv} ? In other words, intuitively, are there any signals/patterns in e_{ij} that can help you tell which residue pairs are co-evolving?

(4) [3pts] Based on the above intuition, can you design a "co-evolution score" $c(i, j)$ using e_{ij} such that $c(i, j)$ has a higher value when residues are co-evolving than when they are not?

(5) [7pts] Sometimes we may be interested in comparing which sequences are more plausible, or more likely to appear in the evolution history (i.e., with a relatively higher probability $p(\mathbf{x})$). Consider a sequence $\mathbf{x}$ where the k-th position is amino acid a, i.e., $\mathbf{x} = (x_1, \ldots, x_k = a, \ldots, x_L)$, and another sequence $\mathbf{x}'$

[1]Note that the summation of e_{ij} here is over $\{i < j\}$, which simplifies the summation over $\{i \neq j\}$, due to symmetry

that differs with $\mathbf{x}$ by only one position: the amino acid at position k changed from $x_k = a$ to $x_k = b$, i.e., $\mathbf{x}' = (x'_1, \ldots, x'_L) = (x_1, \ldots, x_k = b, \ldots, x_L)$. Now we want to decide which sequence is more plausible in the evolution history. One way to do this is to build an MRF on the homologous sequence of $\mathbf{x}$. With the MRF, one can compute $p(\mathbf{x})$ and $p(\mathbf{x}')$ as in Eq. 1 and compare the log-odds ratio of sequence probabilities between $\mathbf{x}$ and $\mathbf{x}'$:

$$\Delta E(\mathbf{x}', \mathbf{x}) = \log \frac{p(\mathbf{x}')}{p(\mathbf{x})} = E(\mathbf{x}') - E(\mathbf{x}) \tag{3}$$

If $\Delta E(\mathbf{x}', \mathbf{x}) > 0$, or equivalently $p(\mathbf{x}') > p(\mathbf{x})$, that means $\mathbf{x}'$ has a higher likelihood of occurrence than $\mathbf{x}$ in the evolution. Now you want to compute the value of $\Delta E(\mathbf{x}', \mathbf{x})$ based on Eqs. 1, 2, and 3. Suppose the MRF model has been built for that MSA data, and all its parameters (e_i and e_{ij}) are stored in an oracle that you can query. Each time, you can query the oracle for the value of a *single* parameter $e_i(a)$ or $e_{ij}(a, b)$ for any position i and any amino acid types a or b. You goal is to minimize the number of queries you make. What is the minimum number of queries you have to make to compute $\Delta E(\mathbf{x}', \mathbf{x})$? And what are those parameters you need to query? You need to justify you answer.

2 Graph Neural Networks (GNNs) Basics [15 pts]

(1) [5 pts] Graph Neural Networks (GNNs) are an important class of deep learning models used for processing graph-structured data. As we have shown in Lecture 15, one of the key properties of GNNs is that they should be able to produce the same output regardless of the input order of the nodes in a graph. In this problem, you will be presented with four undirected unweighted graphs, each with a set of nodes and edges. Your task is to identify which graphs are isomorphic as the first graph.

Formally, we say Graph G and Graph H are *isomorphic* when there exists a bijection f between the vertex sets of G and H such that any two vertices u and v of G are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in H.

Which of Graph 2, Graph 3, and Graph 4 is isomorphic as Graph 1? Explain your reasoning.

Note: All graphs are **undirected** and unweighted, and the edges are given in the table below. Each row in the table represents one edge in the graph. For example, for a graph $G = (V, E)$, an edge $(u, v) \in E$ will be presented as $(Edge[0], Edge[1])$ in one row of the table.

Edge[0]	Edge[1]
1	2
1	8
2	3
2	5
2	6
3	4
4	5
5	6
4	7
4	8
6	7

Table 1: Graph 1

Edge[0]	Edge[1]
1	4
1	8
2	4
2	7
3	5
3	7
3	8
4	5
4	6
5	6
5	7

Table 2: Graph 2

Edge[0]	Edge[1]
1	4
1	7
2	4
2	7
3	4
3	6
3	8
4	5
5	6
5	7
7	8

Table 3: Graph 3

Edge[0]	Edge[1]
1	3
1	5
1	6
1	8
2	7
2	8
3	7
3	8
4	5
4	7
6	7

Table 4: Graph 4

(2) [5 pts] Let's do a simple node classification GNN by hand here. We will use Graph 1 $G_1 = (V, E)$ from Table 1. The following Table 5 shows the features and labels of the nodes.

Node	$x[0]$	$x[1]$	y
1	0	1	0
2	1	1	1
3	3	0	1
4	4	1	1
5	3	1	1
6	2	3	0
7	4	3	0
8	5	1	0

Table 5: Node Features of Graph 1

For every node $v \in V$, we can compute the hidden embedding h_v as follows:

$$h_v = \frac{\sum_{u \in N(v) \cup \{v\}} \sigma(W_1 x_u + bias_1)}{|N(v)| + 1}$$

where $N(v)$ is the neighbour set of v, σ is the activation function. Here W_1 has the format

$$\begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix}$$

$bias_1$ has the format

$$\begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

For every v in V, we get its predicted class from

$$y_{pred,v} = \sigma(W_2 h_v + b_3)$$

W_2 has the format

$$\begin{pmatrix} w_{31} & w_{32} \end{pmatrix}$$

We will use the *step* function as the activation function σ. The definition of the step function is as follows: `step`(x) returns $+1$ if x is positive, and 0 otherwise.

Please answer the following two questions based on this.

If the parameters $w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}$ have the same value as 1.0, the bias b_1, b_2, b_3 have the same value -1.5, what's the predicted class of all the nodes from 1 to 8?

(3) [5 pts] Find a set of parameters $w_{11}, w_{12}, ..., w_{31}, w_{32}, b_3$ to make the predicted y the same as the labels in Table 5.

Submission Notes: For Q2.3, please write down your answers in a text file and named the file as "weights.txt". There should be nine numbers in the file, separated by **spaces**. The order is $w_{11}, w_{12}, b_1, w_{21}, w_{22}, b_2, w_{31}, w_{32}, b_3$. Compress the file weights.txt together with other code files that need to be submitted into a zip and upload it to gradescope.

3 Protein design by hand [15 pts]

In our lecture on protein structure, we have seen a demo of protein design by hand. In this problem, you have the chance to continue working on this task to manually design a protein sequence that could fold into a particular 3D structure. Specifically, you will need to design sequences that can fold into a protein with two helices (Task 1) and a protein with two beta sheets (Task 2). Please see the details in the following notebook.
`https://colab.research.google.com/drive/12sPR20gWv_RwKNa3_EZsIDgejcouAomY?usp=sharing`

4 Programming: Variational autoencoder (VAE) [20 pts]

Please see the statement of this question in the following Google Colab:
`https://colab.research.google.com/drive/1hrDQObKR2HV6ET4mUmdM5I0SXJZ7Gzgr?usp=sharing`

5 Programming: Graph neural network (GNN) [30 pts]

Please see the statement of this question in the following Google Colab:
`https://colab.research.google.com/drive/1OF7qaLcrSnIVapwnp5AsZO4uAHUn4Yq7?usp=sharing`

Submission instructions

For the first part, you will need to write the answers to written questions (including Q3) in a PDF file. Please submit this PDF file to the "HW3-pdf" assignment in Gradescope.

For the second part, you will need to compress all of the answers or codes for the remaining questions into a single zip file. Please make sure to compress the files directly instead of compressing the root directly. Once you have created the zip file, please submit it to the "HW3-Code" assignment in Gradescope.

Please note that it is important to follow these instructions carefully to ensure that your submission will be properly received and graded. If you have any questions about the submission process, please don't hesitate to reach out to your TA or the course instructor for assistance. Good luck with your assignment!