CS 4440 - Assignment #2

Total Marks: 20; Due date: March 18, 2022 (11.59 pm)

STRICTLY NO LATE SUBMISSIONS

What you need to submit:

A single zip file that contains the following:

1. Executable source code
2. A pdf file with your results (*classification accuracy score of 10-fold runs, average accuracy over 10 runs, and predicted labels of the testing sequences*).

Programming language: Use as per the course outline.

The objective of this assignment is to implement and test a basic machine learning classification model. You may reuse your code from assignment 1.

Step1 (Reading and pre-processing data): You will use the provided SARS-CoV-2 dataset (*similar to the one that you worked with in assignment-1*). Read the fasta files containing SARS-CoV-2 sequences (*all 2000 sequences from the training dataset with four variants- alpha, beta, delta, and gamma*). Make sure that your sequences have only the occurrences of uppercase A, C, G, and T.

> **Any inbuilt or existing implementation for reading the fasta files can be used.*
>
> **fasta file is like any other text file. The first line contains "> Header"; the rest of the lines contain the sequence.*

Step2 (Computing feature vectors): You will compute the two-dimensional numerical representations of the genomic sequences using the Chaos Game Representation (CGR). Generate the CGR plots with k = 7 for all sequences i.e., one CGR per sequence. CGR code is provided as part of this assignment. Please note that the CGR at k=7 represents the frequencies of all possible sub-words of length 7 constructed over the alphabet set {A,C,G,T}.

Step3 (Training ML Model): Use CGRs (*from step 2*) as feature vectors to perform supervised machine learning classification using any one algorithm of your choice. You need to use 10-fold cross-validation to compute the classification accuracy score. Perform the following tasks:

1. Use 10-fold cross-validation i.e., make 10 random folds (splits of equal size) of your dataset. Then use sequences from 9 folds for training and sequences from the remaining fold for testing. Repeat the process 10 times, every time selecting a different fold for testing and the remaining 9 folds for training.
2. Use CGRs as your feature vectors. You need to flatten the 2D CGRs to transform them into 1-dimensional vectors. Also, normalize these vectors so that all the frequencies in any CGR-based 1D vector are between 0 and 1. Think about how you can normalize these vectors? You should be dividing these vectors by some number, but what number?
3. Select any machine learning algorithm of your choice. Using 10-fold cross-validation and CGR-based 1D vectors as feature vectors, compute the classification accuracy of your selected model. Your class labels are alpha, beta, delta and gamma. Print the average classification accuracy score (*percentage of correctly predicted sequences over 10-folds*) of your model.

Step4 (Testing using trained ML model): Read the testing sequences from the provided fasta files and generate CGRs for them using k=7. Using the trained (generalized) model from the previous step, predict the labels of the testing sequences (*using their 1D CGRs as feature vectors*). Print the predicted labels of the testing sequences.

CGR code is provided on the next page.

MATLAB:

Example: sequence= ACGGGAT and k-value is 3

cgrPlot = cgr('ACGGGAT', 'ACGT', 3);

```
function [out] = cgr(chars, order, k)
    out = zeros(2^k);
    x = 2^(k-1);
    y = 2^(k-1);
    for i = 1:length(chars)
        char = chars(i);
        x = fix(x/2);
        if char == order(3) || char == order(4)
            x = x + 2^(k-1);
        end
        y = fix(y/2);
        if char == order(1) || char == order(4)
            y = y + 2^(k-1);
        end
        if i >= k
            out(y+1, x+1) = out(y+1, x+1) + 1;
        end
    end
end
```

Java:

Example: String seq="AGGCTAGCCCTT"; String order="ACGT"; int k=3;

int arr[][]=cgr(seq,order,k);

```
public static int[][] cgr(String seq, String order, int k)
  {
    int len = seq.length();
    int pw = (int)Math.pow(2,k);
    int[][] out = new int[pw][pw];
    int x = (int)Math.pow(2,k-1);
    int y = (int)Math.pow(2,k-1);

    for(int i=0;i<len;i++)
    {
      char ch = seq.charAt(i);
      x = x/2;
      y = y/2;
      if(ch == order.charAt(2) || ch == order.charAt(3))
        x = x + (int)Math.pow(2,k-1);
      if(ch == order.charAt(0) || ch == order.charAt(3))
        y = y + (int)Math.pow(2,k-1);
      if(i>=k-1)
        out[y][x] = out[y][x]+1;
    }
    return out;
  }
```

R:

Example:

```
seq="AGGCTAGCCCTT "
res = cgr(seq, "ACGT", 3)
```

```
library(stringr)
cgr <- function(seq, order, k){
ln = str_length(seq)
pw = 2^k
out = matrix(0, pw, pw)
x = 2^(k-1)
y = 2^(k-1)
for(i in 1:ln){
x=floor(x/2)
y=floor(y/2)
ch=substr(seq, i, i)
if(ch == substr(order, 3, 3) | ch == substr(order, 4, 4))
x=x+(2^(k-1))
if(ch == substr(order, 1, 1) | ch == substr(order, 4, 4))
y=y+(2^(k-1))
if(i>=k)
out[y+1,x+1]=out[y+1,x+1]+1
}
return(out)
}
```

Python:

Example:

```
res = cgr(seq="AGGCTAGCCCTT",order="ACGT", k=3)
```

```
def cgr(seq, order, k):
    ln = len(seq)
    pw = 2**k
    out = [[0 for i in range(pw)] for j in range(pw)]
    x = 2**(k-1)
    y = 2**(k-1)

    for i in range(0,ln):
        x=x//2
        y=y//2
        if(seq[i] == order[2] or seq[i] == order[3]):
            x = x + (2**(k-1))
        if(seq[i] == order[0] or seq[i] == order[3]):
            y = y + (2**(k-1))
        if(i>=k-1):
            out[y][x] = out[y][x]+1

    return out
```

C++:

Example:

```
    string seq="AGGCTAGCCCTT";
    char order[]="ACGT";
    int k=3;
    int** res = cgr(seq,order,k);
```

```
#include <iostream>
#include <math.h>
#include <string.h>
int** cgr(string seq, char order[], int k)
{
    int len = seq.length();
    int pw = pow(2,k);
    int** out = new int*[pw];
    for(int i=0;i<pw;i++)
    {
        out[i]=new int[pw];
        for(int j=0;j<pw;j++)
        {
            out[i][j]=0;
        }
    }

    int x = pow(2,k-1);
    int y = pow(2,k-1);

    for(int i=0;i<len;i++)
    {
        char ch = seq[i];
        x = x/2;
        y = y/2;

        if(ch == order[2] || ch == order[3])
            x = x + pow(2,k-1);

        if(ch == order[0] || ch == order[3])
            y = y + pow(2,k-1);

        if(i>=k-1)
            out[y][x] = out[y][x]+1;
    }
    return out;
}
```

C:

Example:

```
    char* seq="AGGCTAGCCCTT";
    char order[]="ACGT";
    int k=3;
    int** res = cgr(seq,order,k);
```

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
int** cgr(char* seq, char order[], int k)
{
    int pw = pow(2,k);

    int** out = malloc(pw*sizeof(int *));

    for(int i=0;i<pw;i++)
    {
        out[i] = malloc(pw*sizeof(int));
        for(int j=0;j<pw;j++)
        {
            out[i][j]=0;
        }
    }
    int x = pow(2,k-1);
    int y = pow(2,k-1);
    int i=0;

    while (*seq != '\0')
    {
        char ch = *seq;
        x = x/2;
        y = y/2;

        if(ch == order[2] || ch == order[3])
            x = x + pow(2,k-1);

        if(ch == order[0] || ch == order[3])
            y = y + pow(2,k-1);

        if(i>=k-1)
            out[y][x] = out[y][x]+1;
        seq++;
        i++;
    }
    return out;
}
```