Table of Contents

=================

- The stages of the processing

- Functions and packages to use

- How to feed the input and get the output

- The input data format

The stages of the processing

============================

The overall stages of the algorithm is as follows (for more details refer to "MCP: a Multi-Component learning machine to Predict protein secondary structure"):

1- Train SVM with edit kernel and get the prediction vector.

2- Apply filtration on the SVM prediction vector.

3- Compute the initial fuzzy membership values for the input data of fuzzy KNN.

4- Apply fuzzy KNN on the input data and get the prediction vector.

5- Apply filtration on KNN's prediction vector.

6- Pass through the SVM and fuzzy KNN prediction vectors to the fusion module in order to aggregate their results.

7- Apply filtration on the output of fusion.

Functions and packages to use

=============================

To follow the above stages using the current MCP available codes, follow these steps:

-Use the "libsvm-string-3.23\_1" package (or later versions) to train and test SVM with edit kernel. For details on how to use the package and format the data refer to "". Your labels must be this form:

‘1’ for ‘H’ class.

‘2’ for ‘E’ class.

‘3’ for ‘C’ class.

-Use the "filtering.m" procedure to filter out the SVM prediction vector. The input vector should be substituted with 'svmFinalLab.mat'.

-To perform steps 3 to 7, run "main.m".

How to feed the input and get the output

============================================

The main.m should receive the three following inputs:

* The filtered output of SVM should be substituted with 'svmLab1.mat'.
* The string train data should be substituted with 'lable1.mat'.
* The char label vector should be substituted with 'lable1.mat'.

The main.m generates these outputs:

* ‘finalLab.mat’ 🡪 An N by 1 vector of fuzzy KNN predicted labels.
* ‘finalLab1.mat’ 🡪 An N by 1 vector of mcp1 predicted labels.
* ‘finalLab2.mat’ 🡪 An N by 1 vector of mcp2 predicted labels.
* ‘finalLab3.mat’ 🡪 An N by 1 vector of mcp3 predicted labels.
* ‘finalLab4.mat’ 🡪 An N by 1 vector of mcp4 predicted labels.
* ‘finalLab5.mat’ 🡪 An N by 1 vector of mcp5 predicted labels.
* ‘finalAcc.mat’ 🡪 An 6 by k matrix of final accuracy of fusions before the final filtration. The first row associates with each fold of fuzzy KNN and rows 2 to 6 associate with the final accuracy of mcp1 to mcp5 before the final filteration. K is the number of folds (for k-fold cross validation).

\* In our fuzzy KNN algorithm some initial fuzzy membership values are calculated based on the existing data, which are to be compared to the entrance test instance. The stage of computing these initial values is considered a training phase. That’s why we have employed cross validation in our fuzzy KNN module.

The input data format

=====================

Input the data as a char matrix of size N by M where N is the number of data sequences and M is the length of the strings.

e.g: METLNDNLKVIEKADNA GEQGSNMECPNNLCCSQ

DIKCGSQSGGKLCPNNLInput the data labels as a character vector of size N by 1. The class labels for the problem of “protein secondary structure prediction” must be as follows:

H🡪 ‘1’

E🡪 ‘2’

C 🡪 ‘3’

For any questions and comments, please email Leila.khalatbari@gmail.com