Neural Network Theory and Applications
Homework Assignment 3

March 27, 2017
Due at April 5, 2017

This homework requires you to implement multi-class SVM classification based on the LibSVM package and solve a practical multi-class classification problem. The LibSVM package can be freely downloaded from http://www.csie.ntu.edu.tw/∼cjlin/libsvm/.

Requirement

1. Implement traditional one-versus-one and one-versus-rest task decomposition methods to solve a multi-class problem mentioned below.
2. Implement part-versus-part task decomposition method to solve the same multi-class problem.
3. Use two different kernel functions, namely linear and RBF, in all your classifiers.
4. Compare the advantages and disadvantages of these three task decomposition methods.

Dataset

The dataset (train.txt, test.txt) contains protein sequences from 12 subcellular locations. 20 dimensions stand for 20 amino acid composition of the protein sequences.

No. of proteins: 7579 (6065 training samples, 1514 test samples).
No. of classes: 12 ($0 \sim 11$).
The data file format:

label	dim1:value	dim2:value	...	dim20:value
0	1 : 0.095861	2 : 0.010893	...	20 : 0.032680

Description of LibSVM

In this homework, you should only use two files of LibSVM, svm.h and svm.cpp. And three interface functions listed below:

1. **struct svm model *svm train(const struct svm problem *prob, const struct svm parameter *param)**;

2. **void svm predict values(const struct svm model *model, const struct svm node *x, double* dec values)**;
   Call this function to get the decision values of prediction, instead of class labels. You need to call it to construct one-versus-others and part-versus-part classifier.

3. **double svm predict(const struct svm model *model, const struct svm node *x)**;
   Call this function to get the class labels of prediction. It worth mentioning that LibSVM has implemented one-versus-one multi-class classifiers. However, in this homework, you should construct one-versus-one from only binary classifier, without directly use the code of LibSVM.

To call these three functions, you should also be familiar with four structs, svm problem, svm model, svm parameter and struct svm node, all of which can be found in svm.h and svm.cpp.