Chris LeBailly

In this project I created a program to study an order-n Markov model. I took one set of sequence data to train the model and then used another set of data to test the model. I then computed the coding cost of the sequences in the test file and computed the cost per character and cost per sequence.

The program was broken into two parts. The first program, count-kmers, would take input from stdin and count the number of (n+1)-mers for an order-n model. These counts were outputted to stdout. The second program, coding-cost, took these kmer-count results in stdin along with a fasta file to test. In both programs the user specified the desired alphabet. The user could also specify multiple pseudo values to test in the coding-cost program.

The coding-cost program had several components. First it would create a table (which was implemented as a dictionary of dictionaries). It had all possible n-mers as contexts for keys (include start characters) in the outer dictionary and all letters plus the stop character as keys for the inner dictionary. The values of the inner dictionary started with just the pseudo value. Next the counts from count-kmers were added to this table. After this, the table was normalized and logs were taken to create a log-probability table. This table was then used to compute the coding cost of the fasta test file. The total number of characters (not including start and stop characters) and sequences were counted to compute the cost per character and cost per sequence.

I ran my program on the two sets of protein data. While testing my program I did train and test on the same data set, but these results are less interesting since the cost per character will always improve as we increase the order. Hence only the results for training on one set and testing on the other are shown. Using the alphabet ACDEFGHIKLMNPQRSTVWY, I had the program test the pseudo values 1e-10, 0.01, 0.1, 0.2, 0.5, 1, 2, 5, 10, and 100. The program then picked the pseudo value that minimized the cost per character and printed the results for that pseudo value. The results show that while increasing the order does improve the coding cost per character of the model, the gains are not significant.

Table 1: Training on Python2_f10_1.seqs, testing on Python2_f10_2.seqs

Order	0	1	2
Pseudo Value	1e-10	1e-10	2.0
Cost/char	4.22597	4.20535	4.19190

Table 2: Training on Python2_f10_2.seqs, testing on Python2_f10_1.seqs

Order	0	1	2
Pseudo Value	100.0	2	2.0
Cost/char	4.22803	4.20802	4.19307