| | |
|---|---|
| University of Wisconsin-Madison | Spring 2019 |
| BMI/CS 776: Advanced Bioinformatics | Homework #1 |
| Prof. Colin Dewey | Due: Thu, Feb 21, 2019 11:59 PM |

**Assignment Goals**

- Gain deeper familiarity with MEME for discovering motifs within biological sequences.
- Gain deeper familiarity with Gibbs sampling for the motif discovery task.
- Understand parameter estimation of motif models that incorporate prior knowledge

**Instructions**

- To submit your assignment, log in to the biostat server **`mi1.biostat.wisc.edu`** or **`mi2.biostat.wisc.edu`** using your BMI (biostat) username and password.
- Copy all relevant files to the directory **`/u/medinfo/handin/bmi776/hw1/<USERNAME>`** where **`<USERNAME>`** is your BMI (biostat) username. Submit all of your Python source code and test that it runs on the biostat server.
- For the rest of the assignment, compile all of your answers in a single file and submit it as **`solution.pdf`**.
- Write the number of late days you used at the top of **`solution.pdf`.**
- For the written portions of the assignment, show your work for partial credit.

**Part 1: MEME Implementation**

Write a program, **`learn_motif.py`**, that takes as input a set of DNA sequences and an integer width **`W`** and learns an OOPS model for a motif of width **`W`**.

Implement the EM algorithm from MEME to learn the motif. You should calculate the log likelihood, $\log P(X|\theta)$, after each iteration and stop the algorithm when the change in this value is less than a small fixed threshold. Use the exhaustive subsequence approach described during lecture (slide 32) to choose the starting point for EM with $\pi = 0.7$. Use pseudocounts of 1 ($d_{c,k} = 1$) when estimating the model parameters.

Your program should be callable from the command line as follows:

```
python learn_motif.py \
      -w W \
      --model=<model> \
      --positions=<positions> \
      --subseqs=<subsequences> \
      <sequences>
```

where

- **<sequences>** is a text file containing one DNA sequence per line. You may assume that the sequences will all have the same length.
- **W** is the width of the motif to learn.
- **<model>** is the name of the text file into which the program will output the learned motif model (i.e., the probabilities for each nucleotide in each column) in a tab-delimited format with the background frequencies in the first column.
- **<positions>** is the name of the text file into which the program will output the predicted starting location of the motif in each sequence one per line. Use 0 for the first position in a sequence.
- **<subsequences>** is the name of the text file into which the program will output the subsequences corresponding to the motif occurrence in each sequence one per line.

Input files, the template **learn_motif.py** with argument parsing code, and example output can be downloaded from **https://www.biostat.wisc.edu/bmi776/hw/hw1_files.zip**

To test your program, you may use the file **hw1_example1.txt** to discover the width 6 consensus motif **TAATCC** and the file **hw1_example2.txt** to discover the width 11 consensus motif **ATGACTCAGCA**. Sample output files for **hw1_example1.txt** and **hw1_example2.txt** are provided in **hw1_files.zip**.

Your program will be evaluated on these example inputs, the sequences in Part 2, and additional datasets that will be kept private.

**Part 2: Motif Discovery**

You will now run your MEME implementation to see how motif finding may be used in a biological study. Suppose the input sequences in the file **hw1_hidden.txt** come from the promoter regions of 100 yeast genes that are differentially expressed in a stress response experiment. We would like to use motif finding to learn more about a candidate regulator (transcription factor) of this expression response.

Use your **learn_motif.py** program to discover the motif of width 8 hidden in the sequences in **hw1_hidden.txt**.

Test that your program runs on the biostat server by running

```
python learn_motif.py –w 8 hw1_hidden.txt
```

from your **handin** directory. Leave the output files in the **handin** directory.

**Part 3: Sequence Logo**

Construct a sequence logo for the predicted motif sequences from Part 2 by using the WebLogo application (http://weblogo.threeplusone.com/create.cgi). Use the contents of the file **subseqs.txt** as input to the WebLogo application. Use PDF as the output format and select Logo-size as large. Save the logo as **logo.pdf** and submit it in your handin directory.

It is also easy to create sequence logos in Python. Instead of using the web server, you can optionally install the **weblogo** package (https://pypi.python.org/pypi/weblogo) with the command:

```
pip install weblogo
```

or run **weblogo** on the biostat server. After installing, you can generate the logo with the command:

```
weblogo -s large -F pdf < subseqs.txt > logo.pdf
```

This requires having **weblogo** on your PATH as in HW0. There is also a Python API to use **weblogo** programmatically, but it is not documented.

**Part 4: Transcription Factor Search**

Search the JASPAR transcription factor binding profile database (http://jaspar.genereg.net/align) using the PWM that you learned in Part 2. Use the contents of the **model.txt** file for this search. Delete the first (background) column, leaving the nucleotide columns, and paste the PWM into the "Please input a custom matrix" field. Select the CORE collection, the fungi Tax group, and the latest versions (non-redundant) in the remaining input fields. Then, click the Align button.

What is the name of the top-ranked yeast transcription factor that binds to this motif? Are there differences between the logo you generated in Part 3 and the JASPAR logo for this transcription factor?

Search the *Saccharomyces* Genome Database (www.yeastgenome.org/) for this transcription factor, entering its name into the search field. Based on the overview description, what cellular responses is it involved in?

**Part 5: Gibbs sampling and parameter estimation with prior knowledge**

Suppose that you are running the Gibbs sampling algorithm for motif finding on a set of 10 input sequences with a motif width of 4 nucleotides. Further, suppose that the current state of the Gibbs sampler is as shown below, with the motif occurrences indicated by the underlined nucleotides:

```
 1: CATGTGAA
 2: CAGCAGGG
 3: ACCTCTTC
 4: CAGACATG
 5: ACCTATCG
 6: GCGGCAGT
 7: GTGTAGTT
 8: CCAGGAAG
 9: ATGACCGG
10: GGATAGTA
```

**(A)** Suppose that in the next iteration of the sampler, sequence **5** is picked and the motif position in that sequence is to be resampled. Compute the estimates of the parameters, $p$, that are calculated in the predictive update step, given that sequence **5** has been selected. Use a pseudocount of 1 for your parameter estimates and assume a standard PWM model that does not incorporate any prior knowledge. Be sure to give the parameter estimates for both the PWM and the background.

**(B)** Given the parameter estimates you computed in (A), compute the probability that position 2 will be selected as the start of the motif occurrence in sequence **5** (which corresponds to a motif occurrence of CCTA). Show your work.

**(C)** Suppose instead that we are assuming a palindromic motif model. Repeat part (A), but with a palindromic motif model instead of a standard (non-palindromic) PWM model.

**(D)** Recall that the four nucleotides can be partitioned into two subsets of chemically similar nucleotides: the *purines* (A and G) and the *pyrimidines* (C and T). We could incorporate this knowledge into our motif model by using a Dirichlet mixture model prior with two components, one for purines and one for pyrimidines. Suppose that the parameters for the two Dirichlet components are those given in the table below and that we assume a uniform prior distribution over the two components. Repeat part (A), but with estimates calculated using this Dirichlet mixture prior (assume a non-palindromic model).

| | $j = 1$ (purine) | $j = 2$ (pyrimidine) |
|---|---|---|
| $\alpha_A^{(j)}$ | 4 | 1 |
| $\alpha_C^{(j)}$ | 1 | 4 |
| $\alpha_G^{(j)}$ | 4 | 1 |
| $\alpha_T^{(j)}$ | 1 | 4 |