CS 262 Tuesday, February 9, 2016
Computational Genomics Handout #3

Homework #3

CRFs and Sequence Assembly

Due at the beginning of class on Tuesday, February 23

Collaboration is allowed in groups of at most four students, but you must submit separate writeups. Please write the names of all your collaborators on your solutions. You are not allowed to copy group work. If you are working alone, we will drop the problem with the lowest score. If you submit your solutions after the submission deadline, you must write the date and time of submission on your writeup. Under no circumstances will a homework be accepted more than three days after its due date.
s

1. **Problem 1** (18 points) **Short Answers**

   Please state whether each of the following statements is true or false and provide a short explanation (two sentences or so).

   a- (2 points) One of the key differences between HMMs and CRFs is that HMMs model $P(\pi, x)$ while CRFs model $P(\pi|x)$ where π is the sequence of hidden states and x the sequence of observed ones.

   b- (2 points) The human polymorphism rate increases with time, since the human population size increases.

   c- (2 points) The heterozygosity of a population can only increase with time.

   d- (2 points) Wild mice are expected to have a higher heterozygosity than humans.

   e- (2 points) All human males inherited their Y chromosome from their father, which means that all Y chromosomes of all humans can be traced back to a single common ancestor. Therefore all human Y chromosomes are identical.

   f- (2 points) Compared to Sanger sequencing, next generation sequencing provides longer reads with lower error rates.

   g- (2 points) With the current sequencing technologies, it is impossible to disambiguate repeats whose length is longer than the read size.

   h- (2 points) Microsatellite repeats are not a big problem when sequencing a genome, since the repeated sequence is usually shorter than the length of the sequencing reads.

   i- (2 points) Assume you attempt to assemble the genomes of organisms A and B. A's genome has been covered at 10x coverage while B's genome has been covered at 5x coverage. It follows that A's genome will always be easier to assemble.

2. **Problem 2** (20 points) **Conditional Random Fileds**

We saw in lecture that in an HMM, the joint probability of a sequence $x = x_1 \dots x_L$ of length L and a parse $\pi = \pi_1 \dots \pi_L$ can be written as

$$p(x, \pi) = \exp \sum_{j=1}^{n} \log(\theta_j) F_j(x, \pi)$$

where n is the number of parameters (starting, emission, and transition probabilities) of the model, θ_j is the j^{th} parameter, and $F_j(x, \pi)$ is a count of how many times that parameter is used (transition is taken, emission is used) in the parse π of sequence x. The conditional probability $P(\pi|x)$ is therefore:

$$p(\pi|x) = \frac{\exp \sum_{j=1}^{n} \log(\theta_j) F_j(x, \pi)}{\sum_{\pi'} \exp \sum_{j=1}^{n} \log(\theta_j) F_j(x, \pi')}.$$

In this problem you will explore the CRF formulation for modeling the generation of sequence x and parse π. We will assume that our CRF model has N states (similar to the states of an HMM) and K features, $f_1, \dots, f_K$. Each feature f_j is a function $f_j(\pi_i, \pi_{i-1}, i, x)$ of the current state, the previous state, the current position in the sequence, and the entire sequence, and is scaled by a weight w_j. We will further assume that f_j is binary (i.e. $f_j \in \{0, 1\}$) and can be computed in constant time. We define

$$F_j(x, \pi) = \sum_{i=1}^{L} f_j(\pi_i, \pi_{i-1}, i, x)$$

Similar to the HMM case, $F_j(x, \pi)$ can be thought of as the total contribution of the feature f_j in the parse π of x. For completeness, let π_0 be a dummy start state of the CRF, so that f_j is defined for $i = 1$. The conditional probability $P(\pi|x)$ is then given by:

$$P(\pi|x) = \frac{\text{score}(\pi, x)}{Z(x)}$$

where

$$\text{score}(\pi, x) = \exp \sum_{j=1}^{K} w_j F_j(x, \pi)$$

and

$$Z(x) = \sum_{\pi'} \exp \sum_{j=1}^{K} w_j F_j(x, \pi').$$

$Z(x)$ is the "partition function", which ensures that $P(\pi|x)$ is a proper probability distribution. $Z(x)$ is the sum of the scores over all possible parses. Recall that a parse is a sequence of states; since each π_i can be any of N states there are obviously N^L possible parses for a given sequence x.

a- (2 points) CRF and HMM are very similar in that they are both generative models. Is this statement correct? why or why not?

b- (2 points) From CRF formulation, we find that CRF needs a normalization term, i.e, $Z(x)$. Does HMM needs such a normalization term too? why or why not?

c- (4 pionts) Please define the feature functions $f_j(\pi_i, \pi_{i-1}, i, x)$ and the weights w_j so that CRF is equivalent to HMM.

d- (6 points) Suppose we have exactly one sequence x and a true parse π and we would like to set the weights w_j such that $P(\pi|x)$ is maximized. In practice, this is done using numerical optimization methods such as gradient descent. As with HMMs, it is normally more convenient to work in log space and instead maximize $T = \log P(\pi|x)$. Prove that

$$\frac{\delta T}{\delta w_j} = F_j(x, \pi) - \frac{\sum_{i=1}^{L} \sum_{\pi'} f_j(\pi', \pi'_{i-1}, i, x)\text{score}(\pi', x)}{Z(x)}.$$

e- (2 points) If we set the derivative $\frac{\delta T}{\delta w_j}$ to zeros, we will find $F_j(x, \pi) = \frac{\sum_{i=1}^{L} \sum_{\pi'} f_j(\pi', \pi'_{i-1}, i, x)\text{score}(\pi', x)}{Z(x)}$. What does this equation imply?

f- (4 points) Gradient descent algorithm as in (d) will often result in a model that cannot generalize well to new data. Why? Also please propose some methods to prevent such phenomenon(please also include explanations why your proposed methods would work).

3. **Problem 3** (15 points) **Modeling the Sequencing Process**

We would like to sequence the human genome, of length G, using a whole-genome shotgun approach. We shear many copies of the genome into N pieces, all of equal length L, distributed randomly and uniformly from across the genome. The sequencing depth of coverage is defined as $C = NL/G$. You may assume that the length of the genome G is much greater than the length of a read fragment L, in particular this means you may ignore end effects and, for example, assume that the left ends of the reads are distributed in the range $[0, G]$. For the following problems, it may help to approximate the selection of read locations as a Poisson process along the length of the genome. You are welcome to review the Lander-Waterman model at:

`http:// www.genetics.wustl.edu/bio5488/lecture_notes_2005/Lander.htm.`

(If the page doesn't display properly, you will have to switch your browsers encoding.) However, please show all steps of your derivations and justify approximation assumptions you make.

a- Answer the following questions

i- (2 points) What is the expected proportion of the bases in the genome covered by at least one read, in terms of C ? (Hint: your initial derivation may involve N, L, and G, but using the fact that $L \ll G$ you should be able to express the nal answer using only C .)

ii- (2 points) If we would like our expected proportion of genome bases covered to be 99%, what depth of coverage C do we need?

iii- (2 points) What about if we want an expected 99.9% to be covered?

b- (5 points) We would like to assemble these sequence reads into contigs, or contiguous sequence. Suppose that whenever two reads overlap or even touch (i.e. when one reads left end starts at position x and another starts at position x+L), we can merge these two reads into a longer contig. This way, contigs extend through contiguous positions in the genome that are covered by at least one read. Compute the approximate number of contigs and average contig size in our assembly, in terms of N, C , and G. Hint: one way of deriving this is the following:

i- If you scan the genome from left to right, assume the incidence of read start locations can be modeled as a Poisson process, which is memoryless.

ii- For a particular read i, compute the probability that read $i+1$ will start at a position greater than L bases from the start of read i.

iii- Using the above result, compute the expected number of gaps.

iv- Now, explain approximately what the number of contigs and what the average contig size should be.

c- (4 points) Suppose that reads must overlap each other by at least K bases before the overlap can be detected and they can be merged together. Again, using the results from the previous problem, compute the approximate number of contigs and average contig size of our assembly.

4. **Problem 4** (15 points) **De-novo Assembly - a Hamiltonian Approach**

In this problem, we would like to assemble our N sequence reads of length L from the previous problem into large contigs. Let R be the set of reads, and we construct the directed overlap multigraph of R, denoted $G(R)$. The nodes in the graph are the reads themselves, and we have an edge $E(a, b)$ from node a to node b with weight w whenever the w-letter suffix of a is the same as the w-letter prefix of b, in other words, read b extends read a to the right with an overlap of w. Note that this is a multigraph and there may be more than one edge between two nodes if there is more than one possible overlap alignment.

a. (5 points) Suppose we are only interested in edges with weight at least W, i.e. the reads overlap by at least W bases. Your goal is to construct $G(R)$ restricted to these edges that does not need to perform all possible $O(N^2)$ alignments between pairs of reads.

   i. Define data structures that you will use to design the algorithm.

   ii. Describe the algorithm and provide the time and space complexity.

b. (4 points) A Hamiltonian path through $G(R)$, or a path that visits each node exactly once, represents an assembly of the reads into a supersequence. Minimizing the length of this supersequence is equivalent to finding the Hamiltonian path with maximum weight, where the weight of a path is defined as the sum of the weights of the edges that make up the path. Unfortunately, the simpler task of deciding whether or not a graph has a Hamiltonian path is in general NP-complete.

   Consider the following greedy approach to maximizing the weights: First, start the path by picking the heaviest edge in the entire graph. Then, incrementally add the next edge to the path by taking all the remaining edges that start at the last node in our path and lead to a node not already in our path, and picking a maximal weight edge. Assume that our graph has been made complete by adding in zero-weight edges between nodes that have no edge. Show that this greedy algorithm does not always find the optimal, maximal-weight path. (Hint: you should be able to construct a very simple example with perhaps 3 or 4 reads such that this algorithm finds a supersequence that is longer than the shortest possible supersequence).

   c- Consider the following set of sequence reads R:

```
A: stersInTheSea?
B: stersLiveWithThese
C: TheseOysters
D: HowDoYouKnowIfThese
E: TheseLobsters
```

i. (2 points) Draw the directed overlap multigraph $G(R)$ for this set of reads, including edge weights.

ii. (2 points) Write out all possible single-contig assemblies of this read data, i.e. supersequences formed by Hamiltonian paths through the graph in part i., and give their total path weights.

iii. (2 points) Introduce only a new read, such that there is only a unique assembly. What is the minimum length required for such a read?

5. **Problem 5** (18 points) **De-novo Assembly - a Eularian Approach**

In the previous problem, we used the directed overlap multigraph for assembly. For the following, we will try a different approach to assemble the sequences - using the ***De Bruijn*** graph. An n-dimensional De Bruijn Graph on 4 symbols (i.e, A,C,G,T) is a directed graph representing overlaps between reads, where there are 4^n vertices, consisting all length-n sequences of A,C,G,T. If the length-n sequence at one node can be expressed as that at another node by shifting all its symbols by one place to the left and adding a new symbol at the end of this node, then the latter has a directed edge to the former node. For instance, suppose $n = 4$, an edge would be assigned as

$$(TAAA) \xrightarrow{TAAAC} (AAAC).$$

Here is the key of constructing a De Bruijn graph for fragment assembly. For a given read of length L, there are $L - k + 1$ substrings of length k which we refer to as k-mers. We will use a $(k-1)$-dimensional De Bruijn graph where the nodes are length-$(k-1)$ substrings in this read.

Now as we consider more reads, we keep incrementing the edge weights in the De Bruijn graph. For instance, if there are only two reads $GTAAAC$ and $TAAACC$, and given $k = 5$, the De Bruijn graph that only includes these two reads would be

$$(GTAA) \xrightarrow[1]{GTAAA} (TAAA) \xrightarrow[2]{TAAAC} (AAAC) \xrightarrow[1]{AAACC} AACC.$$

In order to assemble the fragments, we only need to find a path that visits each edge exactly once (an Eulerian Path). So in this example, we would return $GTAAACC$ as the assembly by going along the path and picking up one letter for each additional node the path visits.

a. Finding an Eulerian Path is an easier problem than finding a Hamiltonian path. Consider a genome of length N, and suppose, in an ideal setting, that reads of length L are sampled without errors. Moreover, assume that every position in the genome has a read that starts there. Furthermore, let us assume that all repeats are of length less than k in the genome.

   i. (5 points) Show that in the $(k-1)$-dimensional De Bruijn graph (with $N > L \geq k$) constructed from ideal reads, there must exist an Eulerian path.

   ii. (Optional: 5 bonus points) Provide a polynomial time algorithm that finds such the Eulerian path in such a graph.

b. Noise is one of the main challenges in assembly in real applications. In which case, the De Bruijn graph may not provide a simple solution. In this problem, you will try to correct for errors. Assume that the reads below come from the same strand and there are only substitution errors.

```
TTCAGC
ATTCAG
CTGTAG
AGCTGT
CAGCTG
TTCTGC
GCTGTA
TCAGCT
CTGTAT
TGTAGC
```

i. (3 points) Construct the De Bruijn graph.

ii. (3 points) Identify tips and bubbles in this graph. These could be caused by errors.

iii. (6 points) Provide one single assembly and justify which reads you think have errors. (Hint: you can form "chains" by merging nodes in the graph that are on a simple path.)

6. **Problem 6** (15 points) **Sequence Assembly**

Suppose we have constructed a prefix-suffix graph, such a De Bruijn graph, where the vertices are the reads $r_1, \ldots, r_n$, and edges are $r_i r_j$ if a suffix of r_i is a prefix of r_j. Suppose the number of edges in the graph is m. Furthermore, assume that we have positive integer cost functions c and d over the edges of the graph

a. (5 points) Provide an $O(m+n)$ (or faster) algorithm to find all the bubbles in the graph.

b. (5 points) A path is sequence of edges in the same direction which connect a sequence of vertices. Suppose we want to find the number of paths that start with read s and end in read t, such that the cost of the path, calculated using c, is between α and β. That is, for a path $v_1, v_2, \ldots v_k$, where $v_1 = s$ and $v_k = t$, we want $\alpha < \sum_{i=1}^{k-1} c(v_i v_{i+1}) \leq \beta$. Provide an $O(\beta m)$ (or faster) algorithm to find the number of such paths.

c. (5 points) Suppose among all paths that satisfy the constraint in (b), we want to find the path that minimizes the cost over d, that is $\sum_{i=1}^{k-1} d(v_i v_{i+1})$ is minimized. Provide an $O(\beta m)$ (or faster) algorithm to find such path.