Mash: fast genome and metagenome distance estimation using MinHash

Dreycey Albin (da39)

April 17, 2019

Scientific question

The scientific question and goal for this paper focuses on *using the MinHash technique for comparing sequences without direct alignement.* While the MinHash algorithm has been used for other applications in bioinformatics, it has not been used as an alignment free method for comparing two seperate DNA sequences.

Hypothesis tested

The hypothesis tested in this research is that *using locality sensitive hashing (LSH) to turn DNA sequences, which may be tremendously smaller than the original sequence, may allow an alignment free method to compare sequences using MinHash.* This hypothesis is tested by using Mash to (1) Clustering all of the genomes in Refseq, (2) genome identification from reads, and (3) clustering massive metagenomic datasets.

Methods/Results

The results and methods for the paper can be subdivided into 2 sections: (1) The technique for the program, Mash; and (2) the test results for using Mash on real data.

0.1 Methods behind Mash

The method for mash works in two steps: converting a sequence to a *sketch*, followed by using this sketch in the MinHash algorithm to get a close approximation for the jaccard index, called a *dist.* For each of these outputs, there is a corresponding function for the algorithm, named the *sketch function* and the *dist function*, respectively.

The sketch function takes k-mers based on a window size and returns the smallest hashes output (refered to as the bottom sketches). The ability for the MinHash technique to distinguish between sketches may be defined by both the siaze of the input genome and the sketch size. After obtaining the sketches, a two-stage minhash and a bloom filter are used to remove uneeded k-mers (assuming reduncancy). Overall, the sketch holds the s smallest hashes from the kmers, as long as the counter for the sketch is also above the c-1 threshold.

The distance function works by implementing the MinHash technique. This essentially allows an approximation to the Jaccard index, between two sets. Using the bottom hashes saved in the sketch, the jaccard index is approximated by comparing all of the different reads. For x shared hashes and processing s' hashes:

$$j = (x/s'),\ with\ an\ error\ bound\ equal\ to\ E = O(1/sqrt(s))$$

Overall, this allows for the calculation of similiarity, called the mash distance:

$$D = -(1/k)ln(2j/(1+j))\ where\ the\ number\ of\ shared\ kmers\ is:\ w/n = 2j/(1+j)$$

0.2 Mash on Real Data

0.2.1 Clustering all of the genomes in Refseq

To show the usefulness of Mash, the author's clustered all of the genomes in RefSeq release 70, compressing the information from 674 GB into 93 MB. The Mash function is able to show a good Average Nucleotide Identity between the output for the Mash program and a subset of bacteria to test against. The comparison with MASH was great for an ANI in the range between 90% and 100%. Using this clustering, the authors noticed that importance of picking sketches and kmer sizes. The sketch size will cause improved accuracy at the expense of increased disk space and time for calculating the distance between sketches in linear time. The kmer size will be a trade off, as always, between sensitivity and accuracy.

0.2.2 Genome identification from reads

The authors also showed the potential to search datasets. They were able to compute the distances for E. coli samples against the RefSeq database that they had made in house. It should be noted that this method actually performed better if the genomes were already assembled, verse just using the reads for the genomes. The discriminatory power of Mash was also shown, as Mash was able to differentiate between *B. cereus* and *B. anthracis* (ANI = 95 %). The authors next tested this on Zika data while streaming off of the Nanopore using the online algorithm for Mash, showing the ability for Mash to be used in real time, while the sequences are coming off of the sequencer.

0.2.3 Clustering massive metagenomic datasets

Two other previous programs that attempt to classify metagenomic samples are DSM and COMMET. DSM uses an exact Jaccard index using all of the kmers that appear more than twice for a given sample, and COMMET uses bloom filters to find a set of similiar reads, essentially presenting an approximation for the jaccard index. To show the unique ability for Mash to work in very fast running time using the MinHash algorithm, the authors used a large scale test using the output from the Human Microbiome project. When comparing the Mash clustering to that of read clustering, both look very similair with samples clustering according to body site. However the authors do note the problem with using simple kmers sets, as the batch effects from sequencing or other, disallowing for Mash to cluster. This can also be seen with trying to use Mash to cluster MetaHIT samples by health status.

Key implications of the results

Mash delivers many benefits, some of which may be to search massive DNA databases, or to look for sequence similiarity, an all vs all fashion, between two datasets. Potential next steps include using this to choose appropriate reference genomes, or extending the program to cluster similiar sketches to reduce the potential search space. The sketches may be merged using a bloom tree. Overall, Mash allows for a fast and efficient method to perform and alignment-free method to compare sequences.