Oral Microbiome Project Report

Bryan Chim

5/10/2014

It has long been known that the degradation of gingival and subgingival tissue fibers common to all periodontal diseases is caused by a cytotoxic microbial environment working in concert with an overly aggressive immune response. Diet, smoking, caries, the presence of other diseases (ie. diabetes mellitus, atherosclerosis) and especially poor dental hygiene are known contributors or correlates to the accumulation of a harmful bacterial/mycotic matrix at the gum-line. This microbial plaque may grow and calcify into a calculus (aka tartar) that is especially destructive to periodontal ligaments and the bony alveolar layer, resulting in inflamed gum tissue and diseased pockets between the teeth and gums.

The diagnosis of periodontal diseases, up until the advent of high-throughput sequencing technologies, has relied primarily on observing gingival tissue loss or detection of a few select pathogenic species. Recently16s rRNA high-throughput sequencing, annotation and characterization of oral microbiome samples have revealed that there are more than 700 microbial species present in the human oral biofilm, all in varying proportions depending on the type and stage of periodontal disease. *Instead of looking for the mere presence of certain individual species, we are finding it necessary to investigate the oral microbiome composition as a whole in order to more accurately assess periodontal disease states.* Our goal was to build a robust and interactive microbiome explorer that could accept large volumes of sample read and classification data, perform statistical analyses and generate plots on the concentration/distribution of oral microbiota.

Our application was built and organized around the data gathered and analyzed in a study performed by Ping Xu’s lab at the VCU Philips Institute of Oral and Craniofacial Molecular Biology. Over the course of about a year (2009-2010), the Xu lab collected oral samples from 92 patients with varying levels of chronic periodontitis. Paired samples were taken from each patient, one at a deep (diseased, > 5mm) and one at a shallow (healthy, < 3mm) site in the periodontal pockets. These 184 samples were purified, amplified, and sequenced at the V4-V6 hyper-variable16s rRNA regions (areas that are known for high genomic diversity across bacterial taxa). They used two-part model, False Discovery Rate and Hierarchical Trees Cluster Analysis to make comparisons between the microbial compositions in deep vs. shallow sites and, secondarily, to determine the influence of patient race, gender, frequency of smoking and caries.

Integral to our particular application is the Ribosomal Database Project Classifier (RDP Classifier). This is a Java-based Bayesian classification tool that assigns individual reads in a given sample to their most likely taxa for each taxonomy level, down to the genus level (if it meets the specified threshold). The RDP Classifier begins with a training set of ~8400 highly accurate 16s sequences, which are each grouped into individual genera via Bergey’s taxonomy. For each genus, it counts the occurrence of every possible 8 nucleotide word in its group of sequences and calculates P(W/G) – probability of word W being part of genus G – for every genus. When a multi-sequence FASTA file is given as input, the RDP classifier compares each sequence to the P(W/G) values derived from its training set, and then calculates P(S/G) – probability of sequence being part of genus G – for every genus. Using this algorithm, each sequence is assigned to the genus with the highest probability of being a genomic match. Finally, the RDP Classifier performs 100 cycles of bootstrapping for each read to calculate confidence scores for its assignments. A threshold can be applied during the output phase to filter out assignments that do not meet a satisfactory confidence score.

We began with the Xu lab’s base set of sequence files following sample separation by barcode: 184 FASTA files containing a total of 1824295 16s rRNA reads from the 92 patients. These FASTA files were each ran through the RDP classifier with a 0.5 confidence score threshold. For reference, the generic command that was used is:

java -Xmx1g -jar <path\_to\_classifier.jar> classify –c 0.5 <path to FASTA file> -o <output file>

This command generates an output file containing all the read names and IDs found in the FASTA input file and, for each read, a list of the taxonomies that most closely match up to it, down to the genus level if it meets the confidence score threshold.

Using the RDP Classifier outputs, we wrote a series of Python scripts to further extract and summarize the taxonomy data for each sample. Our first script generates what we call a “Read Assignment” file which lists:

SampleID MethodID ReadID TaxaName TaxaLevel Score

This script finds and extracts this information *for every taxa for each read in the sample* that has a score exceeding a user-defined threshold. This threshold corresponds with the MethodID (1 = 0.5 score threshold, 2 = 0.8 score threshold). If the taxa does not meet the threshold, it is not extracted.

Our second script generates a “Profile Summary” file which lists:

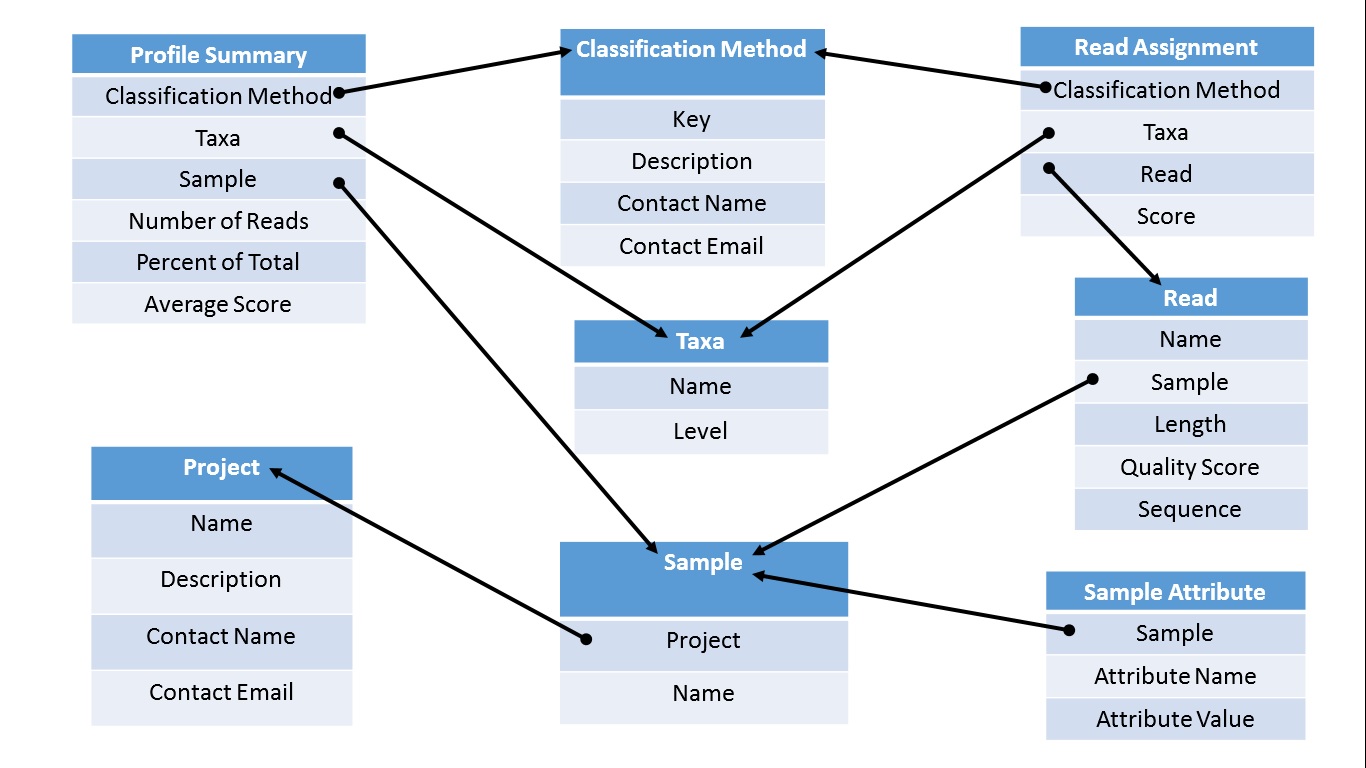
SampleID MethodID ReadID TaxaName TaxaLevel #Reads %Total AvgScore

This script works in tandem with the first script, and actively counts the number of occurrences and sums up the score for every Taxa that meets the threshold for that sample. After the file is parsed, the script calculates and outputs – *for each taxa found in the given sample –* its read count, the percentage of total reads in which it is found, and its average score

After initial testing, these commands and scripts were then integrated into a pipeline consisting of a single script that runs RDP Classifier on the entire directory of FASTA files and performs the extractions described above for every output file. This pipeline was run once with a threshold of 0.5 (Method 1) and again with a threshold of 0.8 (Method 2), yielding 368 Read Assignment files and 368 Profile Summary Files. A simple cat command in UNIX was then employed to merge all Read Assignment files, and another to merge all Profile Summary files. The final result was 2 massive files containing, respectively, the Read Assignments and Profile Summaries for every sample.

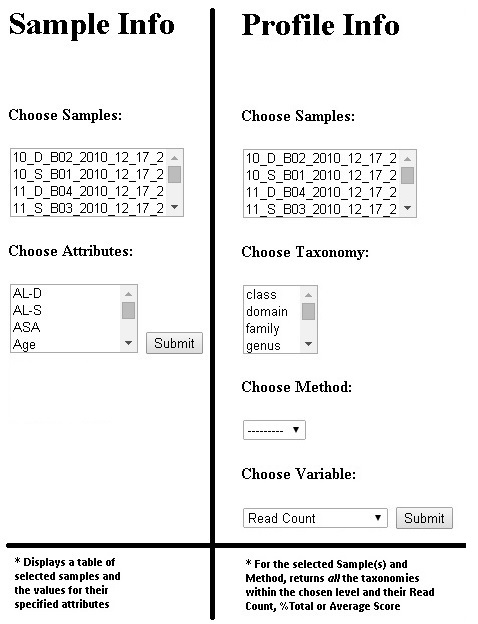
In addition to raw genomic and taxonomic data, we were also interested in a number of attributes that were specific to the preparation of the samples, as well as the patients that they were collected from. There were 34 such attributes that include but were not limited to: sample run date, sample collection date, sample barcode, sample depth, patient gender, patient age, patient race, frequency of smoking and risk level for caries. Unfortunately, due to the samples being collected at different times/locations and being sequenced in separate batches – this resulted in all the data being divided across several tabs in a large and disorganized Excel Spreadsheet. Mapping, merging and organizing all of these sample attributes required a series of Python scripts and UNIX commands that comprised another entire pipeline. In the end, our efforts yielded an output file that consists of, on each line, SampleID, Attribute and Attribute Value – a format that could easily be loaded into our SQL database.

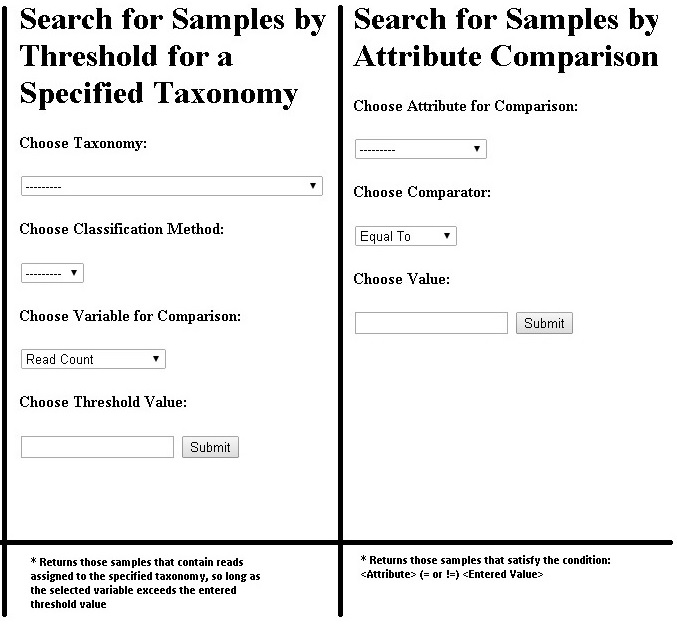
Once all of the data was extracted, formatted and prepared correctly, the next step was to load it all into our SQL database for manipulation in a Python-Django development environment. The diagram below describes all the relevant models derived from this data, as well as all the foreign key connections (in this case, the relationships are all one-to-many: **one** 🡨 **many**).



It should be evident that all the data required to construct these tables can be found in or derived from the Profile Summary, Read Assignment or Sample Attribute files that we had assembled, or parsed from the FASTA files themselves. For initial testing, we manually created and inputted our Project and Classification Method data. All the Sample names were extracted from the FASTA file names via UNIX commands, and all the Read data were extracted from the FASTA files via a series of regular expressions. Taxa names and levels were extracted from the appropriate columns of our Profile Summary File. Using *DictReader* in the *csv* python module, we wrote loadData scripts and loaded the data for each of the tables in this order: Project, Sample, Taxa, Classification Method, Sample Attribute, Read, Read Assignment and Profile Summary. By first loading in data for those tables that have no initial foreign key constraints, we were able to create those links later on by simply passing, matching and mapping via primary keys.

After data load-in was complete, we focused on creating several pages to display information based on search criteria and/or attribute comparison. Here are physical screenshots of each page, followed by their respective function and output underneath:





Each template and view makes extensive use of list read-ins via the ModelMultipleChoiceField forms in Django. These pages function as expected for the basic queries they were designed for but, due to time constraints, we were unable to implement plotting capabilities and statistical analyses for concentration, distribution or cross-sample comparisons. Future directions involve integrating Views logic with the statistical functionality of R and utilizing Javascript graphing modules in our templates for data visualization.

References

Ge, Xiuchun et al. (2013) *Oral Micriobiome of Deep and Shallow Dental Pockets in Chronic Periodontitis*. PLoS ONE 8: e65520.

Liu, Bo et al. (2012) *Deep Sequencing of the Oral Microbiome Reveals Signatures of Periodontal Disease*. PLoS ONE 7:e37919.

Sheth, Nihar (2014) *Bioinformatics for Metagenomics*. BNFO653 Lecture. Center for the Study of Biological Complexity, VCU.