


CS486C – Senior Capstone Design in Computer Science

Project Description

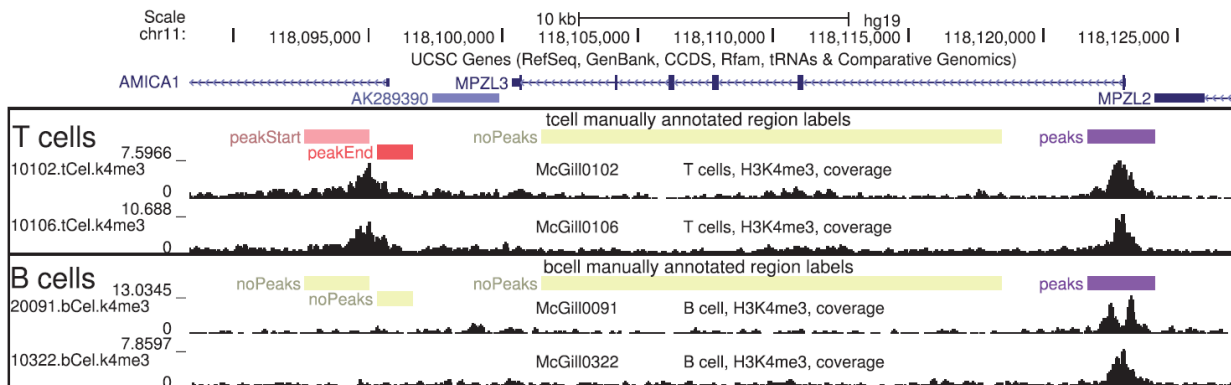
Project Title: PeakLearner, an interactive web app for machine learning in genomic data	
Sponsor Information:  School of Informatics, Computing, and Cyber Systems	Dr. Toby Dylan Hocking , Assistant Professor School of Informatics, Computing, and Cyber Systems toby.hocking@nau.edu Northern Arizona University

Project Overview:

Gene science has been perhaps the biggest medical advance of recent time, allowing unprecedented insights into the works of the living systems, including humans. A particularly promising area of application is in the health science, where genomic techniques can help us understand the genetic foundations of many diseases and health conditions. In particular, medical researchers routinely use genomic data experiments to analyze the genetic mechanisms of diseases such as cancer (i.e. which genes get turned on/off in cancer cells relative to normal cells); this could lead to new treatments, e.g., using gene therapy that prevent cancers from developing or progressing.

Although very promising, these techniques based on genomic analysis are not as simple as one might hope. Genomes have evolved over eons and therefore contain lots of “dead wood”, i.e., long non-coding sequences in the genome that no longer serve any function; distinguishing active coding regions from non-coding regions is an important step in analysis. To make matters more difficult every person’s genome differs slightly, meaning that one has no simple concrete pattern to look for that never varies. This is where machine learning could help!

ChIP-seq and ATAC-seq are high-throughput DNA sequencing experiments used to characterize active genomic regions in specific experimental samples and cell types. An important step in analyzing these data is detecting these active regions, which appear as "peaks" or long contiguous genomic regions with large numbers of aligned DNA sequences. In addition it is important to detect differences between samples. For example, in the four ChIP-seq data sets below (rows of black bars), there is a common peak in all four samples on the right, and a peak only in T cells on the left. An ideal peak detection algorithm would report a model which is interpretable in terms of similarities and differences between samples. The problem is that existing peak detection algorithms result in many errors (false positive peaks predicted where there are none in the data, and false negative lack of peaks predicted where there clearly are some in the data). Machine learning can be used to obtain more accurate peak detection models, but we need large databases of labels for training.



In my lab, we have developed a supervised machine learning concept for peak prediction that should be able to generate labels that indicate specific samples and regions with or without peaks. For example in the figure above, labels are drawn as colored rectangles:

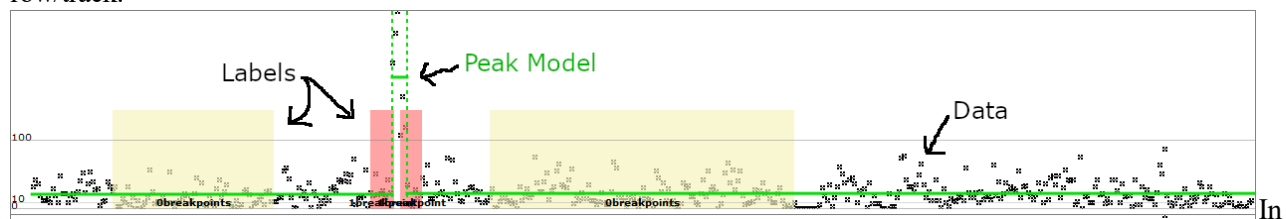
- **noPeaks** indicates a region in which there should be no predicted peaks.
- **peaks** indicates a region in which there should be at least one predicted peak.
- **peakStart/peakEnd** indicate that there should be exactly one predicted peak start/end.

Project details: a GUI-driven machine learning tool

Although we have developed a proof-of-concept prototype of our machine learning approach, it is currently implemented as [what? An R package?] that requires substantial technical and programming knowledge to utilize and produces output in a dense textual fashion. What is urgently needed to make our machine learning approach for peak analysis easier for genomic scientists to use is a clean, easy-to-learn graphical user interface (GUI) which allows viewing such data and interactively specifying the peak labels.

Whereas there are several existing software packages for viewing genomic data tracks such as ChIP-seq [2-3], none support interactive specification of the track-specific labels which are required for our machine learning approach. Specifically, we propose development of a prototype PeakLearner web app with features for (a) viewing genomic data, (b) adding/editing track-specific labels, and (c) using the labels to train a machine learning peak model, which is also displayed on the web app.

The ideal user interface would be similar to our previously proposed SegAnnDB system for interactive machine learning analysis of DNA copy number profiles [4]. An annotated screenshot is shown in the figure below: it displays a noisy data set (black), its labels (colored rectangles), and a predicted peak model (green) on the same row/track.



The Peak Learner tool proposed here would operate in a similar way: <explain>

Although detailed requirements will be developed by the Capstone team, some key functional capabilities for this software product will include:

- A public web site, <http://peaklearner.rc.nau.edu/> which serves as the main interface. You will need to ask ITS to set this up.
- Users should be able to authenticate and log in to the web site in order to create user-specific labels and peak models.
- Should provide a point and click interface for specifying new noisy data sets (via URLs of bigWig files) and projects/groups of related data sets.
- As soon as a new data set is uploaded, server sends peak detection jobs to NAU Monsoon compute cluster. The cluster should run the GFPOP algorithm with up-down constraints [5] in parallel across genomic regions and penalty values, resulting in a set of pre-computed peak models for each data set.
- Display of data, labels, and peak model for a given sample on the same track/row of the genome browser. point and click interface for adding/editing the labels for a given track. e.g. drag to create new label, click to change label type, then click Save to save to server. Click existing label to delete. (most of these features have already been implemented as a JBrowse plugin which you may need to edit a bit).
- After add/delete label, server retrieves a pre-computed peak model as quickly as possible, showing the new peak model on the genome browser track. The goal is instantaneous feedback, so the user can correct any issues with the currently displayed peak model.
- The web GUI provides Python and R export functions for downloading labels and peak models.
- A complete product will include Travis-CI/Selenium/PhantomJS module for headless browser testing

Hot tips: specific technologies to consider/investigate in the design process:

- JBrowse with plugin for simultaneous display of peak model, labels, and data.
<https://github.com/yf6666/PeakLearner-1.1>

- For computing the peak models on the Monsoon compute cluster, the code in the <https://github.com/tdhock/PeakSegDisk> and <https://github.com/tdhock/PeakSegPipeline> R packages can be used.
- For the database to store user labels and peak models, MySQL/Postgres were too slow for instantaneous feedback in the previous SegAnnDB project; recommend using the file system or a key/value store such as BerkeleyDB instead.

Knowledge, skills, and expertise required for this project:

- Will need to learn basics of genome analysis and machine learning techniques (enough to understand/adapt/apply our existing implementations)
- Back-end web server programming (e.g. Python Pyramid).
- Modern Web2.0 web application development: techniques, frameworks, languages.
- HPC/Cluster computing, i.e. launch remote jobs and retrieve results from Monsoon.
- NoSQL database design / programming.

Equipment Requirements:

- There should be no equipment or software required other than a development platform and software/tools freely available online.
- Access to Monsoon will be provided. Team will be required to take one hour training provided by the Monsoon team to enable access.

Software and other Deliverables:

- The web application GUI for peak learning outlined above, installed on a server specified by the client, and fully tested with real data.
- Full and complete documentation accessible from the web app including: Video documentation for end-users who will log on to the system and perform interactive labeling / peak detection analysis; as well as written documentation for end-users who use the web API to download labels and peak models.
- A strong as-built report detailing the design and implementation of the product in a complete, clear and professional manner. Should detail the internals of the code base, for the programmers who will add features in the next version PeakLearner.
- Complete professionally-documented codebase, delivered both as a repository in GitHub, and as a physical archive on a USB drive.

References:

- [1] Hocking et al. Optimizing ChIP-seq peak detectors using visual labels and supervised machine learning. Bioinformatics 2017, <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5408812/>
- [2] <https://jbrowse.org/>
- [3] <https://genome.ucsc.edu/goldenpath/help/trackDb/trackDbHub.html#aggregate>
- [4] Hocking et al. SegAnnDB: interactive web-based genomic segmentation. Bioinformatics 2014. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4029035/>
- [5] Hocking et al. Generalized Functional Pruning Optimal Partitioning (GFPOP) for Constrained Change-point Detection in Genomic Data. 2018, pre-print <https://arxiv.org/abs/1810.00117>