


CS486C – Senior Capstone Design in Computer Science

Project Description

Project Title: PeakLearner, an interactive web app for machine learning in genomic data	
Sponsor Information: 	Dr. Toby Dylan Hocking , Assistant Professor School of Informatics, Computing, and Cyber Systems Machine Learning Research Laboratory Email: toby.hocking@nau.edu Christopher Coffey High-Performance Computing Email: Chris.Coffey@nau.edu

Project Overview:

Machine learning lies at the heart of the renewed excitement about “AI” or “deep AI” systems, which have been steadily making their way out of the lab to revolutionize many real world tasks and industries. For example, machine learning has been extremely successful in recent years for tasks such as spam filtering (e.g. in Gmail), image classification (Detexify), recommendation systems (Amazon) and automatic translation (Google Translate). The NAU SICCS Machine Learning Research Laboratory develops new statistical models, optimization algorithms, interactive systems, and software for machine learning. In particular, a recent focus has been on the field of “supervised machine learning”, in which learning is shaped by algorithms that recognize patterns based on data sets with labeled examples.

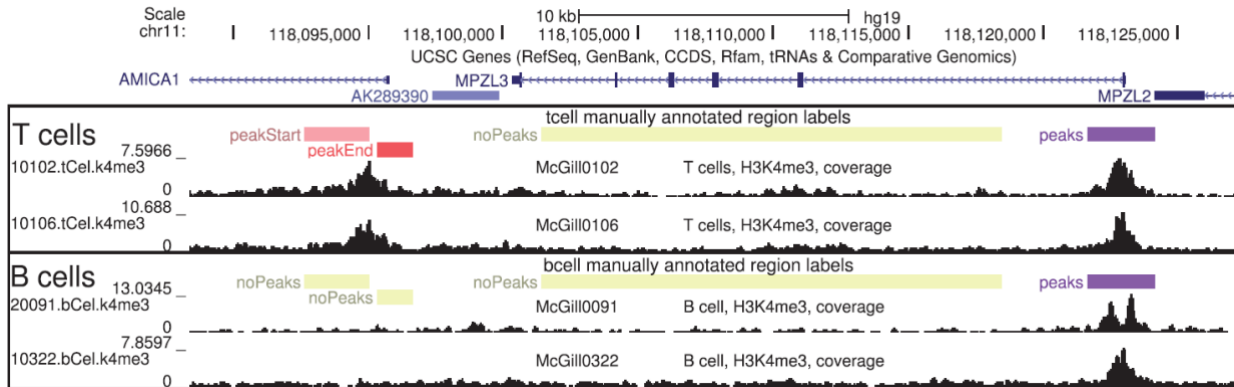
Genomics is another area that holds very high promise for the application of machine learning techniques, especially for challenges involving detection of “interesting” features in large genomics data sets. The advent of ever cheaper, higher-throughput sequencers has made possible completely new techniques and experiments. For instance, detecting cancer or other diseases used to be based on looking at biopsy samples under a microscope to detect cancerous cells. Of course, this requires that a biopsy be available, i.e., there must already be a visible suspicious mass...which is quite late in the cancer process. A much better approach would be to take a general sample of an area (e.g., tissue, swab, feces, etc.) and simply sequence the DNA of all cells present...and see if any known disease DNA is detected.

Solution Overview

This project involves implementing a machine learning web app to be used by genomic scientists and researchers for analyzing and recognizing peak patterns in epigenomic data, which are used as indicators in cancer and other diseases. Two examples are ChIP-seq and ATAC-seq, which 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.



We have proposed a supervised machine learning framework for peak prediction [1], which requires 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.

To make it easier for genomic scientists to use machine learning algorithms on these data, it would be useful to have a graphical user interface (GUI) which allows viewing such data and interactively specifying 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 machine learning.

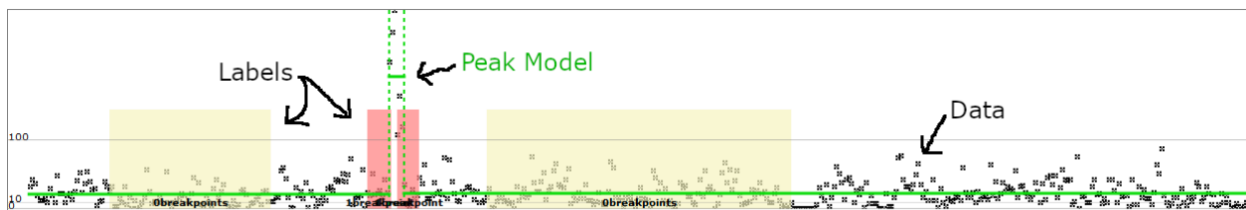
Project Details

The aim of this project is to develop a PeakLearner web app with features for:

- Graphically viewing genomic data uploaded by users
- Adding/editing track-specific labels within the GUI to annotate presence/absence of peaks in the sequence data
- Using the labels to train a machine learning peak model, the results of which are also displayed, superimposed on the graphical sequence view.

A successful capstone project will result in a public machine learning web app which will be used by a large community of genomic scientists, as well as machine learning researchers.

Our lab will be able to provide many of the key ideas needed by the team. For instance, 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.



We expect to work with the team to develop a clear and prioritized view of the features the PeakLearner web app will provide. As a starting point, we see the following as features required in a minimal viable product:

- A public web site, <http://peaklearner.rc.nau.edu/> which serves as the main interface. We will help the team work with ITS to set this up under the ownership/account of the client for easy transition of ownership at project end. If more convenient, the team may use their own development web server for their work during the year.
- Users should be able to log in to the web site in order to upload datasets, view them graphically, and create user-specific labels and peak models.
- 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 via 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.
- Peak detection jobs should be run on a SLURM compute cluster. At the beginning of the project students should set up their own SLURM cluster for development/testing. SLURM is the same software which is used on the NAU Monsoon HPC cluster, so eventually Monsoon can be used for production/deployment.
- 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.
- 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.
- Web API with Python and R clients for downloading labels and peak models.

Knowledge, skills, and expertise required for this project:

- Back-end web server programming (e.g. Python Pyramid).
- Front-end web interface programming (JavaScript).
- HPC/Cluster computing, i.e. launch jobs and retrieve results from a SLURM system such as Monsoon.
- NoSQL database design / programming.

Equipment Requirements:

- No special software or equipment should be required, beyond a development platform and freely available software packages.
- Specific software packages / technologies to consider using:
 - Apollo supports interactive annotation [6]; the JBrowse wigglehighlighter plugin supports displaying labels on top of genomic data [7]; UCSC genome browser supports displaying peak models on top of noisy data via multiWig containers [3]; none of these systems support machine learning.

- For computing the peak models on the SLURM 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; SegAnnDB uses BerkeleyDB instead.

Software and other Deliverables:

- Web app, as outlined above, active and demonstrated on <http://peaklearner.rc.nau.edu/>
- Professionally commented source code in a public GitHub repo owned by the client.
- Testing harness: a Travis-CI/Selenium/PhantomJS harness for headless browser testing.
- Video documentation for end-users who will log on to the system and perform interactive labeling / peak detection analysis.
- Well-written and complete user manual 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. This document should provide a strong basis for future development of the product.

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>
- [6] Dunn et al. Apollo: Democratizing genome annotation. PLOS Comp Bio 2019, <https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1006790>
- [7] <https://github.com/cmdcolin/wigglehighlighter>