GeneViewApp for GenomeLens

Spring 2015

Alexander Niema Moshiri

Sandra Hui

This quarter, we will continue to implement the GeneView portion of the GenomeLens pipeline. Specifically, we will focus on integrating the initial revision of GeneView into the Django framework. This includes creating a generic template for each gene image, as well as a generic template for each user. Using our vcf parsing and image generation from last quarter, we will continue working with Patrick, Yanli, and Hunter to create the proper framework for our app. We plan on having a generic pipeline on the common server to take a vcf file as input for a specific user, parse it, and generate images in the correct place for said user. We will only generate svg images, and the Django framework will dynamically generate the necessary web pages. That is, Django will use each user’s directory of svg images to fill in the template to create the user specific index file that lists links (buttons) to all the genes included in the patient’s vcf file. When a user clicks on one of these links, Django will find the corresponding svg image, and fill in the gene display template.

In the future, we could store the command to create each image, and create and save/cache each image on demand, rather than pre generating the images. Additionally, we’re considering making the lollipops interactive, so that the user can easily zoom or scroll, including creating a focused view for each gene directly within the index page (instead of going to a separate gene-specific page). We’d also like to find a better way to represent MAFs (currently, we’re taking the MAF for the general population and using it to differentially size the lollipops). Finally, we’d like to make the app accept other references besides ESP.

The app (GeneView.py) should be called once per user or patient vcf file.

Usage:

python GeneView.py </path/to/patient\_vcf\_file> </path/to/ref\_vcf\_file> <ref\_type [ESP]> </path/to/username\_or\_output\_dir>

Sample usage (takes about 10 minutes):

python GeneView.py ../../../Dropbox/GenomeLens/SVT/candidates\_CH\_SVT\_Final\_v2.vcf ../esp\_dl/ESP6500SI-V2-SSA137.GRCh38-liftover.chrAll.snps\_indels.vcf ESP ../exampleOut

Inputs:

* /path/to/patient\_vcf\_file: takes any path (relative or absolute) to the patient’s vcf file. This will let us use any input data organization structure, as long as the path is specified
* /path/to/ref\_vcf\_file: This is currently the ESP vcf for all chromosomes (when you download the ESP files from http://evs.gs.washington.edu/evs\_bulk\_data/ESP6500SI-V2-SSA137.GRCh38-liftover.snps\_indels.vcf.tar.gz, they come separated by chromosome. Sandra concatenated them using vcf-concat from vcftools). Right now, it’s stored locally on Sandra’s computer. Once the apps are running on a server, it can be located anywhere (maybe in the common reference folder?), as long as this app is given a path to it
* ref\_type [ESP]: Right now, this app only supports ESP reference files, since it parses the INFO fields based on ESP specific naming conventions. Amino acid changes are found by looking for particular strings in the INFO field. In the future, we’d like to expand this so other references, such as ExAC, can be used
* /path/to/username\_or\_output\_dir: takes any path (relative or absolute) to use as an output folder. All svg images are written here (ie /path/to/username\_or\_output\_dir/\*svg).

Outputs:

* All svg files are written to /path/to/username\_or\_output\_dir/\*.svg (see above). There’s an example in genomelens/GeneViewApp/static/images from the example usage above
* For the given user, an “index” page is generated via Django. The template for the user’s index page is in genomelens/GeneViewApp/templates/index1.html. We are planning on changing the template file’s name from “index1.html” to something like “GeneViewIndex.html” or something as such. Currently, the template accesses a file called “genes”, which is just a list of the genes in the input VCF. GeneView can generate this file, or the template can perform a basic “ls” function on the /output/ folder (this would generate a list of svg files, one for each gene, so a simple SED function can remove “.svg”, resulting in a list of genes). The template currently is a simple page that just has a list of links to each of the genes’ pages. It will be refined in the future to be prettier and/or more informative (i.e. alphabetical sort, “interesting” genes, etc.)  
  For the future, we’re going to implement the focus view for each gene (see above). If this happens, we will not require “geneviewer.html” (see below), as all genes will be embedded directly on one page (i.e. accordion view).
* For each individual gene, a page is generated dynamically via Django. The template for the individual gene pages is in genomelens/GeneViewApp/templates/geneviewer.html. geneviewer.html has a variable {{ gene }} that is set to the specific gene of interest dynamically as the user tries to view the page.