The Architecture, Implementation and Installation

of

CardioCPI (v. 2)

CardioCPI is a web-based application that provides facilities for statistical and plotting functions of gene and non-coding (mir) expression data. In the following the overall architecture and components of the application is described. This is followed by discussion of how the application was implemented. Next, an overview of the installation of the data base and application is given. Finally, in an Appendix, a step-by-step guide to installing and running CardioCPI on an Amazon AWS cloud instance is provided.

Architecture

Because it is web-based, the CardioCPI application can be divided into two functional areas: the client and the server. The client side is made up of the HTML pages and accompanying JavaScript code and libraries and CSS stylesheets. The client side components are delivered by a web server – Apache in the case of this application – although any web server should work. The server side of CardioCPI is connected to a networked data server, SSDB, that executes commands that store and retrieve data that is stored in a No-SQL data store – LevelDB is used for this application.

Components

1. Python (2.7.4) – The implementation language. Download from https://www.python.org/
2. Python modules - these can all be installed using the *pip* utility: http://pip.readthedocs.org/en/latest/index.html
    1. Django (1.6.1) – The web application framework.
    2. numpy (1.8.1 or newer) – Python scientific computing package
    3. pandas (0.13.1 or newer) – Python high-performance data structures (data frames, etc.)
    4. scipy (0.13.3 or newer) – Scientific computing package.
    5. matplotlib (1.3.1 or newer) – 2D plotting library.
    6. pyssdb (0.0.2 or newer) – SSDB client library.
    7. mne (0.7.1 or newer) – Additional statistical functions (p-value correction with False Discovery Rate)
3. SSDB - Downloads and installation instructions for a linux version can be found at http://ssdb.io/ NB: A Windows version can be found at https://github.com/ideawu/ssdb-bin
4. Bootstrap (2.3.2) – This CSS/Javascript library is included with CardiCPI.
5. Select2 (3.3.2 or newer) – This CSS/JavaScript library is included with CardioCPI.
6. Jquery (1.10.1) – This CSS/JavaScript library is automatically loaded by the web browser from the Google Hosted Libraries. NB: Cardiocpi has only been tested with this version of Jquery.
7. Datatables (2.2.1) – This CSS/Javascript library is automatically loaded by the web browser from the Datatables CDN (Content Distribution Network).

Implementation

CardioCPI is a Django application that was developed using IntelliJ IDEA. Python, HTML, JavaScript and CSS are used. In addition, the Select2, Bootstrap and jQuery JavaScript libraries are used for interactivity and the look and feel of the HTML pages

File Organization (primary)

CardioCPI

- manage.py – A Django management program.
- Cardiocpiv2 – Main Python files
  - geo_data.py - Interface to the SSDB data server.
  - plots.py - Implementation of Heatmap and Correlation plots
  - settings.py - Django application settings.
  - urls.py - Django url to view function mapping.
  - views.py - Django implementation of web services.
  - wsgi.py - Django file for web server integration.
- conf
  - cardiocpi.conf -
- media – temporary storage for generated plots.
- static
  - css
    - cardiocpiv2.css – CardioCPI-specific colors, etc.
    - css library files
  - image
    - MMCRI logo, et al.
  - js
    - cardiocpiv2.js – JavaScript to control the application's interactivity.
    - js library files
- templates
  - index.html - Django template of the main application page.
  - statistics.html – Django template for Statistical output; dynamically inserted into index.html

Load TOF Data SSDB

- data - contains the raw GEO “soft” data files
- src
  - Load_From_GEO_Soft.py – dataset initialization program.

Dataset Installation

Preliminaries

Save the dataset creation application – top-level directory is 'Load TOF Data SSDB' – to a local directory. This directory contains a dataset initialization program as well as the raw data files used as the dataset input sources.

Building the Dataset

The dataset for CardioCPI is based on two sets of data from NCBI's Gene Expression Omnibus (GEO) website:

1. GSE35781, Non coding RNA expression in myocardium from infants with tetralogy of Fallot - http://0-www.ncbi.nlm.nih.gov.elis.tmu.edu.tw/geo/query/acc.cgi?acc=GSE35781
2. GSE26125, Gene expression in cardiac tissues from infants with idiopathic conotruncal defects - http://0-www.ncbi.nlm.nih.gov.elis.tmu.edu.tw/geo/query/acc.cgi?acc=GSE26125

The dataset is managed by an independent, long-lived server: SSDB. SSDB must be installed and running before the dataset can be built and during the time that CardioCPI is being used. The best practice is to have SSDB running as an independent process: a daemon on Linux systems; a service on Windows systems. The server can also be run on an as-needed basis.

After SSDB begins running, a Python program (Load_From_GEO_Soft.py) is used to create the dataset. This application will read the GEO 'soft' formatted files that have been downloaded from the GEO website. Please note that this application is very ad hoc and is specifically written to handle two studies that used the same samples. Adapt this program to suit your particular data sources and requirements. Once the application is modified (if needed), building the dataset is accomplished by issuing the following command from within the same directory that the SSDB server is installed:

```
python Load_From_GEO_Soft.py
```

Note that, by default, SSDB creates the dataset files in a sub-directory named 'var'.

Application Installation and Running

Being a web-based application, CardioCPI needs a web server to deliver the web pages, plots, etc. to a browser. There are two ways to accomplish this: via an installed web server (Apache, for example) or by using Django's built-in web server. Both methods are described below.

Preliminaries

Save the application – top-level directory is CardioCPI – to a local directory.

If desired and prior to running the SSDB server, certain configuration parameters must be set. In the settings.py Django configuration file ensure that the variables `SSDB_HOST` and `SSDB_PORT` are set to the same values used when starting the SSDB server. The default values are `"localhost"` and `8888`, respectively.

Installed Web Server

Installation involves letting the web server know where the CardioCPI files are located. For the Apache web server this means modifying Apache's configuration file to point to the CardioCPI application. Consult you system documentation to locate the Apache configuration file; it is often named `"apache2.conf"`. Place the following line at the end of this file:

```
Include "<installation-directory>/CardioCPI/conf/cardiocpi.conf"
```

where <installation-directory> is the directory where the CardioCPI application files have been placed.

You must now modify the `cardiocpi.conf` file so that it points to your installation of the application. This is accomplished by replacing the string `"/home/ubuntu"` with the installation directory.

After these modifications, Apache needs to be restarted. Note that you must do this as a privileged user.

It should be noted that cardiocpi.conf file contains Apache instructions that results in mod_wsgi[8], the Apache plugin that provides Python support, being run in "daemon" mode. This is to allow multiple user access without interference.

Django Web Server

The other way of running the application is to use Django's built-in HTTP server. This server can be run by executing the following command from the home directory of the application:

```
python manage.py runserver localhost:8000
```

The parameter `localhost:8000` can be changed to what is appropriate for you system. Access to the application, then, is via a web browser using the url `localhost:8000` (in this example). Note that the url is different than what is used when the application is served by an installed web server such as Apache. Please note that the Django project does not recommend using their web server for production use because of performance and security reasons. Although we use the Apache installation, when running the Django Web Server we have not experienced any performance problems nor have there been any security concerns because the application runs within our site's firewall.

After one of these web servers has been set up, use a web browser to go to either:

- http://localhost:8000 or
- http://<hostname>/bio/cardiocpi

depending on the web server that you are using, where <hostname> needs to be replaced with the host name or ip address of the computer that is running the web server.

References

[1] Apache - http://httpd.apache.org/
[2] LevelDB - https://code.google.com/p/leveldb/
[3] Django - https://www.djangoproject.com/
[4] Select2 - http://ivaynberg.github.io/select2/
[5] Bootstrap - http://getbootstrap.com/2.3.2/
[6] jQuery - http://jquery.com/
[7] SSDB - http://ssdb.io/
[8] mod_wsgi - https://code.google.com/p/modwsgi/

Appendix

The following is a guide to the installation of CardioCPI on Amazon's AWS cloud facilities. It is assumed that the reader has an AWS account and knows how to create an EC2 instance. The instance is an Ubuntu 14.04 Server. The nano editor is used below but any editor will be OK to use.

AWS EC2 instance creation.

1. Locate the AWS AMI with the ID: ubuntu-trusty-14.04-amd64-server-20140607.1 (ami-864d84ee).
2. Choose the Instance type: t2.micro
3. Ensure that the security group for this instance allows inbound access to port 80. Add a new rule to the existing ssh rule.
4. Start the instance
5. Connect via ssh or, on Windows, PuTTY using the instance's private key.

Load necessary libraries and tools.

1. `cd /home/ubuntu`
2. `sudo apt-get update`
3. `sudo apt-get install apache2`
4. `sudo apt-get install libapache2-mod-wsgi`
   The Apache web server should have started. Go to a browser and enter the public IP address that has been assigned to this instance. You should see Apache's welcome page.
5. `sudo apt-get install python-pip python-numpy python-scipy python-matplotlib python-pandas`
6. `sudo apt-get install unzip`
7. `sudo pip install mne`
8. `sudo pip install pyssdb`
9. `sudo pip django==1.6`

Download, build and start the SSDB server.

1. `wget --no-check-certificate https://github.com/ideawu/ssdb/archive/master.zip`
2. `unzip master.zip`
3. `rm master.zip`
4. `cd ssd-master`
5. `make`
   Many messages and warnings will appear; you may ignore them.
6. `cd ..`
7. `./ssdb-master/ssdb-server ./ssdb-master/ssdb.conf &`

Download and run the Load CardioCPI program to populate the database.

1. `wget https://github.com/pcmarks/Load_CardioCPI_Data_SSDB/archive/master.zip`
2. `unzip master.zip`
3. `rm master.zip`
4. `cd Load_CardioCPI_Data_SSDB-master`
5. `python src/Load_From_Geo_Soft.py`
6. `cd ..`

Download the CardioCPI application.

1. `wget https://github.com/pcmarks/CardioCPI/archive/master.zip`
2. `unzip master.zip`
3. `rm master.zip`
4. `mv CardioCPI-master/ CardioCPI`
5. `sudo nano /etc/apache2/apache2.conf`
   Add the following line to the end of this configuration file:
   `Include "/home/ubuntu/CardioCPI/conf/cardiocpi.conf"`
6. `chmod a+wr CardioCPI/media/`
   This command gives permission to write plot files.

Restart the Apache webserver.

1. `sudo /etc/init.d/apache2 restart`
   If there are errors, make sure you have edited the apache2.conf file correctly.
2. Go to a browser and enter: <Assigned IP address>/bio/cardiocpi