UI for RNA-seq quantification [*]

xxxx xxxxx
SBU ID: xxxxxx
Department of Computer Science
Stony Brook University
xxxxx@cs.stonybrook.edu

xxxx xxxxx
SBU ID: xxx
Department of Computer Science
Stony Brook University
xxxxx@cs.stonybrook.edu

xxxx xxxxxx
SBU ID: xxx
Department of Computer Science
Stony Brook University
xxxxx@cs.stonybrook.edu

Jian Yang
SBU ID: 110168771
Department of Computer Science
Stony Brook University
yang16@cs.stonybrook.edu

Abstract

RNA sequencing, is a technology that uses the capabilities of next-generation sequencing to reveal a snapshot of RNA presence and quantity from a genome at a given moment in time. There are many modern algorithms for RNA-seq but there is few visualization about the algorithms' results. Based on this status, we implemented a UI for RNA-Seq quality comparison between various of algorithms. The UI could upload results from algorithms, generate plots and tables for quality metrics and a user could interactively view any data on the plots.

1 Introduction

2 UI Architecture

We implemented this from four components: backend scripts, server, front-end, webpage.

2.1 Backend scripts

At backend scripts, we did the following jobs.

1. Calculate the statistics from the transcript.
   We calculate the data such as GC content, ration of each nucleotides.
2. Load data from result files for algorithms and also the ground truth from profile files according to given parser.
3. Concat the data of results, the ground truth and the statistics based on transcript name as index to a large table.
4. Provide the API to query the data for UI layers.
5. Calculate the performance evaluation matrix.

The API contains the following methods.

[*]Course Project Report of CSE 549.

3 UI Features

We implemented the following features for the quality metrics with interactive modes to compare and view quality about algorithms. Currently we support 3 algorithms and a ground truth is added to compare the precision and the recall ratios.

3.1 Bar Chart for algorithms

We support the bar chart for all the algorithms to calculate the number of reads for each algorithm mapped in the experiment. One example is showed in figure **??**.

3.2 Scatter Plot for a Single Algorithm

For a single algorithm, we support the scatter plot about any two values such as transcript GC content versus abundance estimates. To plot a scatter plot, the user could go to the scatter plot for single algorithm page, choose an algorithm and the x-axis feature and the y-axis feature. With such a plot, every node on the scatter plot is one transcript and the coordinate is the chosen value pairs and as one example, the scatter plot for sailfish is showed in figure **??**.
For an interactive mode, you could move the mouse to a node you want to view and the information would automatically showed as in figure **??**.

3.3 Scatter Plot for two Algorithms on the same metrics

We could also compare two algorithms based on the same scatter plots. For example, we could compare the distribution between GC Content and Number of Reads for both Sailfish and Kallisto, as showed in figure **??**.

3.4 Scatter Plot for two Algorithms on the same feature

The same features from two algorithms are also comparable. For example, Number of Reads for both Sailfish and Kallisto. We expected they are similar if both of them are goo algorithms. And we could also compare it with the ground truth.

3.5 Performance Evaluation

We could also calculate the performance evluation based on ground truth. The ground truth could be uploaded at upload page showed (??add a figure or not???).
By given the ground truth, we now could calculate the following metrics: Spearman corr., TPEF, TPME, MARD, wMARD [1].

4 Extensibility

4.1 Upload new data

Currently the uploading of ground truth is added to re-calculate the performance evaluation results.
We also considered the extensibility in our implementation. Currently the result files are directly uploaded to the server side and by upload and rename to the targeted name, call $DataAnalyzor.load_data()$, the server could reload all data to replace the old metrics.

4.2 Add new algorithm

To add a new algorithm, one just need to add a related method in *AnalyzorBuilder* with the corresponding result file, parsing method same as in file $ParsingUtility.py$ and the new algorithm could automatically be supported by the whole framework.
At both the front-end and back-end, the algorithms are queried from the *DataAnalyzor*. Once the algorithms and the corresponding methods for reading the data in, the new algorithm and the data would be showed as same as other old features.

4.3 Add Features to compare

Currently we support a wide range of comparison: all possible pairs of

5 Acknowledge

Thanks and regards to Professor Rob Patro's lectures. We were inspired by the algorithms and the vivid examples.
Thanks and regards to TA Hirak. He helped us a lot on the project.

References

[1] Srivastava, Avi, Hirak Sarkar, and Rob Patro. "RapMap: A Rapid, Sensitive and Accurate Tool for Mapping RNA-seq Reads to Transcriptomes." *bioRxiv* (2015): 029652.