D3 Visualizations

Atif Ahmed (aa3931)

Outline

- Overview of the neural architecture
- Problem statement
- Preprocessing
- Prototypes/Approaches
- Separate visualizations
- Full visualization
- Technologies used
- What next?
- References

Overview of the neural architecture

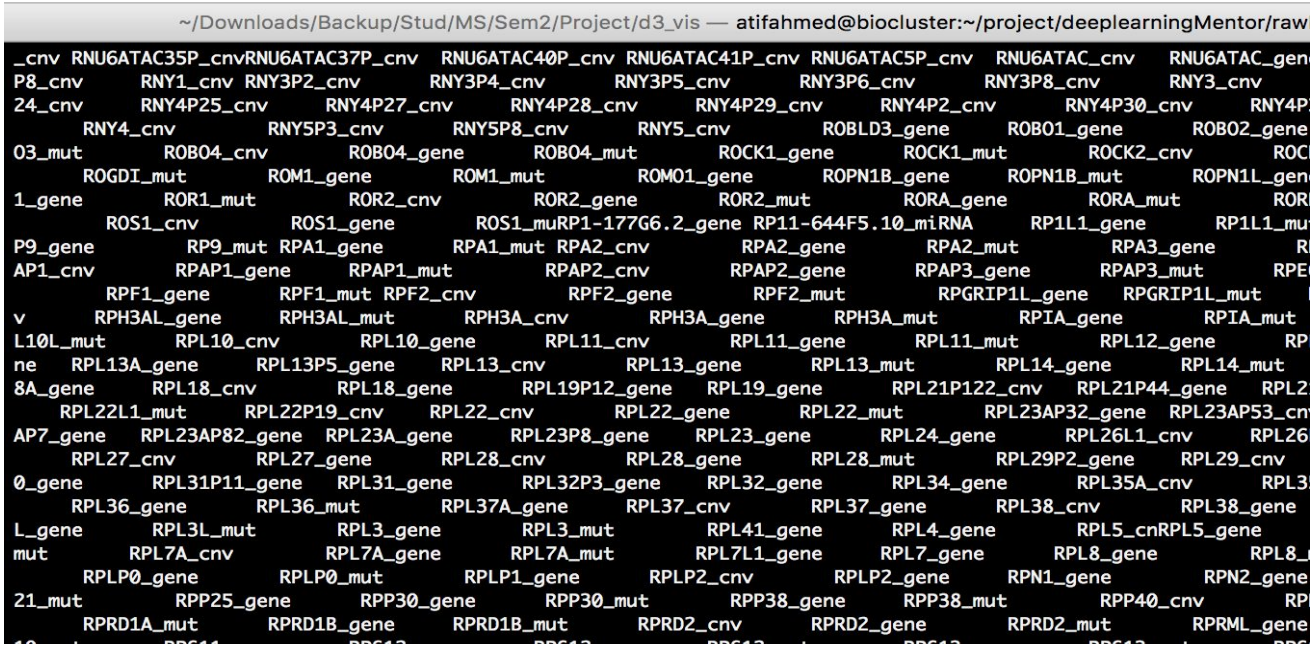
- Genomic and demographic data of large number of patients.

- High dimensional data (~4000 features).

- Uses group lasso and cox regression model for survival analysis.
  - Group lasso is used for taking care of any related features.

Overview of the neural architecture

- Uses only 2 layers (1 hidden and 1 output) and generates pretty 0.95+ accuracy.



Problem Statement

- We need to improve accuracy and predicted probabilities by fine tuning the neural architecture.

- For fine tuning the architecture, different combinations of hyperparameters such as number of hidden units, number of iterations, the activation function used etc. should be tried.

- Therefore, we need to build a tool to visualize these various configurations and how the neural network is working at each iteration.

Preprocessing

- Output produced by the model is stored in 4 different files:
  - test...predict.txt
  - test...groundTruth.txt
  - test...result.txt
  - test...weight.txt

- Simple non-serialized text files which makes getting object data from them impossible. Javascript cannot read it.

Preprocessing

- We need to write our own scripts to convert the text files into a structured format (json).

- test...predict.txt

Prototypes/Approaches

- D3.js was identified as the best approach. Has been used by playground.tensorflow.org also.

- First, existing designs were tried to visualize the network diagram of iCAGES.

- It was decided that predicted probabilities for patients should be shown with circles with colors density according to the magnitude of the probability.

Prototypes/Approaches

- It was decided that the accuracy and cost at each step (iteration) should be updated and shown dynamically.

- For the network architecture, basic circles and other svg shapes can be used.

Separate visualizations

- Visualizations were divided into 3 components:

  - Network architecture with changing output probabilities at each step.

  - Circles changing color densities according to the predicted probability at that step.

  - Dynamic visualization of accuracy and cost at each step.

Full visualization

- Integrated and synchronized all separate visualizations which are updated now simultaneously at each step.

- Dropdown options to update various hyperparameters. Visualization can be started with new choice of hyperparameters.

Technologies used

- d3.js for all the visualizations.
- Python for converting text data to d3-readable json files.
- SimpleHTTPServer for local web hosting and Github pages for hosting on Github.
- Twitter Bootstrap for prettifying the UI.

What next?

- We need to be able to show weight changes on the architecture.
- We need to be able to change even more hyperparameters.
- Use something like convnet.js to make everything only on browser and specific to our dataset and network (e.g. playground.tensorflow.org).

References

- d3Vienno tutorials (20 videos) for d3.js
  https://www.youtube.com/user/d3Vienno

- Blocks blog for better understanding of d3.js transitions and updating with new data
  http://bl.ocks.org/benjchristensen/1148374

- D3.js official examples for initial prototyping
  https://github.com/d3/d3/wiki/Gallery