10701 Project Proposal: Single Cell Analysis

Derun Gu (derung), Zhengyu Chen (zhengyuc), Zihao He (zihaohe)

1 Project Introduction

Genes contain information of instructing how to build a molecule, and different cells in our body express different subsets of these genes and at different levels. A technique named *single-cell RNA sequencing* (scRNA-seq) can be used to measure the activity levels of some annotated genes in a single cell, the result of which is termed *gene expression profile*. Same type of cells (e.g. lung cell, brain cell, skin epidermis) are closely connected with its gene expression profile.

The goal of the project is to classify the type of a cell based on the gene expression profile. The input is a real-valued vector with each element indicating the expressing level of each gene. The output should be the cell type it most likely belongs to.

2 Project Plan

There are a number of classification methods to approach this problem such as K-Nearest Neighbor, Random Forest, Neural Network etc. Due to the high dimension of the input data, it may be preferable to reduce dimension at first to remove noises and decrease data amount. But there are also trade-offs among different dimension reduction methods like unsupervised Principal Component Analysis, supervised Linear Discriminant Analysis and some non-linear algorithms. Moreover, there are also some prior knowledge as mentioned in [1] that we can encode in the algorithms to improve accuracy.

We plan to take advantage of scikit-learn Python library to try on some conventional machine learning algorithms, and use TensorFlow to train neural networks. The coding work will be collaborated through Github on personal laptops and the training and testing will be performed on AWS machines.

Details about each idea are presented as follows.

1. Pretrain Method:
   Several pretrain method might be used to speed up the training process. For instance, dimensionality reduction method like PCA, LLE, LDA can be used. According to our obeservation, the train data is a huge 21389×20499 dimesion matrix, with a lot sparsity. It is a bit too large for any of the training method and will take a lot of time to train while lots of information is redundant. Dimensionality reduction will make the training method focus on more important information and reduce the time required for each iteration of training.
2. Training Method:

   - SVM: Since the input data has more than 20,000 dimension, the the input data is relatively sparse in the space. SVM is suitable for this kind of problems as it assumes that there exists a hyper-plane separating the input data into different classes, which is easy to be done in high-dimensional space. Here we can use one vs. all solution for multi-class classification where we create a classifier for each class against all other data. After then for a new point we use all classifiers and compare the margin for all selected classes. Note that although it is not what we train the SVM for, but it often works well in practice.

   - Neural Network: As mentioned in [1], a shallow layered neural network performs well on the classification. The structure would be one or two layered fully connected

network and a *denoising autoencoder* (DAE) is used for pre-training. The hidden layer in DAE can be seen as a representation with reduced dimension which can suppress noises as well as decrease numbers of parameters. Furthermore, expert prior knowledge can be fused into the hidden layer which only connects a part of the input genes indicating different clusters of the input features.

- Decision Tree:
  We can make use of decision tree (e.g. ID3, C4.5, etc.) to do classification. Since each feature can take any real value, we can calculate the maximum and minimum possible values for each feature and set several thresholds based on them. At the same time, we can also make use of random forest to avoid overfitting. After collecting results from all decision trees, we can use the label with most voting as the final output.

3 Schedule

- Week 1 (Until 11/11): Do literature review and read through relevant papers.
- Week 2 (Until 11/18): Implement different methods and compare performance.
- Week 3 (Until 11/25): Focus on one or two methods and fine-tune the model.
- Week 4 (Until 12/2): Test on different parameters and test on ensemble methods to improve accuracy.
- Week 5 (Until 12/9): Print off a poster and prepare the presentation on 12/10.
- Week 6 (Until 12/16): Finish final paper in NIPS format.

4 Relevant Work

Lin etc.[1] predict the label by one or two layered fully connected neural networks, some of the which incorporate prior biological knowledge, and can even correctly group cells not used in the training. Alavi etc.[2] implement a web server which can download, process and annotate publicly available scRNA-seq datasets online and employ fast matching methods to determine cell types of scRNA-seq data posted to the server. Jang etc.[3] experiment on several types of classifiers (e.g. SVM, random forest etc.) on the scRNA-seq data and it turns out that SVM has the best accuracy.

References

[1] Chieh Lin, Siddhartha Jain, Hannah Kim, Ziv Bar-Joseph. Using neural networks for reducing the dimensions of single-cell RNA-Seq data. Nucleic Acids Research, Volume 45, Issue 17, 29 September 2017, Pages e156, `https://doi.org/10.1093/nar/gkx681`

[2] Amir Alavi, Matthew Ruffalo, Aiyappa Parvangada, Zhilin Huang, Ziv Bar-Joseph. scQuery: a web server for comparative analysis of single-cell RNA-seq data. bioRxiv 323238; `https://doi.org/10.1101/323238`

[3] Hankyu Jang, Samer Al-Saffar. Classifying Single-Cell Types from Mouse Brain RNA-Seq Data using Machine Learning Algorithms. `http://hankyujang.com/Papers`