

## 論文の内容の要旨

応用生命工学専攻

平成 29 年度博士課程 入学

氏 名 張 子龍

指導教員 清水 謙多郎

## 論文題目

### **Comprehensive evaluation of preprocessing methods for visualizing single-cell RNA-seq count data**

(単一細胞 RNA-seq カウントデータ視覚化のための前処理法の評価)

#### **Introduction (Chapter 1)**

Single-cell RNA sequencing (scRNA-seq) provides RNA expressions of a number of individual cells with high resolution. The technology can profile up to one million cells at a time, but the sequencing per cell is shallow. It results in false zero count observation for an expressed gene, often referred to as dropout event. It also has the overall high levels of noise mainly due to the low amounts of input RNA. Therefore, an important step when analyzing scRNA-seq count data is preprocessing (including imputation and smoothing). A number of preprocessing methods such as DrImpute and DCA (Deep Count Autoencoder) has been developed.

A typical scRNA-seq count matrix contains tens of thousands of genes or dimensions in rows and hundreds or thousands of cells in columns. After preprocessing, the dimensions are reduced to capture the underlying structure in the data and to visualize the data. The commonly used dimensionality reduction methods include the principal component analysis (PCA) and the t-distributed stochastic neighbor embedding (t-SNE). Although they are not designed specifically for scRNA-seq data having dropouts, many preprocessed data have been visualized specifically using t-SNE. A visual representation of the data with two or three dimensions (2D or 3D) has led to the discovery of new cell types and/or subtypes. However, since new methods are proposed one after another, it has become difficult to determine which method is suitable for one's own data.

This study mainly focuses on preprocessing methods for visualizing scRNA-seq data. I developed an autoencoder-based preprocessing method (called DAE) in this framework. The purpose of this study is to evaluate eleven preprocessing methods with t-SNE as well as five other methods. Comprehensive analysis with both simulated and real datasets provides sound recommendations regarding an optimal guideline for the dataset at hand.

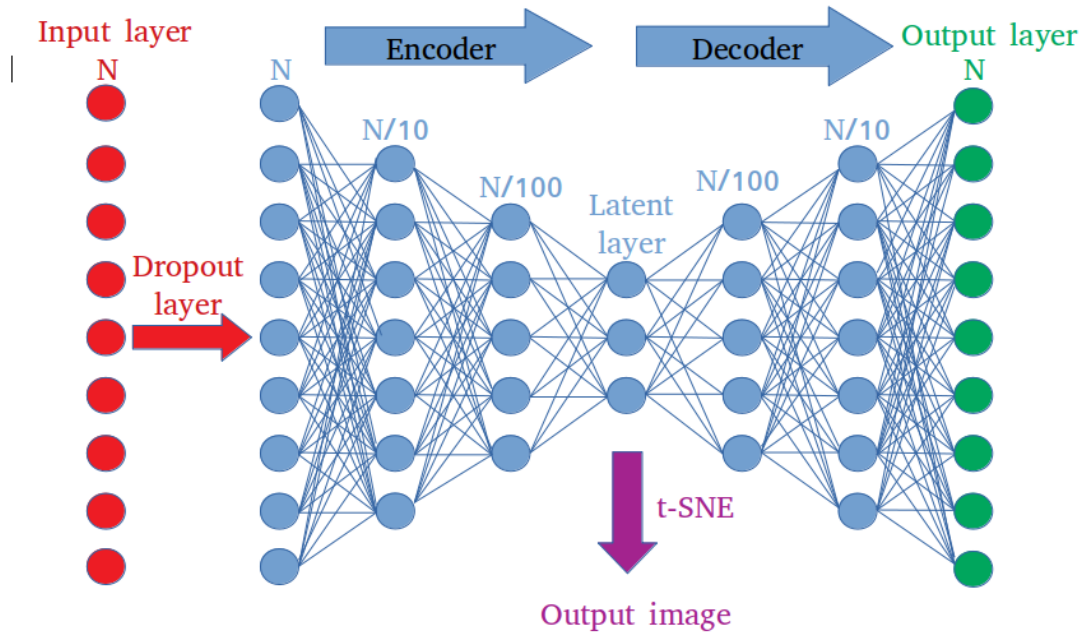
## Materials and methods (Chapter 2)

Simulated data were obtained using two R packages (Splatter and powsimR). A total of 17 conditions were evaluated: eight from Splatter and nine from powsimR. Each Splatter dataset contained 1,000 cells split into 2–5 groups (or cell types) and 1,000–5,000 genes of which 5–20% were differentially expressed (DE) across the groups. Each powsimR dataset contained 200–2000 cells split into three groups and 10,000–30,000 genes of which 10% DE. For real data, a total of 15 raw count datasets (five from humans and ten from mice) were obtained from the original papers. Each real dataset contained 56–3,005 cells split into 3–32 groups and 19,020–41,480 genes.

A total of 16 pipelines with raw count data as input and output reduced data were evaluated. They can be divided into three categories. The first category, consisting of two methods (PCA and t-SNE), only performs dimensionality reduction (i.e., visualization). The second category, consisting of four methods (ZIFA, PHATE, CIDR, and MAGIC), performs from preprocessing to visualization in one way. The third category, consisting of ten preprocessing methods (DCA, SAVER, scImpute, autoImpute, SAVER-X, DrImpute, LSimpute, kNN-smoothing, scRMD, and DAE), does not have a means of visualization by themselves. Accordingly, t-SNE was subsequently applied to the outputs of these preprocessing methods for obtaining the final 2D representation.

Similar to DCA, our DAE employs an autoencoder framework that tries to learn a representation of high dimensional data. It consists of four kinds of layers: an input layer, a dropout layer, hidden layers, and an output layer (Figure 1). The number of neurons/nodes ( $N$ ) in both the input and output layers is the same as the dimensions/genes in the count matrix. The dropout layer was designed to convert items in the matrix to zero at an arbitrary rate. The default dropout rate was set to 0.5. The hidden layers can further be divided into two kinds of networks, i.e., Encoder and Decoder. The encoder networks are designed as fully connected. By adjusting the weights of the neural networks, the autoencoder learns in an unsupervised manner how to efficiently compress the data. This compression corresponds to the dimensionality reduction. The latent (or bottleneck) layer is set in the middle of the hidden layers. Since this layer size can be changed arbitrarily, five different sizes (i.e., 10, 20, 30, 40, and 50 neurons/nodes) were investigated. The information in this layer corresponds to the low dimensional representation for the original high dimensional data and is the basic output of DAE. Although the decoder networks (the right side of the latent layer) themselves reconstruct the denoised count data, it is not important in this study.

The outputs of competing methods were evaluated using  $k$ -means clustering. It measures how well the low-dimensional space allows simple method (i.e.,  $k$ -means) to recover the true groups (or cell types). To evaluate a best-case scenario, the number of clusters ( $k$ ) was set to the number of known groups. I assessed methods by adjusted rand index (ARI), between predicted and true group labels as well as NMI and HOMO. For all criteria, the better method is closer to 1.



**Figure 1** Our DAE model.

### Results and discussion (Chapter 3)

In general, methods have substantial sensitivity to hyperparameters. In case of our DAE model, it corresponds to the latent layer size. I first evaluated it with five possible numbers by Splatter simulation and determined the use of 30 neurons would be the best. The rest of the analysis was performed using the 30 neurons for our model and default settings for the other methods.

Table 1 shows average ARI values obtained from individual analyses. For the simulated data generated by Splatter, three methods (kNN-smoothing, SAVER, and SAVER-X) showed nearly perfect ARI values on average. This trend was also observed when the other metrics (i.e., NMI and HOMO) were compared. For the simulated data generated by powsimR, PHATE performed the best overall. When the nine conditions were divided into three kinds according to the numbers of cells (i.e., 200, 1000, and 2000 cells), PHATE performed well on the large number of cells (i.e., 2000 cells). Although CIDR was the second best on average, it performed well on the small number of cells (i.e., 200 cells). The nine conditions can also be divided according to the numbers of genes (10000, 20000, and 30000 genes). Interestingly, best performed methods clearly differed: DrImpute and scImpute on 30000 genes, PHATE on 20000 genes, and kNN-smoothing on 10000 genes. These results suggest that the high performance of kNN-smoothing on Splatter simulation was simply because the small numbers of genes (1000 and 5000) used on the simulation were convenient for the method.

Many previous studies have evaluated performance by simulation using Splatter with relatively

small numbers of genes ( $\leq 5,000$ ). However, the 15 real datasets consist of much more numbers of genes ( $> 19,000$  and  $26,084$  genes on average). For the real datasets, DrImpute showed the best, followed by PHATE. This result is consistent with the powsimR results on the different numbers of genes. Taken together, it is better to change the method used depending on the number of genes in the data at hand. Although our DAE model outperformed DCA as a main competitor that also employed autoencoder, it failed to show outstanding performance. The unexciting results of DAE were probably due to parameter tuning only using a Splatter simulation with a small number of genes. Therefore, the improvement would be expected by performing parameter tuning with a large number of genes.

Table 1 Average ARI values.

Methods	8 conditions (Splatter)	9 conditions (powsimR)	15 real datasets
PCA	0.058	0.414	0.346
t-SNE	0.749	0.428	0.428
ZIFA	0.591	0.536	0.438
PHATE	0.741	<b>0.692</b>	0.520
CIDR	0.857	0.618	0.321
MAGIC	0.900	0.511	0.476
DCA	0.568	0.432	0.281
SAVER	0.999	0.546	0.439
scImpute	0.766	0.577	0.250
autoImpute	0.089	0.614	0.337
SAVER-X	0.998	0.523	0.450
DrImpute	0.781	0.544	<b>0.535</b>
LSImpute	0.780	0.371	0.377
kNN-smoothing	<b>1.000</b>	0.617	0.454
scRMD	0.778	0.577	0.353
DAE	0.827	0.554	0.485

#### Conclusion (Chapter 4)

The main finding of this study is that the preprocessing performance for visualizing scRNA-seq count data varies greatly depending on the number of genes. PHATE or DrImpute is practically recommended. Although this conclusion was based on the investigation of a total of 16 pipelines with both simulated and real datasets, further study is needed, especially the numbers of genes/groups as simulation parameters.