ECE 449 Report

Batch corrected scETM model in RNA-seq task

Yao Wentao, Kong Zitai, Liu Chang, Xu Ke

1. Abstract

Nowadays, the single cell RNA sequencing task with machine learning method become more and more popular in the biological research field. And the experiment result shows that the RNA-seq with machine learning can reach very high accurate. Among all these machine learning methods, the unsupervised learning with clustering is very fit for this task. Here, we explore the scETM model, which is a embedded topic modeling that is applied into the RNA-seq task. However, with our deeper research, we find that the batch effect is a big challenge in the RNA-seq model with machine learning. Therefore, we propose a batch correction network to improve the batch correction of scETM model and improve the performance of it.

2. Introduction

Single-cell sequencing technology is developing rapidly at the single-cell level and is being used in genomics, transcriptomics, proteomics and other fields for cell type recognition, tissue composition and reprogramming. More specifically, single-cell transcriptome sequencing, or single-cell RNA sequencing, has become the dominant technology in many cutting-edge research areas, such as disease progression and drug discovery. scRNA-seq is already having a real impact in the cancer field and is becoming a powerful tool for understanding invasion, intra-tumor heterogeneity, metastasis, epigenetic changes, detection of rare cancer stem cells, and therapeutic response. Currently, scRNA-seq is being used to develop personalized treatment strategies that have potential usefulness in cancer diagnosis, treatment resistance during cancer progression, and patient survival. scRNA-seq is also being used to combat COVID-19 to shed light on how miscommunication between the innate and adaptive host immune systems leads to the deterioration of immunopathology that occurs during viral infection. These studies have led to the storage of large amounts of scRNA-seq data in public databases, such as the 10X Single cell Gene Expression Dataset, human cell Atlas, and mouse cell Atlas. On the other hand, due to biological and technical factors, scRNA-seq data have six complex characteristics, such as missing expression values, high technical and biological variance, noise and sparse gene coverage, and elusive cell identity, which make it difficult to directly apply commonly used bulk RNA-seq data analysis techniques. Therefore, new statistical methods are needed to clean up the scRNA-seq data, as well as computational algorithms for data analysis and interpretation. To this end, specialized scRNA-seq analysis pipelines (such as Seurat and Scanpy), as well as a host of mission-specific tools, have been developed to deal with the complex technical and biological complexities of scRNA-seq data. In recent years, deep learning has shown significant advantages in natural language processing and speech and face recognition in the context of large amounts of data. These advantages open up the application of deep learning in the analysis of scRNA-seq data, which is competitive with traditional machine learning methods to reveal cell clustering, cell type recognition, gene input and batch correction in scRNA-seq analysis. Compared with traditional machine learning methods, deep learning method is better able to capture complex features of high-dimensional scRNA-seq data. It is also more flexible; a single model can be trained to handle multiple tasks, or adapted and transferred to different tasks. In addition, deep learning training is more adaptable to the number of cells in the size of the scRNA-seq data, making it particularly attractive for

handling the increasing volume of single-cell data. In fact, a growing number of deep learning-based tools have demonstrated the exciting potential of deep learning as a learning paradigm that could greatly improve the tools we use to interrogate scRNA-seq data.

3. Related Works

3.1 scETM model

3.1.1 Network Frame

The scETM model is composed of a variational autoencoder and a linear decoder. The variational autoencoder serves to extract the topic information of a cell into Gaussian distributions and the linear encoder provides prediction of cell-gene distribution.

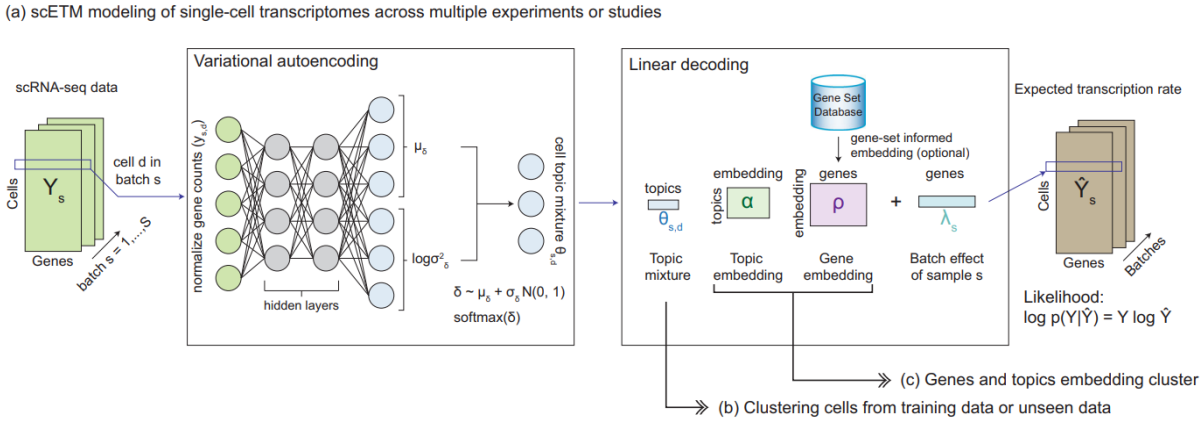
The input scRNA-seq data is preprocessed into a couple of matrixes, a matrix will a cell in a row and the genes expressed of that cell in a column, and each matrix represent a batch of the biological experiments. Then the matrixes are feed into the model.

The variational autoencoder composes of a 2-layer neural network plus a softmax layer. The neural network has hidden sizes of 128 and uses ReLU activations. For each cell, the variational autoencoder can learn the mean and logarithm of variance of its gene expression amount, and each pair will be put into a normal distribution after softmax, which represents the topic of each cell. All these cell-topic relation will be put together to be a cell topic mixture matrix θ. The gene embedding dimension is set to be 400, and the number of topics is set to be 50.

$$\delta_d \sim N(0, I), \quad \theta_d = softmax(\delta_d) = \frac{e^{\delta_{d,k}}}{\sum_{k=1}^{K} e^{\delta_{d,k}}}$$

In the linear decoder, we will have three matrixes. One is the cell topic mixture (cells-by-topics) matrix θ comes from encoder, the second is a mixture of topic and gene information from gene set database, i.e., topic embedding matrix α and the last one is the gene embedding matrix ρ coming from gene set database. Then we add a batch effect correction matrix λ to remove the batch effect. Finally, we can get the expected amount of gene expression (of cell d in batch s) by

$$\mathrm{softmax}(\theta_s, d\alpha\rho + \lambda_s)$$



3.1.2 Dataset

We use 4 batches of Human's data, each with 8569 cells by 20125 gene/cell and 2 batches of mouse's data, each with 1886 cells by 14878 gene/cell. The Human pancreatic islet dataset come from the GEO or EMBL-EBI database under the accession codes GSE81076, GSE85241, GSE86469, E-

MTAB-5061, and GSE84133. Mice's comes from the GEO database under the accession code GSE84133.

In the dataset, each row represents all gene expression of a cell, each column represents all cell's gene expression of a certain gene. There are also barcode showing the origin of the cell and cell label showing ground truth cell type.

	barcode	assigned_cluster	0610007P14Rik	0610009B22Rik	0610009E02Rik	0610009L18Rik	0610009O20Rik	0610010F05Rik	0610010K14Rik
mouse2_lib1.final_cell_0001	GAAAGATTGT-AAACCTCC	beta	0	0	0	0	0	0	0
mouse2_lib1.final_cell_0002	ACTCCGCAT-GTTAACCA	beta	6	2	1	0	0	0	0
mouse2_lib1.final_cell_0003	AGGGAACGA-GCTTTCCA	beta	0	0	0	0	0	0	0
mouse2_lib1.final_cell_0004	AAATGACCC-ACTCACCG	ductal	2	0	0	0	0	0	0
mouse2_lib1.final_cell_0005	AAGTGAAAG-GAAGTGCC	beta	0	1	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
mouse2_lib3.final_cell_0391	TGATTGCACGC-CATTTGTT	beta	0	0	0	0	0	0	0
mouse2_lib3.final_cell_0392	TAACTACT-AAGTAATC	beta	0	0	0	0	0	0	0
mouse2_lib3.final_cell_0393	TGACCTGTTAT-TGATGCCC	ductal	0	0	0	0	0	0	0
mouse2_lib3.final_cell_0394	TGAGTAATCCC-AACCCTTG	quiescent_stellate	0	0	0	0	0	0	0
mouse2_lib3.final_cell_0395	AGCACCTCT-ATTCCTTG	ductal	0	0	0	0	0	0	0

3.1.3 Details of Training

The encoder is chosen to use Adam optimizer with a 0.005 learning rate; 12,000 epochs; a minibatch size of 2000; 5k-20k training steps to converge; 1D batch normalization; 0.1 drop-out rate between layers. To prevent over-regularization, the original paper starts with zero weight penalty on the KL divergence linearly increase the weight of the KL divergence in the ELBO loss to 10^{-7} during the first 1/3 epochs.

3.1.4 Loss Function

In original paper, they use Evidence Lower Bound (ELBO). The first term is the expectation of reconstructed log-likelihood based on variational posteriori distribution, which shows the upper bound of estimation. The second term is the KL divergence of predicted distribution and the true distribution. We optimize this loss to make the proposed result as close to ground truth as possible.

$$ELBO = E_q[\log p(Y|\theta)] - KL[q(\theta|Y)||p(\theta)]$$

$$q(\delta_d|y_d) = \mu_d + diag(\sigma_d)N(0, I) = N(\mu_d, diag(\sigma_d))$$

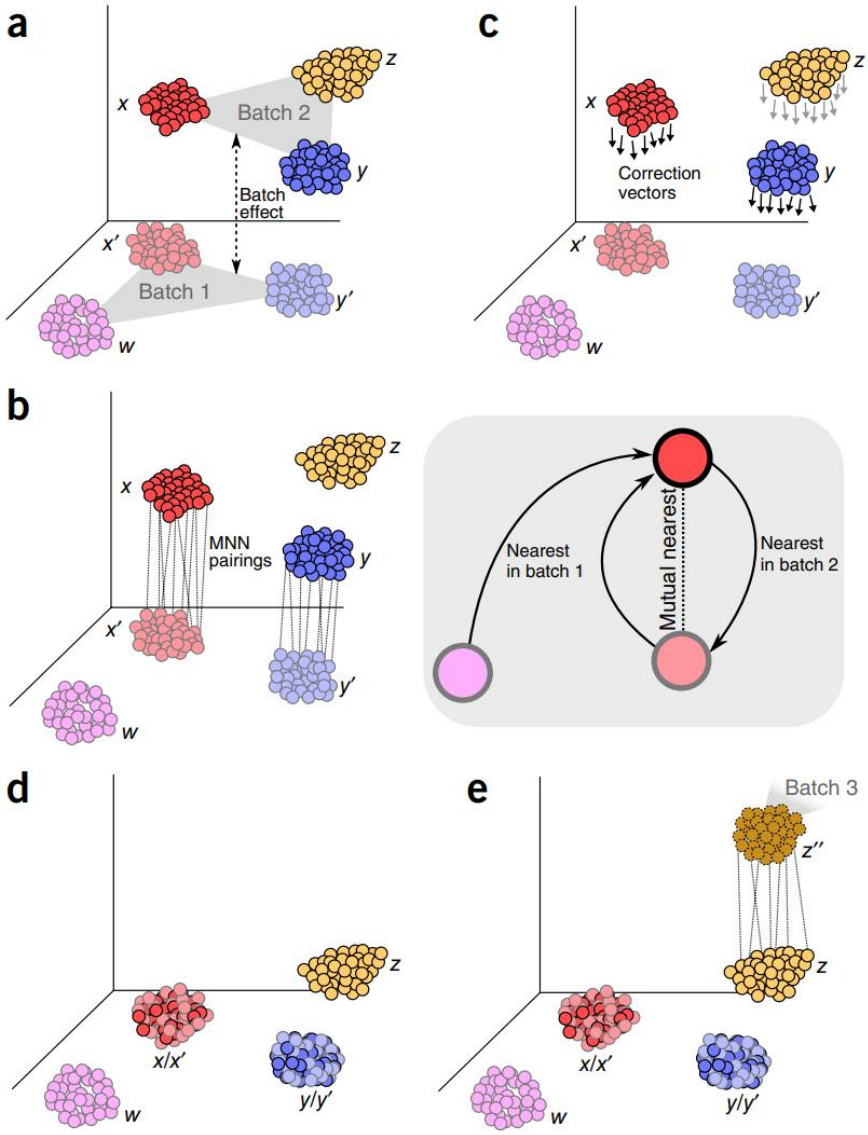
3.1.5 Batch Effect and Problems

Batch effect is the error caused by different batches in experiments, normally caused by different person, devices or reagents applying on the same experiment. This may cause the data from the same source to diverge.

Since scETM model only considers a single categorical batch variable by adding a batch correction matrix λ, more effective ways to correct batch effects needed to be involved.

3.2 Batch correction models

Through our research, we find that the batch effect is a big challenge and problem in the RNA-seq task. Therefore, we do some research on this field. And there are some papers that are of our references. Firstly, in the paper from (Laleh et al, 2018), it presents the basic mutual nearest neighbor method. As shown in the figure below.



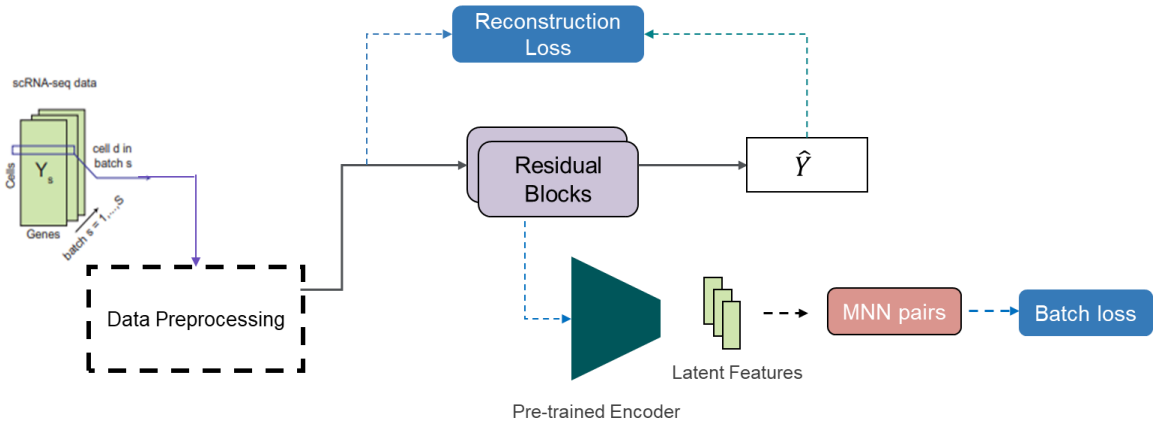
From each sample in a batch, we search every nearest neighbor from other batches. If one of its neighbors also has a nearest neighbor of it, they become a mutual nearest pair. The goal of it is to minimize the distance among the nearest neighbor pair. This method can apply well in the task of RNA-seq. And in the paper (Li et al, 2020), the author presents an autoencoder based method. In the article, the author mentions that the autoencoder do well in extracting the feature of Gene expressions. It uses the latent feature extracted from the encoder to do the clustering and get the classification. In the paper (Shaham U et al, 2018), the author use the residual network with the Maximum Mean Discrepancy (MMD) loss to generate the data with removing the batch effect. And in the paper (Zou et al, 2021), the author combines the mutual nearest neighbors and the residual network together. In that case, the network can generate the data that is with similar expression with the original data and at the same time.

Method

4.1 Model Overview

Based on our research, we find that one outstanding challenge in the RNA-seq task is the batch effect issues, which is caused by different experimental conditions and different batch of samples. Therefore,

we proposed our model based on scETM model. Inspired by the combination between MNN method and residual network, we proposed our batch – corrected scETM model. The framework of our model is shown in the figure below. As shown here, it is a two staged model. The scETM model used in our model is nearly the same as the original work. The only difference is that we remove the original batch correction strategy. In replacement of it, we propose our batch correction network. The Gene matrix data will be processed by this framework to remove the batch effect. After that, the processed data will be feed into the scETM model to start the training or inference process. Our batch correction network has four different steps: data preprocessing, dimension reduction, searching MNN pairs and training network with reconstruction and batch loss. The input gene matrix will firstly receive data preprocessing. Then, the latent feature of gene matrix will be extracted by a pretrained encoder. And then, the algorithm can be applied to these latent features to find MNN pairs. And our network will be trained on our batch correction network. In the inference stage, the data feed into our network will generate the batch corrected data through residual network.

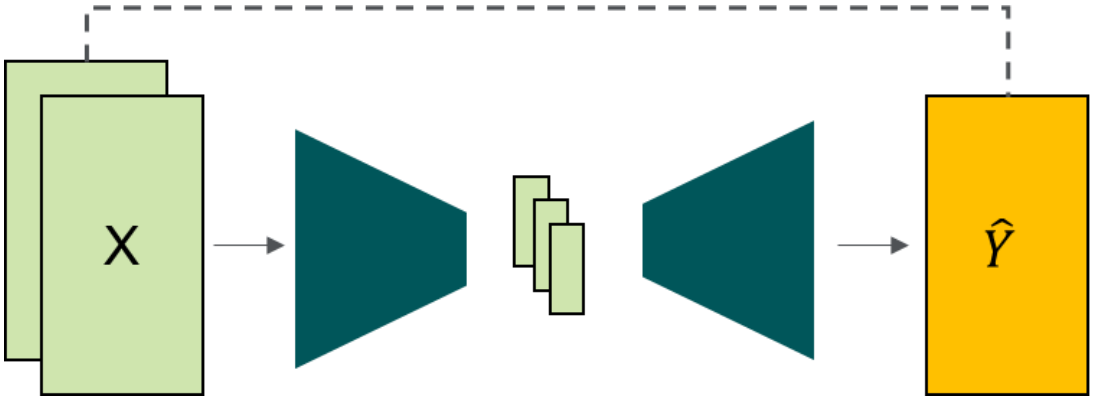


4.2 Data Preprocessing

In this part, steps in preprocessing is nearly the same as the DeepMNN model. 1. The data will be filtered by the users' defined criteria. 2. The cell expressions will be normalized with the total expressions. And then the data will be log-transformed. 3. Then, since the gene matrix is in very high dimension, and many genes are not significant in our task with very fewer expression. Therefore, we extract the 2000 highest expressed genes. After that, the gene expression data is still in very high dimension. Therefore, we need to further reduce the dimension of the data.

4.3 Dimension reduction through pre-trained encoder

Here, we have a pre-trianed encoder in my framework. If all the data are in the same gene subspace, they will share the same encoder in our model. This encoder is used to extract the latent gene feature used for searching MNN pairs. And in the training stage, we use these two encoders to extract the latent feature of our gene expression data and minimize the distance of a MNN pair through the batch loss.



4.4 Searching MNN pairs

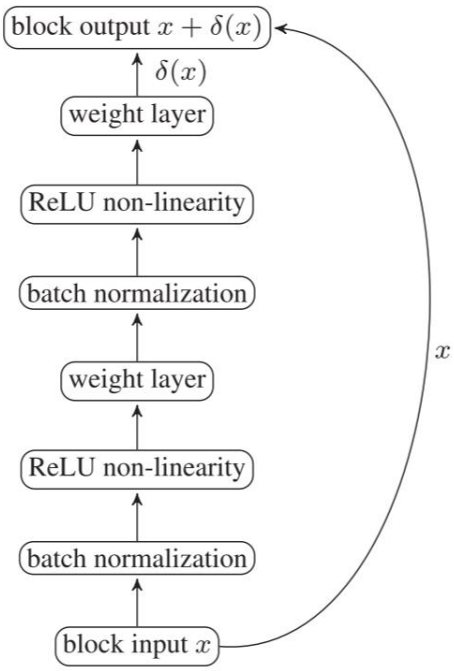
The search of MNN pairs is based on the latent features extracted from our pre-trained encoder. For each cell, we will search 20 nearest neighbors in other batches. And if any cell in the nearest neighbors also has a neighbor of that cell. In that case, they become a mutual nearest neighbor pair. After that process, we have the set of all MNN pairs. And all these data will make up to our training batches. The model will train on these data to minimize the distance of MNN pair and reconstruct the gene expressions data.

4.5 Loss function

In our framework, we have two different losses. Reconstruction loss and batch loss. The batch loss is used to reduce the distance within a MNN pair. It is the Euclidean distance in the feature subspace in a batch of training data, as shown in the expression below. Here, for a training batch b, and the MNN pair k, we calculate the distance between this pair, and finally add all these distance together, as shown in the following expression:

$$L_{batch} = \sum_{k} ||Y_{k0}^{b} - Y_{k1}^{b}||$$

Also, we have another loss function, it is the reconstruction loss. The preprocessed data will be feed into the residual network. Since the residual network has the residual term and identity term. So, it is easy for this network to learn the identity function of the input value. The structure of the residual network is shown in the figure below:



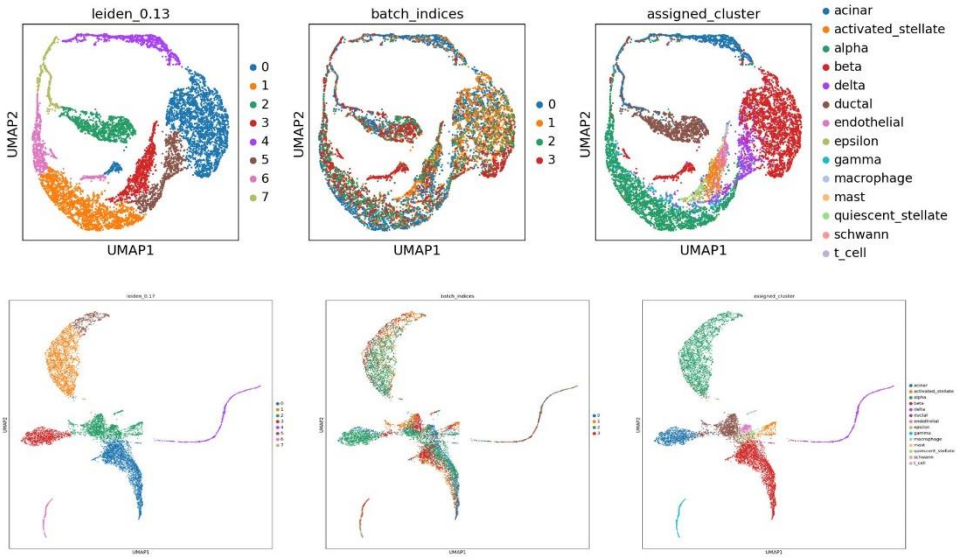
In that case, this network will serve as a calibration for our generated data. The reconstruction loss is also the distance between the generated data and the original data, as shown in the expression below:

$$L_{rec} = \sum_{k} ||\widehat{Y}_k - X_k||$$

So, here we have the final loss function, which is the linear combination of the batch loss and reconstruction loss. As shown in the expression below:

$$L = \alpha \cdot L_{batch} + \beta \cdot L_{rec}$$

5 Experiment



The result of our batch effect correction on human cells is shown above. The figure above is the result of our model and the other one represents the original model introduced in the paper. In each figure, the

graph on the left shows the result of our classification, the one in the middle shows the distribution of data samples from all batches in different colors and the one on the right is the manually labelled ground truth. The original model can not eliminate the batch effect very well as the dots are distributed unevenly. In some area, the density of green dots are obviously higher than other colors. On the contrast, after our method is applied, the dots of different colors mixed together more evenly. It's also worth noting that the reason why the shape of distributions are different is that the data were normalized during pre-processing.

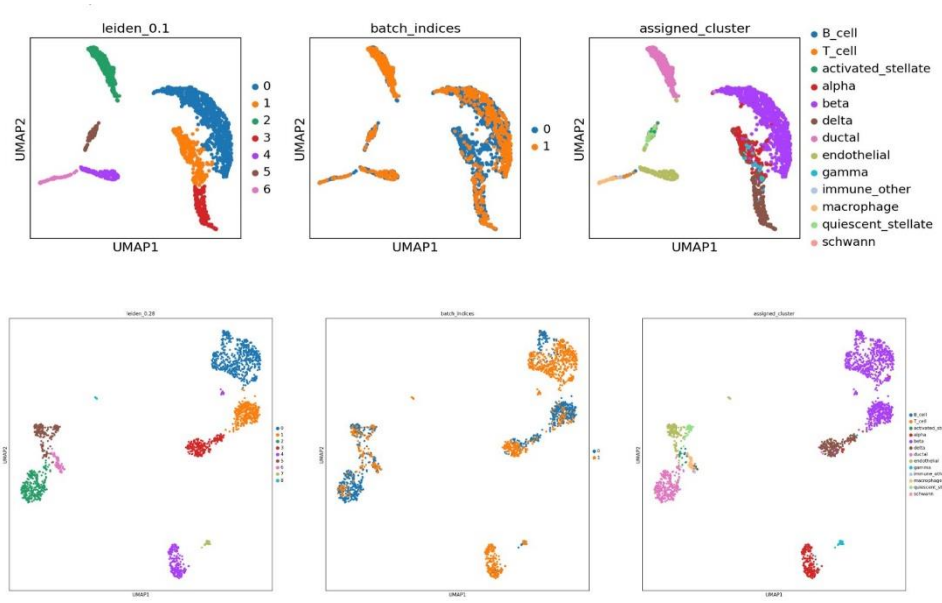
Our Model:

Our Model Clustering Performance on Data Integration Tasks			
Resolution	ARI	NMI	bARI
0.1	0.5907	0.6266	0.0492
0.13	0.6694	0.6509	0.028
0.19	0.5459	0.6203	0.0476
0.22	0.487	0.6112	0.0447
0.25	0.5212	0.6252	0.0504
0.28	0.4253	0.5921	0.0389

Origin Model:

scETM Unsupervised Clustering Performance on Data Integration Tasks			
Resolution	ARI	NMI	bARI
0.08	0.5809	0.6992	0.0167
0.12	0.5474	0.6816	0.0251
0.17	0.7754	0.7747	0.0401
0.23	0.6473	0.737	0.0416
0.3	0.6	0.7338	0.0519
0.4	0.6007	0.735	0.0538

As for the results of the clustering, our model outperforms the original one in low resolution condition.



The effect of batch effect correction is still obvious on the second data set of mice cells. The group of yellow and blue cells mixed together more evenly after applying our method.

Our Model:

Our Model Clustering Performance on Data Integration Tasks			
Resolution	ARI	NMI	bARI
0.1	0.6654	0.6666	0.0557
0.13	0.7311	0.7313	0.054
0.19	0.7644	0.742	0.0456
0.22	0.7644	0.742	0.0417
0.25	0.7799	0.766	0.0415
0.28	0.7799	0.766	0.0415

Origin Model:

scETM Unsupervised Clustering Performance on Data Integration Tasks			
Resolution	ARI	NMI	bARI
0.1	0.549	0.6867	0.1441
0.13	0.549	0.6867	0.1441
0.19	0.6386	0.7627	0.138
0.22	0.6386	0.7627	0.138
0.25	0.6386	0.7647	0.1373
0.28	0.6527	0.7873	0.1376

Moreover, we got better clustering results on this dataset, with almost all ARI values higher than the origi5nal model. A preliminary guess is that as there are only two batches of data in this data set, the task of

batch effect correction is simpler, and our model performed better in this case.

6 Discussion

By applying our modified deep MNN model for data preprocessing, we get quite good results. For our future work, we can firstly apply our methods on more datasets to verify its performance, we can also further improve the batch effect correction network, improving its efficiency, and we will also explore the scETM model, improving its interpretability.

7 Reference

[1] Haghverdi, L., Lun, A., Morgan, M. et al. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. Nat Biotechnol 36, 421–427 (2018). https://doi.org/10.1038/nbt.4091

[2] Li, X., Wang, K., Lyu, Y. et al. Deep learning enables accurate clustering with batch effect removal in single-cell RNA-seq analysis. Nat Commun 11, 2338 (2020). https://doi.org/10.1038/s41467-020-15851-3

[3] Zou Bin, Zhang Tongda, Zhou Ruilong, Jiang Xiaosen, Yang Huanming, Jin Xin, Bai Yong deepMNN: Deep Learning-Based Single-Cell RNA Sequencing Data Batch Correction Using Mutual Nearest Neighbors 10.3389/fgene.2021.708981

[4] Shaham U, Stanton KP, Zhao J, Li H, Raddassi K, Montgomery R, Kluger Y. Removal of batch effects using distribution-matching residual networks. Bioinformatics. 2017 Aug 15;33(16):2539-2546. doi: 10.1093/bioinformatics/btx196. PMID: 28419223; PMCID: PMC5870543.

[5] Zhao, Y., Cai, H., Zhang, Z. et al. Learning interpretable cellular and gene signature embeddings from single-cell transcriptomic data. Nat Commun 12, 5261 (2021). https://doi.org/10.1038/s41467-021-25534-2

[6] Mario Flores et al. Deep learning tackles single-cell analysis – A survey of deep learning for scRNA-seq analysis. https://arxiv.org/abs/2109.12404