Transfering Cell Type Annotations between Single Cell Experiments

Chew Chai, Xingting Gong, Pankaj Sharma
chewc@stanford.edu gongx@stanford.edu pansword@stanford.edu

June 11, 2020

Abstract

Single cell RNA sequencing (scRNA-seq) revolutionized life science by bringing unprecedented resolution to study heterogeneity in cell populations, enabling many new biological discoveries. With an increasing speed of data generation, single cell biology calls for bioinformatics approaches to transfer cell type annotations in an automated manner, given a large compendium of expert annotated cell atlas. However, the cavets to scRNA-seq analysis are batch effects, dropout, technical noise and curse of dimensionality. To tackle all these challenges altogether, we implemented an autoencoder using zero-inflated negative binomial loss function with adversarial network, which we then feed the inputs to multiclass classifier to transfer cell type annotation in an automated manner. Our current model achieves an accuracy of 90-96% on various size of training and test datasets.

1 Introduction

In recent years, large scale single cell RNA-sequencing (scRNA-seq) has made it possible to measure the entire transcriptome of individual cells, allowing scientists to resolve heterogeneities in gene expression amongst cells with great detail [1]. One of the most important tasks in single cell analysis is cell type annotation, which aims to label groups of cells according to their gene expression patterns. With an increasing speed of data generation, single cell biology calls for bioinformatics approaches to transfer cell type annotations in an automated manner, given a large compendium of expert annotated cell atlas.

However, a challenge of scRNA-seq analysis is the existence of batch effects, wide variations in both experimental and technical conditions (i.e. cell barcoding, sequenced regions, library preparation, sequencing method, read depth) that make differences in the captured data difficult to interpret [2]. For some lowly expressed genes, their small numbers of RNA molecules and the stochastic nature of transcriptional processes could lead to spurious zero entries in the expression matrices of scRNA-seq data. In addition, the input data is extremely high dimensional, with order $\sim 20,000$ genes for anywhere from thousands to millions of cells being sequenced at once. Here, we want to capitalize on the removal of batch effects and technical noise to transfer the cell-types annotation between experiments of different sequencing platform as well as across species, for instance, mice to human to address the issue of human cell shortage. We propose a two-step learning model. First, we will use an **Autoencoder to reduce the input data into a lower dimensional latent space**. The latent space will capture the meaningful features of the input, thereby reducing the noise due to batch effects. Second, we will then build a **multi-class classifier to label the cell-type of single cell experiments of interest**, which takes the latent features from the autoencoder as inputs.

2 Related work

Previously, a few deep learning approaches have been implemented to address the challenges. Shaham et al [3] proposed one of the first method to remove batch effect using the residual neural networks (ResNets). Since ResNets can easily learn mapping functions that are close to the identity function, and thus are suitable for calibrating a source sample to match the target sample in the batch effect removal. Eraslan et al [4] proposed

a method called "DCA" (deep count autoencoder) to solve denoising and imputation altogether by defining the loss function in terms of noise model, zero-inflation neative binomial (ZINB). DCA has two advantages: (1) its ability to capture the nonlinear dependencies among genes (2) scalability to millions of cells through GPU usage. Several deep learning methods for dimensionality reduction have also been proposed. Ding et al [5] used the varational autoencoder (VAE) to infer the approximate posterior distributions of low-dimensional latent variables, thus learning the parametric mapping from a high-dimensional space to a low-dimensional embedding.

Most of the above methods are each focused on one or two tasks of data analysis, thus, it would be desirable to integrate different tools into one joint framework [6]. Additionally, more interpretability is needed for scientific knowledge discovery; a desriable aspect of interpretability is the automatic construction of a model that is able to simulate the single cell dynamics. For that, generative adversarial network (GAN) model is promising [7]. With this in mind, we combine autoencoders with adversarial network in our approach to have more interpretable noise profiles.

3 Dataset and Features

We will be using Smart-Seq2 mouse dataset (32000 cells) from Tabula Muris project (https://tabula-muris.ds.czbiohub.org/) [8]. As shown in Figure 1A, the input data is a two dimensional matrix, with rows as individual cells and columns as gene expression counts (23000 genes). Since the autoencoder is an unsupervised learning problem, we do not need labelled data. In figure 1B is the break-down percentage of cells for 18 different organs.

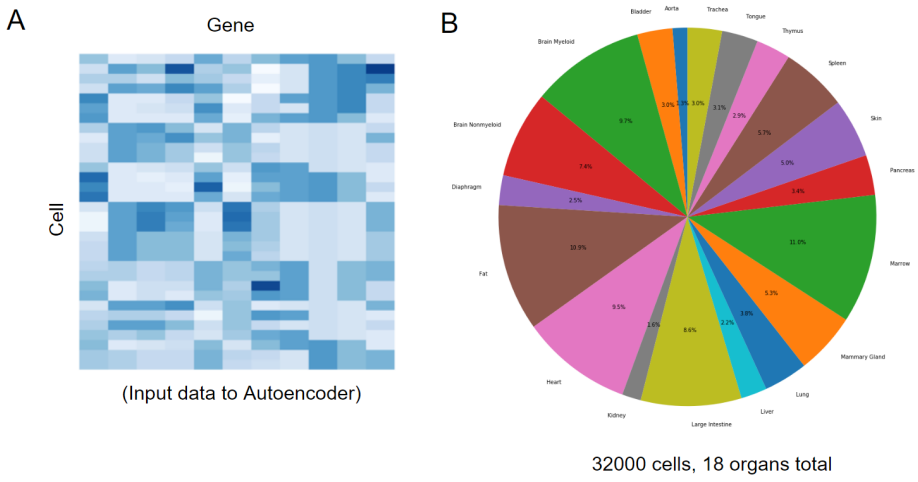


Figure 1: (A) Input gene expression data. The columns denote gene expression counts for each cell(row). (B) Training dataset of 32000 cells and 18 organs of Tabula Muris mouse atlas

The two test dataset that we will be using to evaluate our model are single cell experiments done using a different sequencing platform for mouse (10X Genomics microfluidic droplet based), and human dataset [9] for across species. Figure 6A and 6B in the Appendix provides the break-down percentage of cells of different organs that will be tested.

4 Methods

4.1 Autoencoder

To address the challenges of batch effects and dropout noise posed by single cell analysis, we employ zero-inflated negative binomial loss function from DCA for our autoencoder architecture as shown in the top

part of Figure 2. To further reduce technical noise in the latent features we extracted, we further modify our autoencoder by incorporating adversarial network, in which we minimize AE_{loss} and maximize G_{loss} by preventing the encoder/generator from generating standard normal distribution noise (Figure 2). All the mathematical details are included in the figure along with the network architecture schematics.

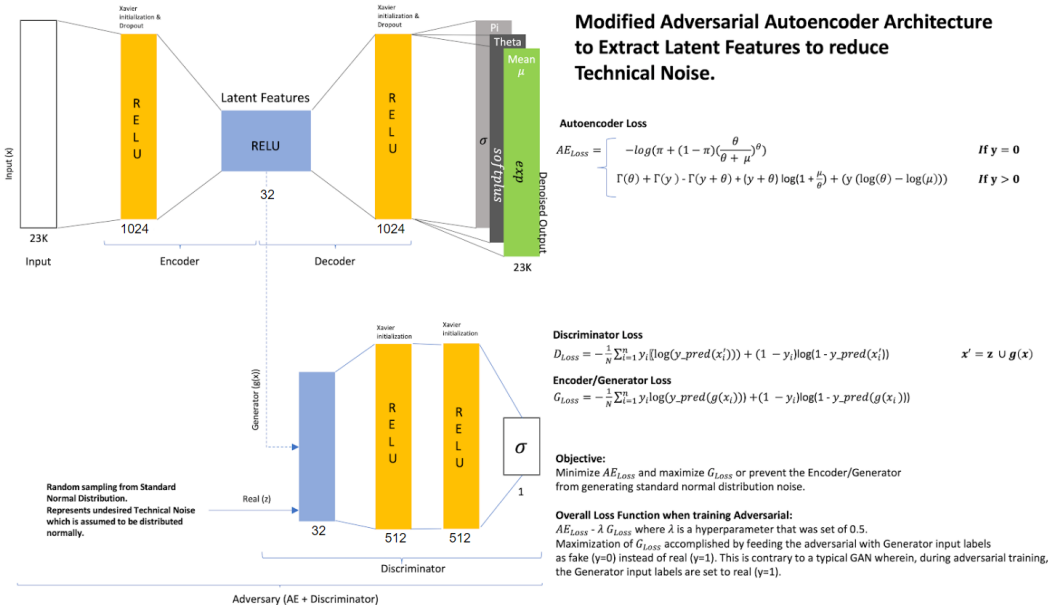


Figure 2: Autoencoder with ZINB loss function to account for batch effect and dropout as well as adversarial network for extracting technical noise

4.2 Multiclass classification

Using the latent features output from the autoencoder, we then feed it into the classifier network, which employed categorical cross entropy as a loss function, Adam as optimizer, RELU hidden nodes, and one-hot encoded cell-type labels as shown in Figure 7 in the Appendix.

5 Results and Discussion

5.1 Investigation of different models

Before deciding on the final model for Latent feature extraction, we investigated several variations of a base Autoencoder model. The four different models that were investigated were: (1) AE with Mean Squared Error (MSE) loss, (2) Adversarial AE with Mean Squared Error (MSE) loss (3) AE with ZINB loss and, (4) Adversarial AE with ZINB loss. We found that although the accuracy of each of the models were comparable, using the Adversarial AE had a stabilizing effect on the the variation in the loss and accuracy curves as shown in Figure 4. Based on the these outcomes were decided to use the Adversarial AE with ZINB as our final model.

5.2 Hyperparameter search

We did a hyperparameter search on the simplest models to get a baseline for what works well. We thus started by focusing on an Autoencoder (AE) without adversarial and a simple, single-layer classifier. The Autoencoder has 8 hyperparameters: learning rate, mini batch size, dropout rate, L2 regularization, learning decay rate, training epochs, and the number of hidden units in the first and second layers (the size of the

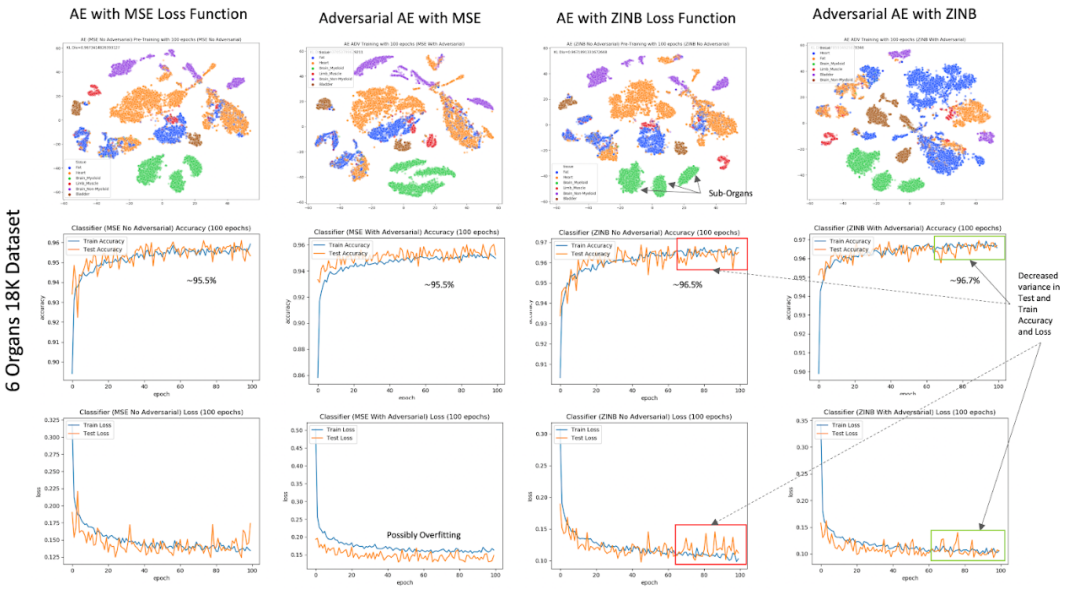
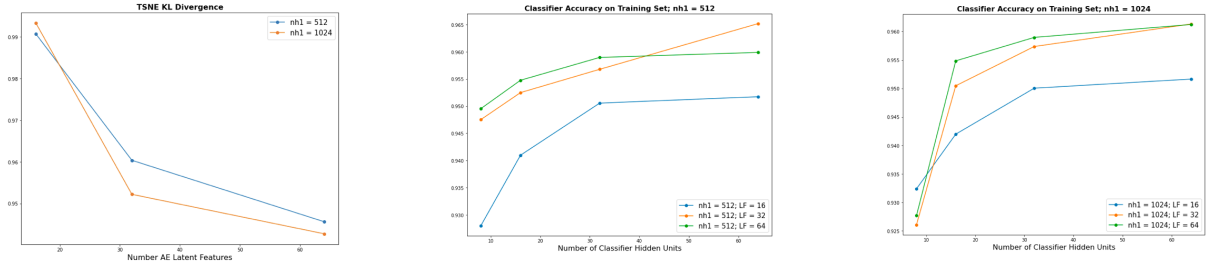


Figure 3: Autoencoder Model explored: (A) AE with MSE loss function (B) Adversarial AE with MSE loss function (3) AE with ZINB loss function (C) Adversarial AE with ZINB loss function

last AE layer is the same as the first). The second layer of the AE also corresponds to the number of latent features which is inputted into the classifier.

Let us denote the number of nodes in the first and second layers of the AE as num_hidden_1 and num_latent_features, respectively, and denote the number of units in the classifier as num_class. We search over the following parameters, and fix all others: num_hidden_1 = [512, 1024], num_latent_features = [16,32,64], and num_class = [8, 16, 32, 64]. The results of these networks from the training data are plotted on Fig. 5. To compare how well the various AE's are clustering, we compute the TSNE KL Divergence for each model. Fig. 5a shows that models of higher complexity (more nodes) tend to cluster better, which is an intuitive result. Figs. 5(b-c) show the classification accuracy for predicting cell types on the training data for various num_hidden_1, num_latent_features, and num_class. Again, models with higher complexity fit to the training set better.



(a) TSNE Kl Divergence

(b) Classification Accuracy, nh1 = 512

(c) Classification Accuracy, nh1 = 1024

Figure 4: (a) TSNE KL Divergence plotted for AE of varying layer sizes. (b) Classification accuracy for AE num_hidden_1 = 512, and varying latent feature and classifier layer sizes. (c) Classification accuracy for AE num_hidden_1 = 1024, and varying latent feature and classifier layer sizes.

5.3 Training and Testing

The models were trained on three different datasets to evaluate performance. We started with a five organ dataset with 3K cells, then expanding to a six organ dataset with 18K cells and the final training was completed on 18 organ data set with 32K cells (Fig 3 and Fig 5A). As expected we achieved 90-96 percent accuracy on the training and the test set. However, while making predictions using the pre-trained weights of the 32k cells models on the same training dataset as a checkpoint, we realized that our predictions seemed to be limited to 6 organs as can be observed from confusion matrix (Figure 5B). Upon further investigation, we isolated the error to a label generation logic which was not adjusted for the final execution when we train on the 32 k cells dataset. We are in the process to re-training the model and believe that retraining will give us better prediction. A key learning from this was to built in automated checks for model parameters (expected vs current) so as to catch such errors early in the process. We plan to add this automated model validation to our source code.

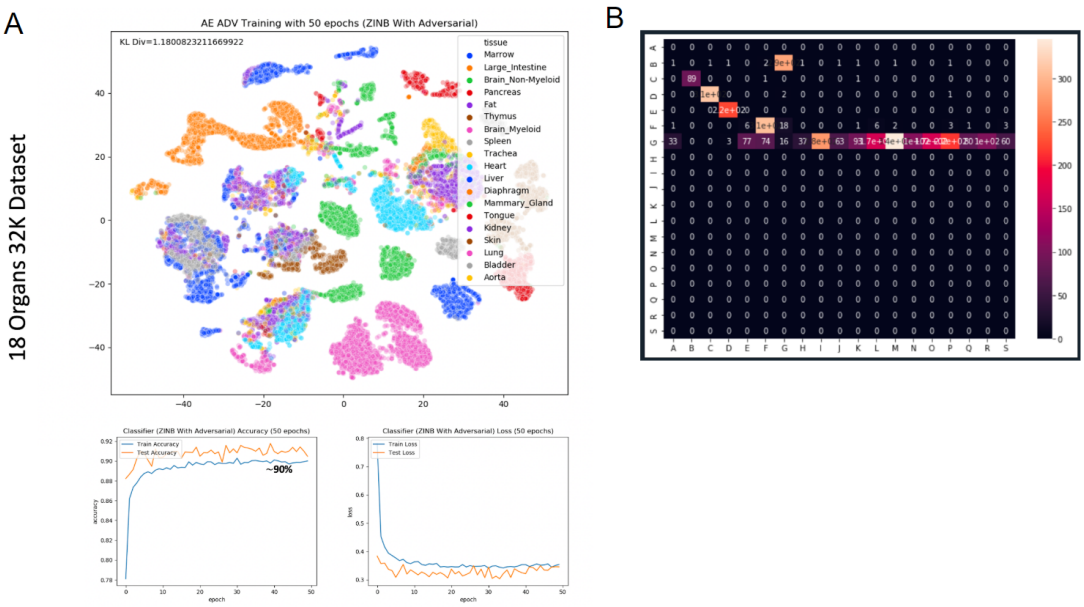


Figure 5: (A) Tsne plot and accuracy and loss curve of 32k cells training dataset comprising 18 organs (B) Confusion matrix of the prediction using the pre-trained weights of the training model on the training dataset

6 Conclusion/Future Work

In summary, we implemented an autoencoder to tackle the challenges of batch effects, dropout, technical noise, and curse of dimensionality posed by scRNA-seq analysis. Upon exploration, we found the autoencoder of ZINB loss function along with adversarial network to perform the best as it has a stabilizing effect on the variation in the loss and accuracy curve while the accuracy for the four models we explored were comparable. We also implemented the multi-class classifier that takes the input of the autoencoder to make predictions of the cell types of the dataset in an automated manner.

As an immediate next step, we hope to perform a parameter search over not only the neural network architectures but also vary the regularization parameters and see if we get better results. We also hope to retrain the network with the 32 k dataset with the right 18 organ labels so that we can transfer the annotation from the mouse training set to human test set.

7 Contribution

C.C proposes the idea of the project, and takes a lead on data collection and writing basic source code, P.S takes a lead on exploring various network architecture, X.G takes a lead on hyperparameter search. All team members contributed equally to the final form of the project.

References

[1] Mereu, E., Lafzi, A., Moutinho, C. et al. Benchmarking single-cell RNA-sequencing protocols for cell atlas projects. *Nat Biotechnol* (2020).

[2] Zheng, J., Wang, K. Emerging deep learning methods for single cell RNA-seq data analysis. *Quantitative Biology* **7**, 247-254 (2019)

[3] Shaham, U., Stanton, K. P., Kluger, Y., et al. Removal of batch effects using distribution=matching residual networks. *Bioinformatics* **33** 2539-2546 (2017).

[4] Eraslan, G., Simon, L.M., Mircea, M. et al. Single-cell RNA-seq denoising using a deep count autoencoder. *Nat Commun* **10** ,390 (2019).

[5] Ding, J., Condon, A., Shah, S.P. Interpretable dimensionality reduction of single cell transcriptomic data with deep generative models. *Nature Communication* **9** (2002).

[6] Lopez, R., Regier, J., Cole, M.B., Jordan, M.I., Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature Methods* **15** 1053-1058 (2018).

[7] Dincer, B., Janizek, J., Lee, S.I. et al. Adversarial Deconfounding Autoencoder for Learning Robust Gene Expression Embeddings. bioRxiv (2020).

[8] Schaum, N., Karkanias, J., Neff, N.F. et al. Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris. *Nature* 562, 367–372 (2018).

[9] Han, X., Zhou, Z., Guo, G., et al. Construction of a human cell landscape at single-cell level. *Nature* **581**, 303-309 (2020).

8 Appendix

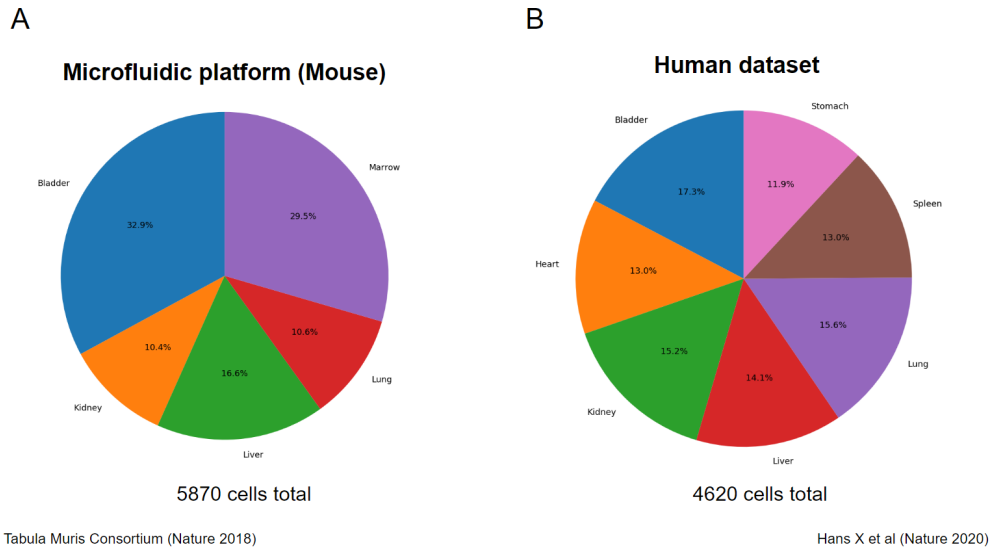


Figure 6: (A) Testing dataset of 10X Genomics based single cell experiments (B) Testing dataset of human cell

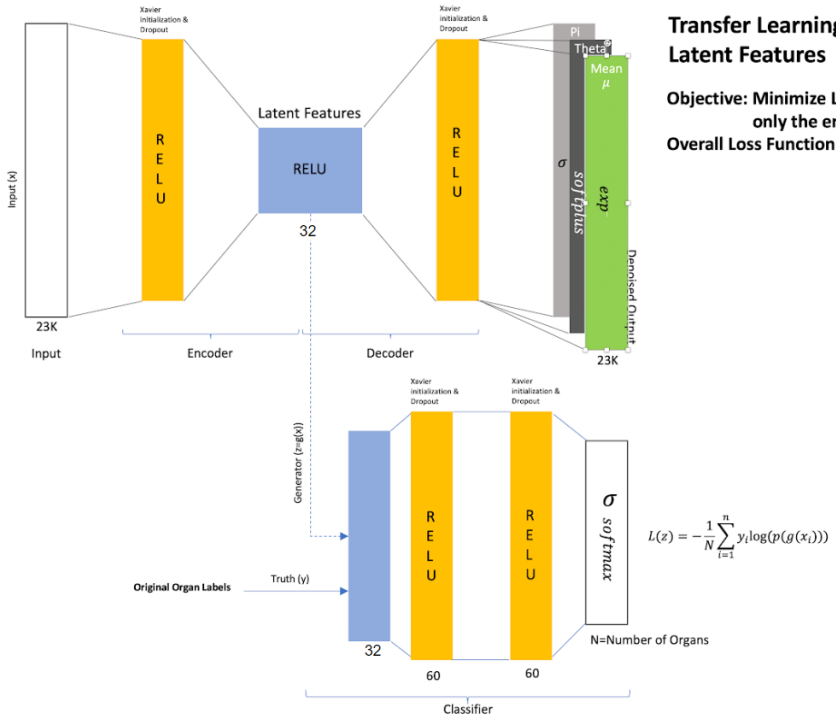


Figure 7: Latent features from autoencoder are fed into multiclass classifier