Published online 25 May 2020

NAR Genomics and Bioinformatics, 2020, Vol. 2, No. 2 **1**
doi: 10.1093/nargab/lqaa039

Deep soft *K*-means clustering with self-training for single-cell RNA sequence data

Liang Chen[1,†], Weinan Wang[1,†], Yuyao Zhai[2] and Minghua Deng[1,3,*]

[1]School of Mathematical Sciences, Peking University, Beijing 100871, China, [2]Mathematical and Statistical Institute, Northeast Normal University, Changchun 130024, China and [3]Center for Quantitative Biology, Peking University, Beijing 100871, China

Received January 31, 2020; Revised April 24, 2020; Editorial Decision May 12, 2020; Accepted May 14, 2020

ABSTRACT

Single-cell RNA sequencing (scRNA-seq) allows researchers to study cell heterogeneity at the cellular level. A crucial step in analyzing scRNA-seq data is to cluster cells into subpopulations to facilitate subsequent downstream analysis. However, frequent dropout events and increasing size of scRNA-seq data make clustering such high-dimensional, sparse and massive transcriptional expression profiles challenging. Although some existing deep learning-based clustering algorithms for single cells combine dimensionality reduction with clustering, they either ignore the distance and affinity constraints between similar cells or make some additional latent space assumptions like mixture Gaussian distribution, failing to learn cluster-friendly low-dimensional space. Therefore, in this paper, we combine the deep learning technique with the use of a denoising autoencoder to characterize scRNA-seq data while propose a soft self-training *K*-means algorithm to cluster the cell population in the learned latent space. The self-training procedure can effectively aggregate the similar cells and pursue more cluster-friendly latent space. Our method, called 'scziDesk', alternately performs data compression, data reconstruction and soft clustering iteratively, and the results exhibit excellent compatibility and robustness in both simulated and real data. Moreover, our proposed method has perfect scalability in line with cell size on large-scale datasets.

INTRODUCTION

Cells are the basic units of growth and development of organisms, and each cell has its own unique biological function (1). The heterogeneity of genetic information, such as transcriptome, leads to heterogeneity among cells. Traditional bulk cell RNA sequencing is performed at a multicell level. This means that the resultant sequencing data are the average expression of multiple cells, thus missing information on individual heterogeneity (2). In recent years, the rapid development of single-cell RNA sequencing (scRNA-seq) technology has made it possible to obtain the transcriptional expression of each cell, thereby extracting the heterogeneity of cells at the RNA level (3–5). However, due to the technical difficulties such as the inefficiency of RNA capture during sequencing, the bias of PCR amplification and the depth of sequencing, single-cell transcriptional expression profiles usually have substantial zero elements (6,7). These kinds of high-dimensional and sparse noisy data have a highly non-linear complex structure, which makes further subsequent statistical analysis challenging. Furthermore, droplet-based sequencing technologies have been able to effectively profile tens of thousands of cells in a single experiment, which undoubtedly places higher requirements on the development of new analytical methods (8,9).

Analysis of scRNA-seq data can conduct cell identification and enable more accurate determination of cell types and relationships (10). An essential procedure in studying the structure of cell subsets is cell clustering, since the subsequent analysis, such as identifying marker genes (11,12), studying different stages of cell cycle (13,14), elucidating cell–cell communication between different cell types (15,16) and constructing gene expression regulation network (17,18), depends on specific cell populations. In the past few years, a great deal of clustering methods for single-cell transcriptional expression profiles have been developed. CIDR is a method of combining dropout events' imputation and clustering, which uses hierarchical clustering on the top coordinates through PCoA on the dissimilarity matrix (19). SIMLR learns the cell similarity measure by multi-kernel learning and graph diffusion technique. Then, the learned latent representation can be used for spectral clustering (20). Phenograph and Seurat are based on the shared nearest neighbor graph and they use the Louvain algorithm to detect cell community (21,22). RaceID, which is customized for identifying rare cell types, proposes that replac-

*To whom correspondence should be addressed. Tel: +86 13522856599; Email: dengmh@pku.edu.cn
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.

ing *K*-means with *K*-medoids can improve cluster performance (23). SC3 is a consensus ensemble clustering method that first uses PCA and Laplacian transformation to reduce gene dimension, and then measures pairwise cell distances like Euclidean, Pearson and Spearman distances (24). The consensus matrix is constructed by applying *K*-means clustering to these six projections and finally taken as the input of hierarchical clustering. SAFE is another consensus clustering method that integrates clustering results from *t*-SNE, CIDR, Seurat and SC3 (25). One common characteristic of the above-mentioned methods is that they usually directly learn a similarity or distance representation based on the original noisy data matrix or reduced matrix through simple linear dimension reduction techniques, and then apply the hard clustering methods. However, on the one hand, dimension disaster and high noise make distance measurement no longer accurate and linear dimension reduction methods cannot capture non-linear structures hidden in scRNA-seq data. On the other hand, separating dimension reduction from clustering often leads to spurious clustering results.

Recently, the deep learning technologies have been applied to computational biology and they exhibit superior performance over traditional machine learning algorithms in most supervised and unsupervised learning scenarios (26,27). Autoencoder is a neural network that can learn an efficient compression of data through encoder and decoder in an unsupervised fashion (28). The reconstruction error is generally defined as loss function to train the autoencoder. It is most remarkable that autoencoder can realize non-linear dimension reduction of high-dimensional data in the essential latent space and reconstruct the denoised data in the meantime. Ding *et al.* (29) proposed an interpretable dimensionality reduction method for scRNA-seq data using the deep generalized model, which assumes that the gene expression vector follows a Student's *t*-distribution given the low-dimensional latent feature vector with the multivariate Gaussian distribution as prior. Considering the sparsity characteristic of scRNA-seq data, Eraslan *et al.* (30) proposed a deep count autoencoder network called DCA to denoise the original single-cell data. They take the count distribution, overdispersion and sparsity of the data into account using a zero-inflated negative binomial (ZINB) model that can capture the non-linear gene–gene dependencies. This modeling idea inspired several related works reporting on scRNA-seq data analysis. Grønbech *et al.* (31) proposed a variational autoencoder framework called scVAE for single-cell gene expression count data clustering. They assume that the data points in latent space follow the Gaussian or mixture Gaussian distribution, and then derive the corresponding variation lower bound as part of loss function. On the one hand, the complex latent space actually falls on inenarrable manifold but not mixture Gaussian distribution. On the other hand, variational inference model is often tricky and hard to optimize since the approximate lower bound is often trapped in the local minimum. Tian *et al.* (32) developed a single-cell model-based deep embedded clustering method called scDeepCluster that combines DCA modeling with DEC clustering algorithm (33,34) and achieves clustering while reducing dimensions. However, scDeepCluster ignores the pairwise distance between cells and does not consider the affinity constraint of similar cells. Moreover, scDeepCluster does not preselect some informative genes as input features, which not only loses a part of the clustering accuracy but also leads to high time consumption and memory requirements. Li *et al.* (35) proposed a deep embedding clustering algorithm called DESC similar to scDeepCluster, but they just utilize the deep autoencoder to pretrain the data construction and actually separate the clustering procedure and data denoising, which cannot learn more cluster-friendly latent space. Besides, DESC uses the traditional MSE loss as data reconstruction error and also ignores the distance between similar cells, which cannot preserve the global and local structure of data well.

To the best of our knowledge, although some existing deep learning-based clustering methods combine dimension reduction and clustering, like scDeepCluster, they usually neglect the pairwise distance of similar cells, resulting in failure to learn more cluster-friendly latent space with high confidence. Therefore, in this paper, we first combine the modeling idea of DCA (30) and assume that the scRNA-seq data denoised by autoencoder can be depicted by NB models with or without zero inflation (NB and ZINB). We utilize the negative log-likelihood function as the data reconstruction loss instead of traditional MSE (36) to capture the global probabilistic structure of data. Meanwhile, we propose a weighted soft *K*-means clustering algorithm with inflation operation for data points in the latent space, which aims to realize the soft clustering process instead of hard clustering. Moreover, we introduce a new constraint for the data representation, which combines the idea of *t*-SNE and self-training strategy in DEC, to enhance the association between similar cells. Our method aggregates data modeling, dimensionality reduction and cell clustering by alternately updating data denoising and clustering processes. We apply our method on both simulation datasets and real datasets and the results show that our proposed method outperforms other state-of-the-art scRNA-seq data clustering algorithms considering accuracy, robustness and scalability.

MATERIALS AND METHODS

Data pre-processing

We collect the scRNA-seq count data matrix by means of the quality control process. Suppose the single-cell expression count matrix is Y, where Y_{ij} $(1 \le i \le n, 1 \le j \le p)$ represents the expression of jth gene in the ith cell. We next discard the genes that have expression values in less than one cell and then filter the cells without gene expression. Considering neural network numerical optimization stability, we need to transform discrete data into continuous smooth data. Concretely, we first normalize the count matrix $Y_{n\times p}$ through dividing each row by its row sum and multiplying it by the median of total expression values of all cells, and then we take a natural log transformation on data. Since most genes have little information to identify and describe cell types, we pick top m highly variable genes according to their normalized dispersion values' ranking calculated by scanpy package (37). Finally, we transform the logarithm data into z-score data, which implies that each selected gene has zero mean and unit variance. This normalized data matrix, recorded as $X'_{n\times m}$, is used for the neural network in-

put and its corresponding original count matrix, recorded as $X_{n\times m}$, is used for data modeling.

Zero-inflated negative binomial autoencoder model

Since the variance of gene expression in samples is often larger than its corresponding mean, we assume X_{ij} follows NB distribution with mean parameter μ_{ij} and dispersion parameter θ_{ij} instead of the Poisson distribution, namely

$$P_{\mathrm{NB}}(X_{ij}|\mu_{ij},\theta_{ij}) = \frac{\Gamma(X_{ij}+\theta_{ij})}{\Gamma(X_{ij}+1)\Gamma(\theta_{ij})} \times \left(\frac{\theta_{ij}}{\theta_{ij}+\mu_{ij}}\right)^{\theta_{ij}} \times \left(\frac{\mu_{ij}}{\theta_{ij}+\mu_{ij}}\right)^{X_{ij}}. \quad (1)$$

The frequent dropout events resulting from RNA capture inefficiency cause the count matrix to contain amounts of zero elements; thus, we model the data distribution as a mixture of zero component and NB distribution, namely ZINB, which can be written as

$$P_{\mathrm{ZINB}}(X_{ij}|\pi_{ij},\mu_{ij},\theta_{ij}) = \pi_{ij}\delta_0(X_{ij}) + (1-\pi_{ij}) \times P_{\mathrm{NB}}(X_{ij}|\mu_{ij},\theta_{ij}), \quad (2)$$

where π_{ij} is the weight coefficient of the point mass at zero. Now (π_{ij}, μ_{ij}, θ_{ij}) makes up the parameters to be estimated. With the complex gene–gene interactions among genes, these parameters are not independent from the statistical aspect, and they are more likely to fall on a low-dimensional manifold, which cannot be captured by a simple linear model. So, we utilize a neural network like autoencoder to approximate the underlying parameter space.

In this paper, we use the denoising autoencoder to train the corrupted scRNA-seq data and recover the underlying undistorted transcriptional profiles. Unlike the traditional autoencoder setting that has one output layer, we have three output layers to estimate the three sets of parameters, dropout rate, mean value and dispersion of ZINB model separately. Assume latent representation is Z and the decoder output is D; then the architecture of our autoencoder model can be written as

$$Z = \phi_{\mathrm{w}}(X'), \quad (3)$$

$$D = \varphi_{\mathrm{w}}(Z), \quad (4)$$

$$\pi = \mathrm{sigmoid}(DW_\pi), \quad (5)$$

$$\mu = \exp(DW_\mu), \quad (6)$$

$$\theta = \exp(DW_\theta), \quad (7)$$

where ϕw and φw represent the encoder function and decoder function, respectively, and these W represent network weight parameter matrices. Besides, π uses the sigmoid activation function because we need to limit the dropout probability to the range from 0 to 1. Since the mean and dispersion parameters are non-negative, we chose the exponential function as their corresponding activation function. As we all know, the library size is critical in studying scRNA-seq count data; thus, we normalize the sum of output μ to original total count size for each cell to avoid overfitting. For the reconstruction loss function, we naturally take the negative log-likelihood of ZINB distribution as loss function, which is given by

$$L_1(\pi,\mu,\theta|X) = -\log(P_{\mathrm{ZINB}}(X|\pi,\mu,\theta)), \quad (8)$$

$$L_1(\pi,\mu,\theta|X) = -\sum_{i=1}^{n}\sum_{j=1}^{m}\log(P_{\mathrm{ZINB}}(X_{ij}|\pi_{ij},\mu_{ij},\theta_{ij})). \quad (9)$$

Based on statistical theory, we should minimize L_1 to obtain parameter estimation.

Self-training weighted *K*-means clustering

In the last subsection, we note that parameters of the ZINB model distribute on the low-dimensional manifold consisting of latent variable Z. Therefore, instead of performing clustering on the denoised data space, we can implement the clustering algorithm in the learned embedding space. Suppose there are K clusters with centers υ_r ($1 \le r \le K$) in the latent space. Then, a direct approach would involve applying the K-means algorithm to cluster the latent representation Z. Here, we use a weighted soft K-means model to realize the clustering in the latent space with objective function

$$L_2(\upsilon, Z) = \sum_{i=1}^{n}\sum_{r=1}^{K} w_{ir}||Z_i - \upsilon_r||^2. \quad (10)$$

This continuous clustering loss function can benefit from the efficiency of stochastic gradient descent. In fact, when latent space representation Z_i ($1 \le i \le n$) and weights w_{ir} ($1 \le i \le n$, $1 \le r \le K$) are known, cluster center υ_r ($1 \le r \le K$) has an explicit closed-form solution for minimizing L_2 loss, which is

$$\upsilon_r = \frac{\sum_{i=1}^{n} w_{ir} Z_i}{\sum_{i=1}^{n} w_{ir}}. \quad (11)$$

We can see that the cluster centers are weighted sum of those low-dimensional representations of data points. We naturally want to increase the weight of those data points closer to the cluster center, while reducing the contribution of other data points farther away from the cluster center. Therefore, the weights w_{ir} should be a decreasing function with the distance between the data point and the cluster center. To a certain extent, we transform the hard cluster label allocation to assign labels according to affinity probability. In this paper, we consider utilizing the Gaussian kernel function as weight measure since the exponential function can smooth the gradient descent optimization process. Specifically,

$$\tilde{w}_{ir} = \frac{\exp(-||Z_i - \upsilon_r||^2)}{\sum_{k=1}^{K}\exp(-||Z_i - \upsilon_k||^2)}. \quad (12)$$

In order to speed up the algorithm convergence, we take the inflation operation on the weights, inspired by the idea of Markov clustering algorithm (38),

$$w_{ir} = \frac{\tilde{w}_{ir}^{\alpha}}{\sum_{j=1}^{K}(\tilde{w}_{ij}^{\alpha})}, \quad (13)$$

where α is a hyperparameter that is >1. Its default value in our codes is 2. Obviously, this soft K-means clustering approach forces the data point to move closer to its closest cluster center. However, this procedure neglects the pairwise distance and movement of similar cells. We naturally pursue similar points to be clustered to the same cluster. Thus, we design a specific KL divergence loss function to preserve and strengthen the association between similar cells. First, we use the t-distribution kernel function to describe the pairwise similarity among data points in latent space like in t-SNE (39), because the tail of this distribution is heavier than Gaussian distribution and it allows moderate distances in the high-dimensional space to be modeled by much larger distance in the low-dimensional space, which can prevent squeezing different clusters together in the latent space, as

$$p_{j|i} = \frac{(1 + ||Z_i - Z_j||^2/t)^{-(t+1)/2}}{\sum_{k \neq i}(1 + ||Z_i - Z_k||^2/t)^{-(t+1)/2}}, \quad (14)$$

where $p_{i|i} = 0$ $(1 \leq i \leq n)$. Inspired by DEC (33), we propose to alternately refine the pairwise data points by learning from their high similarity with the help of an auxiliary target distribution. This target distribution should strengthen the affinity between similar data points and put less emphasis on those pairwise data points with low similarity. Specifically, we take the auxiliary distribution below, as

$$q_{j|i} = \frac{p_{j|i}^2 / \sum_{i \neq j} p_{j|i}}{\sum_{k \neq i}(p_{j|i}^2 / \sum_{i \neq j} p_{j|i})}. \quad (15)$$

With these two similarity distribution functions, we define another loss function as follows:

$$L_3(Z) = KL(q||p) = \sum_i \sum_j q_{j|i} \log \frac{q_{j|i}}{p_{j|i}}. \quad (16)$$

We utilize this self-training strategy to learn a more cluster-friendly latent space. Having finished model construction, we now summarize the three main components: denoising autoencoder based on ZINB likelihood, weighted soft K-means clustering and the self-training strategy for pairwise points in latent space. We can see that the latent space representation Z_i $(1 \leq i \leq n)$ is contained in both L_2 and L_3; that is, Z_i $(1 \leq i \leq n)$ and υ_r $(1 \leq r \leq K)$ depend on each other. In the process of algorithm implementation, we found that training L_2 and L_3 separately poses a risk of collapse of different clusters. Therefore, we integrate them together and the total objective loss function is given as

$$L(\pi, \mu, \theta, \upsilon, Z|X) = L_1 + \gamma L_2 + \lambda L_3, \quad (17)$$

where the hyperparameters γ and λ are the coefficients that control the relative importance of L_2 and L_3. In this paper, our method is implemented in Python3 using Tensorflow, which can perform automatic differentiation and update of variables. The default values of γ and λ are both taken as 0.001 during the following experiments. We select top 500 (i.e. $m = 500$) highly variable genes as network input by default. The hidden layer size of encoder network is 256 and 32, and the decoder network has the reverse setting of encoder. The bottleneck layer (namely latent space) has a size of 32 and the minibatch size is set to 256 during the training process. We utilize the Adam optimizer with a learning rate of 0.0001 to update the neural network parameters and cluster centers by a back-propagation algorithm. As for the model training strategy, we first pretrain the L_1 loss for 1000 (default) epochs and then initialize the cluster centers by standard K-means algorithm in the latent space; finally, we train the whole model L until the cluster membership assignment no longer changes. We have summarized the whole clustering algorithm in Supplementary Algorithm S1 and the general workflow of our method is displayed in Figure 1. For convenience, we call our model scziDesk (single-cell zero-inflated deep soft K-means). When we replace ZINB distribution with NB distribution, we name it as scDesk. In the 'Results' section, we will show the superior clustering performance of our methods through simulation study and real dataset analysis.

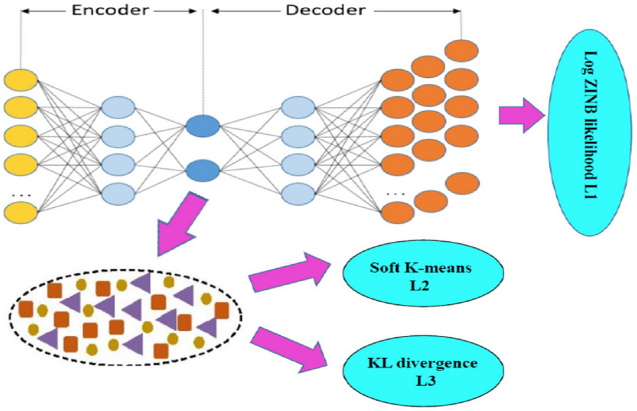

Figure 1. The neural network schematic diagram of scziDesk. The encoder and decoder are symmetrical structures, and the outputs are three sets of parameters, dropout rate, mean value and dispersion value, in ZINB modeling. In latent space, embedded points are clustered using a weighted soft K-means clustering algorithm with self-training procedure. The self-training strategy aims to aggregate the similar cells and pursue more cluster-friendly latent space.

RESULTS

In this section, we test scDesk and scziDesk in both simulation and real datasets. For simulation datasets, we assigned cells into different cell types before generating gene expression. For real datasets, we used the cell type information provided by authors. They determined cell types by experiments or other biological methods. All of this can be regarded as the reference gold standard of clustering. Then, we apply our method to get a clustering result and compare its similarity with the true label using two criteria: adjusted Rand index (ARI) (40) and normalized mutual information (NMI) (41) (see the Supplementary Data for details). The higher the ARI and NMI values, the better the clustering effect. Moreover, we compare the clustering performance with existing customized scRNA-seq data clustering algorithms, including CIDR (19), SIMLR (20), RaceID3 (23,42), SOUP (43) and a deep learning-based method called scDeepCluster (32). Other methods, such as Seurat (22), SC3 (24) and so on, are not in consideration

of comparison. Seurat is a useful and efficient package for biologists to analyze single-cell data. However, due to a lot of parameters, especially 'resolution' parameter, being elaborately selected by users, Seurat usually tends to overestimate the cluster number that cannot perform well under default parameters. For the sake of fairness, we do not show the clustering performance of Seurat here. SC3 is a consensus method that may achieve perfect performance in small datasets, but it is extremely time consuming and does not scale to large-scale datasets with >5000 cells. We will show the additional comparison with Seurat and SC3 in the 'Discussion and Conclusion' section. Besides, in order to show the advantages of ZINB or NB modeling, we additionally regard the traditional MSE autoencoder with our soft self-training *K*-means clustering algorithm as a comparison.

Simulation study

In simulation analysis, we first used an R package called 'splatter' (44) to generate simulation datasets. Our experiments can be divided into two parts, one involves cell clusters with the same size and the other involves cell clusters with different sizes. We call them 'balanced experiments' and 'imbalanced experiments', respectively. In the first part, we generate datasets with three, four, five, six and seven clusters, and each cluster with 500 cells. In the second part, we generate datasets fixed at five clusters and total 2500 cells, but the cluster group size gives an equal ratio sequence where ratio ranges from 0.6 to 1.0, and smaller ratio means larger difference in cluster size. For both, we set the dropout rate ranging from 5% to 25% (dropout.type = 'experiment', dropout.shape = −1, de.facScale = 0.2 and dropout.mid ranges from −1.5 to 0.5; the other parameters are set to default values) to simulate the 'dropout' event, and the gene number is fixed at 2500 for the convenience of calculating. For every scenario, we generate 10 datasets to inspect the average performance among different methods by calculating their median ARI and NMI values.

First, we investigated total performance of the eight methods by the box plot for balanced experiments and imbalanced experiments (Figure 2A). From the box plot, the median values of ARI for scDesk and scziDesk are 0.76 and 0.85 in the balanced experiment, respectively, which are obviously higher than those for CIDR (0.21), SOUP (0.55), SIMLR (0.20), RaceID (0.19), scDeepCluster (0.16) and MSE (0.44). We found that the ARI value of scziDesk is concentrated at the top of the box and >0.75. Only a few results decrease below 0.5 as a consequence of the difficulty of clustering from the increase of dropout rate. Meanwhile, other methods like SIMLR and RaceID3 never achieved an ARI value >0.5 for all parameters. Although SOUP performs well in some datasets, its ARI value is still a bit lower than that of scziDesk, and it also has a large variance without the ability to handle dropout events. We can get the same conclusion from comparing NMI values (Supplementary Figure S1) and from the imbalanced experiment results.

We then applied line graph to find the performance change through different parameters. Every subfigure in Figure 2B shows a change in ARI value with increasing dropout events. SczDesk performs quite well (ARI > 0.9) when the dropout rate is small. Although the performance

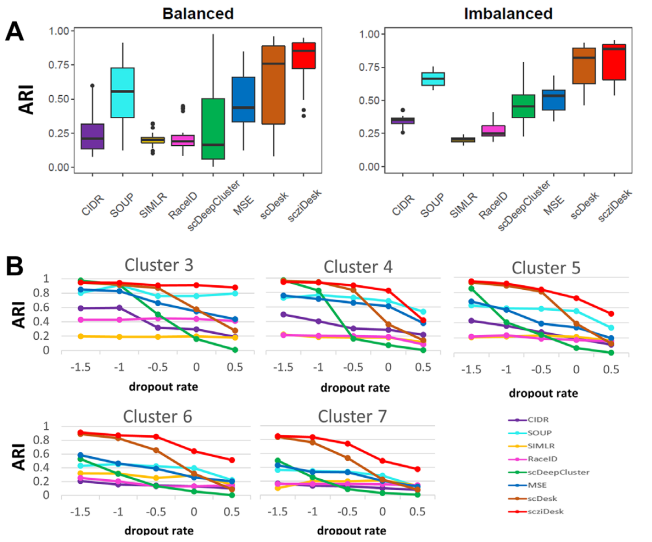

Figure 2. Simulation dataset analysis. (**A**) Box plot of ARI values in balanced and imbalanced experiments. (**B**) Change of ARI values with the increasing dropout rate in a balanced experiment.

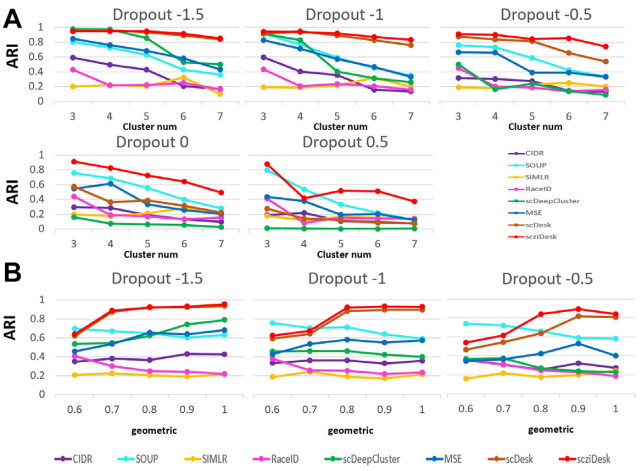

Figure 3. Simulation dataset analysis. (**A**) Change of ARI values with the increasing cluster number in a balanced experiment. (**B**) Change of ARI values with the increasing geometric ratio in an imbalanced experiment.

of scziDesk decreases in relation to the dropout percentage of simulation data, those of all other methods also decrease and it still performs better than other methods in most scenarios. Figure 3A fixes the dropout rate in every subfigure and illustrates the change in ARI value with increasing cluster numbers. When the dropout rate is low, our two methods perform extremely well and are nearly unaffected by cluster number, while the performances of CIDR, SOUP and RaceID drop significantly when cluster number increases. When the dropout rate increases, scziDesk can handle a small number of clusters well and still always remains superior to the other methods. By comparison, scDesk performs very poorly with a high dropout rate, because it does not deliberately model the zero-inflation phenomenon. The line graph for imbalanced experiments can be seen in Figure 3B. When we fix the dropout rate and decrease the ge-

ometric ratio, scziDesk gives the highest ARI value among eight methods, fully demonstrating that scziDesk can better detect rare clusters. The NMI values (Supplementary Figures S1 and S2) also support our conclusion, whether in balanced or imbalanced dataset.

From these two groups of simulation datasets, our methods always outperform the other six in different conditions (cluster number, dropout percentage, cluster size). Moreover, scziDesk is a little better than scDesk because its performance still exceeds that of others at high dropout rate and large cluster numbers. This is in line with our expectations because the scziDesk considers the zero inflation and adds the corresponding zero-inflated probability in likelihood function.

Analysis of real data

For real data analysis, we select 20 real datasets from recently published papers of scRNA-seq experiments. It is worth pointing out that a team from Stanford University generated mouse scRNA-seq data for 20 organs by Smart-seq2 sequencing and 12 organs by 10x Genomics sequencing (45). We selected nine datasets from it, five from Smart-seq2 (beginning with Quake_Smart-seq2, hereinafter 'QS' in this paper) and four from 10x Genomics (beginning with from Quake_10x, hereinafter 'Qx' in this paper). All 20 datasets originate from different species, including plant, mouse and human, and different organs, e.g. brain, kidney or lung. Moreover, these datasets were acquired by different sequencing platforms (CEL-seq2, Drop-seq, inDrop, 10x Genomics and Smart-seq2). Their basic information including cell number, cell type number and reference paper can be found in Supplementary Table S1. In the next several comparison experiments, we assumed to know the true cell type number before running all methods.

In order to ensure the reliability of results for each method, we ran them 10 times by setting 10 random seeds 1111, 2222, …, 9999, 10000. Once 10 ARI and NMI values are obtained, we calculated their median as the evaluation value of performance for each method. First, Figure 4A and B shows the overall performance of eight methods in 20 real datasets. From the box plot, we can see that the ARI (NMI) value changes among 20 datasets and scziDesk has the highest mean value with lowest variance. The dot plot (Figure 4C) shows the specific performance of eight methods in 20 datasets. The size of scatter stands for the ARI ranking among eight methods and the color stands for ARI value. From the figure, we can see that our two methods are all purple or red with the biggest size, which implies they rank in the top four among eight methods, and their ARI values are >0.6 in all datasets. Overall, SIMLR gives relatively competitive performance among four traditional statistics-based methods. Nevertheless, it still contains some gray scatter with the smallest size, which means it performs the worst among eight methods and has ARI value <0.2. Other methods, such as CIDR and RaceID3, perform very poorly and always rank the last among the eight methods. The result of comparing NMI value (Supplementary Figure S4A) is almost the same as that in ARI. We also applied histogram to compare all methods for each dataset (Supplementary Figures S5 and S6); the conclusion is also

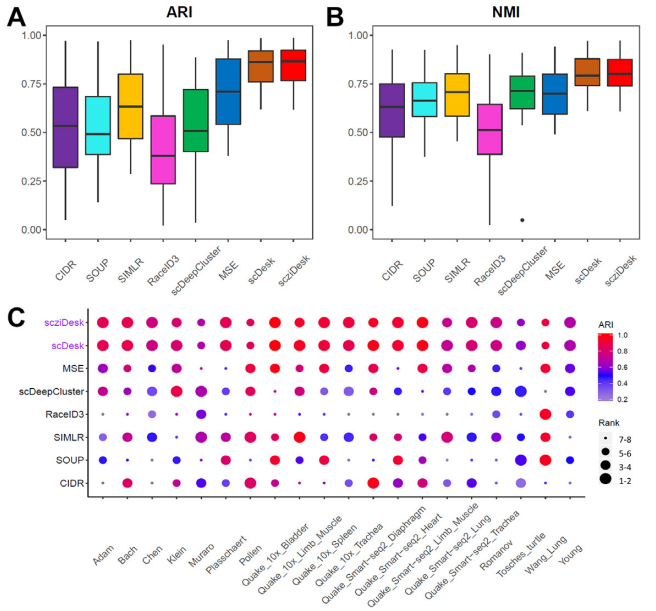

Figure 4. Real data analysis. (**A**, **B**) Box plot of ARI and NMI values in 20 real dataset experiments, respectively. (**C**) Dot plot of 20 real datasets. Every point on the *x*-axis stands for a dataset and on the *y*-axis a method. The scatter reflects the corresponding performance of a method in a dataset, where the color stands for its ARI value and the size stands for its ranking according to ARI value among the eight methods. The gray scatter means that its ARI value is <0.2.

the same. Therefore, comprehensively comparing the performance over 20 datasets, both of our methods achieve better performance than other algorithms, especially scziDesk.

To intuitively compare the accuracy of clustering methods, we plotted the 2D visualization graph through data after using dimension reduction methods, such as *t*-SNE (39) and UMAP (46). The latent space in scDesk and scziDesk is a low-dimensional embedded representation of the high-dimensional input data. Here, we first attained the latent space representation of 32 dimensions and then applied *t*-SNE with the default parameter to visualize the embedded points in the 2D plane. The 2D visualization of scDeepCluster and MSE can be obtained in the same way. For other four traditional methods, we used the procedure and code provided by the author with default parameter to get the 2D visualization plot. The visualization results of two datasets are summarized in Figure 5. Figure 5A is for 'Qx_Bladder' dataset with four cell types: bladder, bladder urothelial, leukocyte and endothelial cells are dyed with blue, red, yellow and green, respectively. Figure 5B is for 'QS_Limb_Muscle' dataset with six cell types: B cell, endothelial cell, macrophage, mesenchymal stem cell, skeletal muscle satellite cell and T cell are dyed with light blue, navy, green, orange, purple and red, respectively. For 'Qx_Bladder', we can see that scziDesk can separate four cell types clearly in 2D plane with almost no overlap between different types. Other methods, especially CIDR and SOUP, fail to separate cells into four parts. For another dataset 'QS_Limb_Muscle', we can easily see that all scatters are divided and aggregated into six groups from the last two subpictures in Figure 5B, while other methods fail to

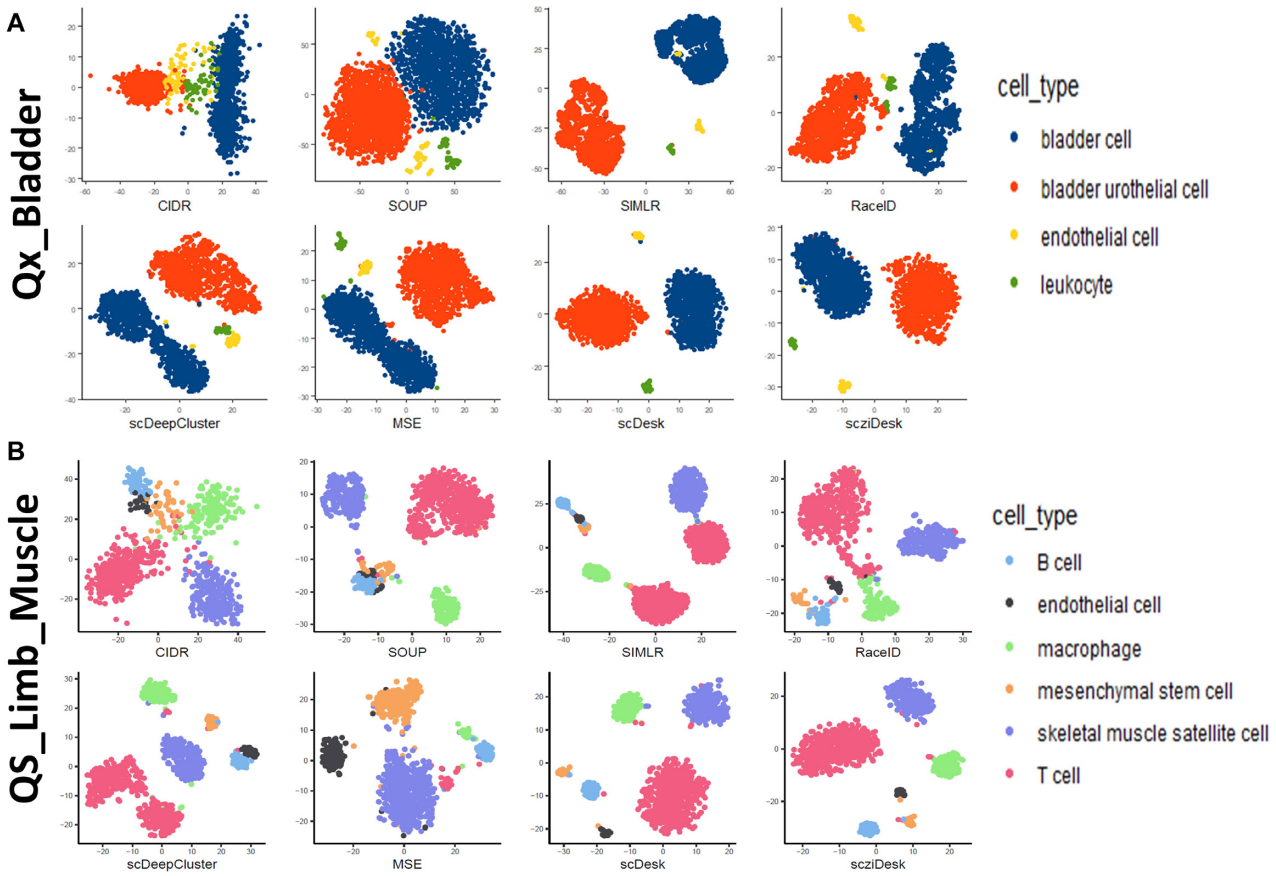

Figure 5. Visualizing high-dimensional data in 2D plane by eight methods. Each color stands for one cell type. (**A**) Results of 'Qx_Bladder' dataset. (**B**) Results of 'QS_Limb_Muscle' dataset.

distinguish B cell, endothelial cell and mesenchymal steam cell types clearly. Moreover, the red scatters, which stand for T cells, are divided into two separate parts by SIMLR and scDeepCluster. Only our two methods can distinguish cells from different cell types and concentrate on the same type of cells together in the visualization plot simultaneously. More examples are shown in Supplementary Figure S4B. Therefore, from an intuitive perspective, our methods are superior to the others in both simple datasets with few cell types and complex datasets with many cell types.

Robustness of methods

We further validated the stability and robustness of our methods by two artificial experiments, including downsampling and dropout. We selected 8 out of 20 real datasets for these experiments, which we term 'Adam, Bach, Plasschaert, Qx_Bladder, Qx_Limb_Muscle, Qx_Trachea, QS_Lung, Romanov'. For downsampling experiments, we assume that we can only get partial data containing 80% and 60% cells from whole data. We used the same 10 random seeds as noted earlier to randomly sample cells. For dropout experiments, we assume that all datasets will encounter 'dropout' events with a dropout rate of 15%. We applied above 10 random seeds to randomly select 15% of non-zero expression data to be zero and got 10 different datasets with artificial dropout. After that, we tested eight methods in these noisy downsampling and dropout datasets. The results of two experiments are shown in Figure 6A. The first row of Figure 6A shows ARI values of complete data, 20% missing data and 40% missing data, and the second row shows ARI values of raw data and dropout data with 15% dropout rate. From the results in these five group experiments, our methods, the red and brown lines, have little decrease of ARI values after downsampling or dropout. They still perform better than most of other methods. The results of comparing NMI values are also the same (Supplementary Figure S7). These experimental results confirmed that our methods still have excellent performance, even after missing information of raw datasets or artificial dropout of non-zero expression. Therefore, our methods are stable and robust enough to be used in many datasets and are almost insensitive to dropout or absence, which could change the structure of data.

Scalability of methods

With the development of sequencing techniques, the scale of single-cell data becomes larger and larger. The total cell numbers of scRNA-seq datasets increase from several hundreds to >40 000. Therefore, to satisfy the increasing demand of clustering large-scale scRNA-seq data, scziDesk

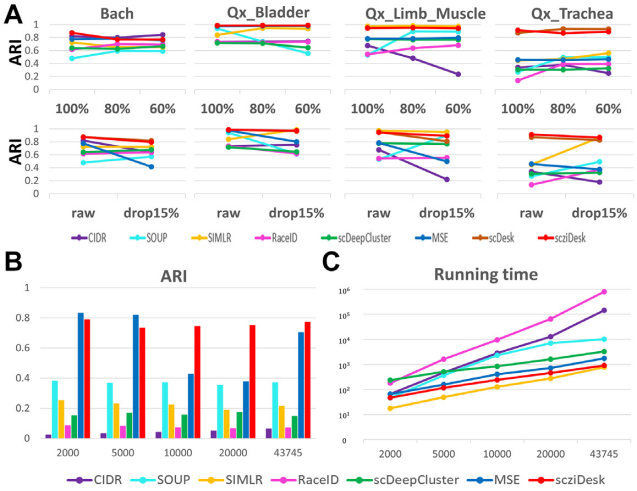

Figure 6. Robustness and scalability experiments in real datasets. (**A**) The upper part shows the change of ARI values from whole data to 80% and 60% downsampling data and the bottom part shows the results of raw data and disturbed data with 15% artificial dropout in four real datasets. (**B**) ARI values in different scales of 'Park' data. (**C**) Average elapsed time (seconds) in different scales of 'Park' data.

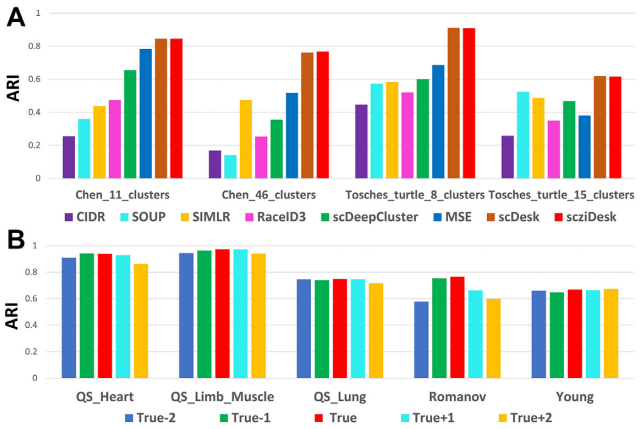

Figure 7. Perturbation experiments for cluster number. (**A**) Comparison of ARI values in two real datasets with coarse and fine divisions. (**B**) Change of ARI values with the disturbed cluster number in five real datasets.

should be able to finish the clustering procedure within a reasonable time frame. Here, we will compare the running time of sczDesk with the other six methods in a mouse kidney dataset called 'Park' (47), which contains 43 745 cells in total. We use the same 10 random seeds to downsample data with 2000, 5000, 10 000 and 20 000 cells. Since the running time is too long for some methods, we only tested their performance of entire data for the first four random seeds when we use CIDR, SOUP and RaceID. Figure 6 and Supplementary Figure S8 compare the ARI values (Figure 6B), NMI values (Supplementary Figure S8A) and the average elapsed time in seconds (Figure 6C) of different scale data for each method. From the histogram of ARI values, we can see that sczDesk, the red one, performs excellently in every data size. Besides, little change of ARI and NMI values can be seen through downsampling, which indicates the robustness of sczDesk. Meanwhile, the running time of sczDesk is less than other six methods, except SIMLR with large-scale version. Actually, SIMLR uses approximation methods for large-scale datasets, which would reduce the accuracy of clustering results. Although it is faster than sczDesk, its ARI value is very low. Contrary to previous methods, such as CIDR and RaceID, the running time of which increases dramatically through the data scale, the running time of sczDesk increases linearly and it can handle datasets with 40 000 cells in 1000 s. The efficiency of our method allows for analyzing scRNA-seq data with even larger scale, e.g. >100 000. Therefore, our method holds the better scalability to integrate running time with clustering accuracy.

SczDesk is insensitive to cluster number

It is worth noting that the authors of two datasets, 'Chen' and 'Tosches_turtle', provided coarse and fine cluster divisions in their original papers. During the previous experiments, we used their fine divisions: 'Chen' has 46 cell types and 'Tosches_turtle' has 15 cell types. We also performed experiments on their coarse divisions ('Chen' has 11 cell types and 'Tosches_turtle' has 8 cell types). The specific ARI and NMI results can be found in Figure 7A and Supplementary Figure S8B. It can be seen that when cluster number is reduced, the performance of almost all methods has been improved, except for SIMLR on 'Chen' dataset and SOUP on 'Tosches_turtle' dataset. SczDesk has improved significantly on both datasets, especially in 'Tosches_turtle' dataset, from the original ARI of 0.62 and NMI of 0.61 to ARI of 0.91 and NMI of 0.85. This explains to some extent that sczDesk performs clustering well regardless of whether the division of the dataset is coarse or fine. In addition, we also tested the performance of sczDesk for other datasets by adding artificial cluster number perturbations. Specifically, assume that the true cluster number is k; we set the experimental cluster number in $\{k-2, k-1, k, k+1, k+2\}$. Figure 7B and Supplementary Figure S9 show that the best clustering performance of sczDesk may not necessarily correspond to the true cluster number on some datasets, but basically all appears in the case where it is increased or decreased by one cluster, and the difference between whole results is generally not large. When the true cluster number is small, such as 'Klein' (four clusters), 'Qx_Trachea' (five clusters) and 'QS_Diaphragm' (five clusters), reducing the cluster number near the true one shows a greater impact than increasing the cluster number, which is reasonable since assuming that the dataset contains only two or three cell populations would severely disrupt its structure. Overall, sczDesk shows favorable stability for varying cluster numbers.

Selecting highly variable genes and self-training strategy improve clustering performance

The ultimate difference between scDeepCluster and sczDesk mainly includes two points: sczDesk selects top (default 500) highly variable genes (the impact of highly variable genes' number will be discussed in the next section), while scDeepCluster chooses whole genes as

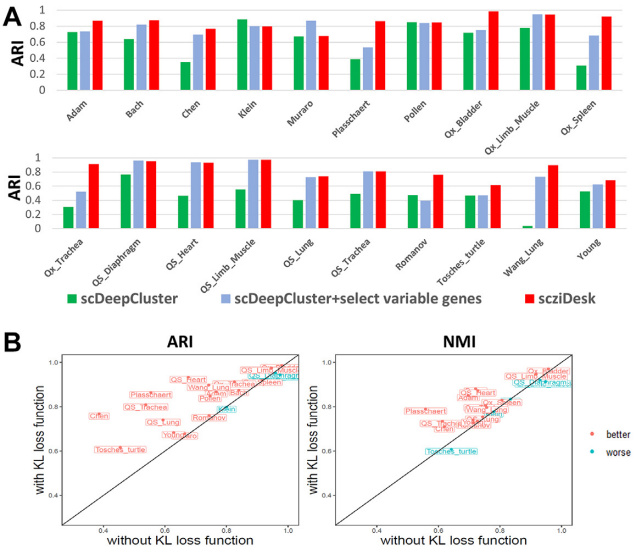

Figure 8. Comparison between scDeepCluster and scziDesk. (**A**) Comparison of ARI values for three methods scDeepCluster, scDeepCluster + select variable genes and scziDesk in 20 real datasets. (**B**) Comparison of ARI and NMI values for scziDesk with and without KL divergence loss in 20 real datasets.

input features, resulting in slow running speed and high memory requirements. ScDeepCluster just considers the relationship between each point and the cluster center in the latent space and neglects the pairwise distance between cells, while scziDesk adds affinity constraint of pairwise similar points on the basis of clustering process. From the comparison of previous experimental results, scziDesk is far superior to scDeepCluster in both simulated and real datasets. Specially, we also used the one-sided pairwise *t*-test to compare their results on 20 real datasets and the *P*-values for ARI and NMI are 6.8×10^{-6} and 3.7×10^{-3}, respectively, which fully validates our claim. In order to further illustrate the impact of these differences, we additionally designed two groups of control experiments on 20 real datasets. First, we changed the network input of scDeepCluster to the same expression profile of 500 highly variable genes as in scziDesk. From the experimental results in Figure 8A and Supplementary Figure S10, the modified scDeepCluster achieves better clustering performance than original scDeepCluster on >16 datasets. We use the one-sided pairwise *t*-test to compare these two sets of results, and the *P*-values are 1.2×10^{-4} (ARI) and 1.1×10^{-2} (NMI), respectively, which demonstrates that the improvement is significant. However, the modified scDeepCluster is still inferior to scziDesk. Specifically, the ARI and NMI values of scziDesk are significantly higher than modified scDeepCluster on >15 datasets, with the *P*-values of 3.6×10^{-3} (ARI) and 1.5×10^{-2} (NMI) given by one-sided pairwise *t*-test. All of the *P*-values comparing the results of scDeepCluster, modified scDeepCluster and scziDesk are shown in Supplementary Figure S11A.

In addition, we explore the effect of self-training strategy on the clustering results, i.e. the importance of the L_3 loss function to the clustering performance in the algorithm. Here, we compare the scziDesk performance with and without L_3 in loss function using the same 20 real datasets. From the results in Figure 8B, we see that the overall clustering performance decreases when we throw away L_3 loss. Specifically, 17 datasets give the significantly lower ARI value than before once we delete L_3 loss. We also conducted one-sided pairwise *t*-test to compare the results with and without L_3 loss function. The *P*-values of ARI and NMI are 4×10^{-4} and 2×10^{-3}, respectively, which shows that there is a significant improvement in clustering results after considering self-training procedure as part of training. So far, we have proved that similar points' affinity constraint has a significant effect on clustering performance, which is in line with our expectation since the self-training procedure can bring the cells with similar expression patterns closer together and avoid the influence of outliers on the results.

DISCUSSION AND CONCLUSION

ScRNA-seq plays a vital role in the understanding of cellular heterogeneity, which is concealed by traditional bulk RNA sequencing. To this end, developing a method to distinguish different cell types from sequencing data is of great importance. In this paper, we proposed a model-based clustering method scziDesk for scRNA-seq data, which combines data likelihood modeling with a self-training soft *K*-means algorithm. Our method has four main advantages: first, we applied a deep autoencoder technique with probabilistic statistical modeling to estimate the expression of single cells, denoising the data while performing non-linear dimension reduction on the data simultaneously. Second, we select top (default 500) highly variable genes and only use the expression of these genes to perform cell clustering. This step removes those redundant low-expression genes that may contaminate cluster results and can increase clustering speed dramatically. Third, we use the soft *K*-means clustering algorithm with probability allocation instead of the hard *K*-means clustering method, which can redress the bias caused by incorrect allocation to a certain extent. Fourth, on the basis of soft *K*-means clustering loss, the customized self-training strategy is added as a part of training to strengthen the relationship between adjacent cells dynamically. We conducted more experiments to show the merit of our method in other aspects.

Compare with and without stack autoencoder architecture

In our neural network setting, we used the traditional autoencoder network that has two hidden layers for encoder and decoder parts. The dimensions of the input layer and two hidden layers are 500, 256 and 64, respectively. Another popular autoencoder structure called stack autoencoder is often used. As the term suggests, the stack autoencoder implements twice or more data compression and data reconstruction. In order to explore whether this network structure can improve clustering performance significantly, we compared scziDesk with and without stack autoencoder architecture in 20 real datasets. The dimensions of the input layer and three hidden layers for stack encoder are 500, 256, 64 and 500, respectively. From Figure 9A, we can see that the result without stack structure is slightly better than that with stack structure in 20 real datasets. We used one-sided

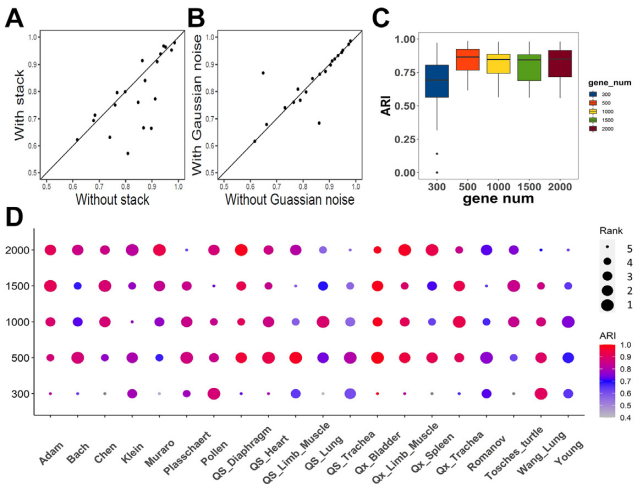

Figure 9. Network sensitivity analysis. (**A**) Comparison of ARI values with and without stack autoencoder. (**B**) Comparison of ARI values with and without Gaussian noise. (**C**) Box plot comparing ARI values of scziDesk using different numbers of highly variable genes. (**D**) Dot plot of comparing ARI values of selecting different numbers of highly variable genes. The size of the dot stands for the rank among five results and the color stands for the ARI values.

pairwise *t*-test for them and got a *P*-value of 0.017, which illustrates that utilizing stack structure is redundant. Overall, our neural network structure selection is appropriate.

Compare with and without Gaussian noise

Overfitting is an important topic in machine learning and deep learning. In order to prevent this phenomenon, we add Gaussian noise in each input for hidden layers in our encoder network. To test the influence of Gaussian noise for clustering results, we also implement control experiments with and without Gaussian noise. Figure 9B shows that the performance with Gaussian noise is better in nearly half of datasets, but worse in the other half. The *P*-value of two-sided pairwise *t*-test for them is 0.84, which demonstrates that the difference between them is not significant. Thus, Gaussian noise has virtually nothing to do with clustering performance, and our neural network does not appear to suffer from the overfitting phenomenon.

Compare selecting different numbers of variable genes

In our method, we selected the top 500 highly variable genes by default to conduct clustering analysis. Actually, highly variable genes can capture more biological information than lowly variable genes that have little influence over determining cell types. Moreover, by selecting highly variable genes, we can reduce the model and time complexity of our clustering algorithms. To test the influence of highly variable genes' number for results, we vary them from 300 to 2000 and apply scziDesk on 20 real datasets. We used box plot (Figure 9C) and dot plot (Figure 9D and Supplementary Figure S12) to show the ARI (NMI) values of 20 real datasets by selecting 300, 500, 1000, 1500 and 2000 highly variable genes, respectively. Overall, the performance of the highly variable genes' number 300 was a little worse

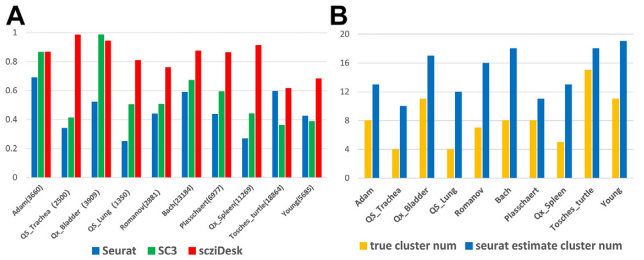

Figure 10. Additional real dataset analysis. (**A**) Comparison of ARI values between Seurat, SC3 and scziDesk in 10 real datasets. (**B**) Comparison of optimal cluster number estimated by Seurat with the true cluster number in 10 real datasets.

than the other four cases, and the mean and variance of the other four groups of results did not seem to differ much. Furthermore, we first apply ANOVA variance test to find out that the number of selected genes is a significant factor for clustering performance (*P*-value for ARI is 7.68×10^{-4} and 2.21×10^{-4} for NMI). Then, we conduct pairwise *t*-test between every two groups from five groups and find that only the performance of 300 highly variable genes is significantly lower than others, while the differences between other groups are not significant. The *P*-value of these tests can be found in Supplementary Figure S11C. Specifically, we find that for those datasets with massive cell types, such as 'Chen', 'Tosches_turtle' and 'Young', selecting more highly variable genes does help improve clustering performance. Therefore, considering the relatively small change of clustering performance in general, we recommend using the top 500 highly variable genes for clustering as a priority.

Additional comparison with Seurat and SC3 in real datasets

As we mentioned in the 'Results' section, we did not compare with Seurat and SC3 in whole simulation and real datasets for fairness. We indeed compare our method with Seurat and SC3 on 10 real datasets. Since clustering performance of Seurat is highly dependent on model parameters, especially 'resolution', here we evaluate its performance under default parameters of pre-process and PCA selection procedure, and then take the resolution parameter in the clustering step to range from 0.5 to 1.5 by 0.1, which is recommended by the author of Seurat. We consider the maximum ARI values under all of these resolutions as its results and simultaneously record the corresponding cluster number estimated by itself. For SC3, we used the approximation version of SC3 when the cell number is >5000 since it is really time consuming on large-scale datasets. Figure 10A shows the ARI values of Seurat, SC3 and scziDesk in five small datasets and five large datasets. Despite selecting the optimal resolution parameter from the candidate range, Seurat still achieves unsatisfactory performance in ARI values, which is mostly due to its linear dimension reduction and overestimation of the cluster numbers (shown in Figure 10B). As for SC3, although in small datasets it may perform well, its results decrease dramatically and the ARI values are much lower than scziDesk when the number of cells exceeds 5000. Overall, scziDesk is not inferior to these two

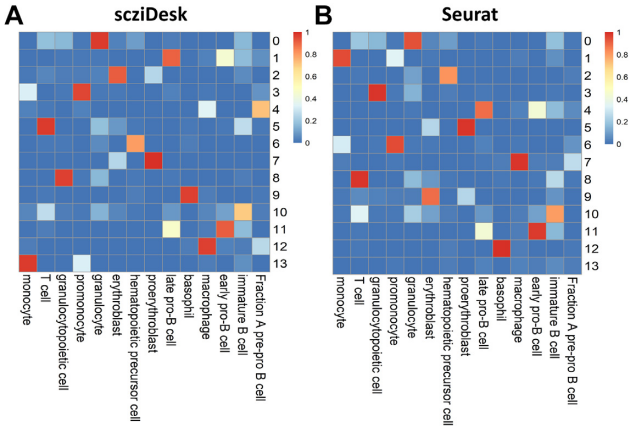

Figure 11. Comparing the similarity of differentially expressed genes. (**A**) Comparison of similarity of DEGs in 14 scziDesk clusters with golden standard 14 cell types. (**B**) Comparison of similarity of DEGs in 14 Seurat clusters with golden standard 14 cell types.

widely used methods and can be applied in both small and large datasets well.

Table 1. Optimal parameter settings of scziDesk for real datasets

	scziDesk			
	γ	λ	ARI (default)	ARI
Adam	0.001	0.001	0.8680	0.8953
Bach	0.001	0.001	0.8738	0.9270
Chen	0.001	0.001	0.7677	0.8637
Klein	0.001	0.01	0.7984	0.9356
Muraro	0.1	0.1	0.6784	0.8839
Plasschaert	0.1	0.1	0.8634	0.9216
Pollen	0.1	0.01	0.8476	0.9225
Qx_Bladder	0.001	0.001	0.9858	0.9900
Qx_Limb_Muscle	0.1	0.001	0.9441	0.9754
Qx_Spleen	0.01	0.01	0.9197	0.9369
Qx_Trachea	0.1	0.1	0.9123	0.9555
QS_Diaphragm	0.1	0.01	0.9517	0.9715
QS_Heart	0.1	0.001	0.9324	0.9578
QS_Limb_Muscle	0.1	0.01	0.9743	0.9837
QS_Lung	0.01	0.01	0.7401	0.8659
QS_Trachea	0.001	0.001	0.8085	0.8565
Romanov	0.001	0.001	0.7603	0.7831
Tosches_turtle	0.1	0.001	0.6165	0.7210
Wang_Lung	0.001	0.1	0.8975	0.9717
Young	0.01	0.01	0.6836	0.7939

The default value of ARI refers to the median value of 10 repeated tests under default parameter setting. Qx refers to Quake_10x and QS refers to Quake_Smart-seq2.

Compare selected differential expression genes' list

Identifying cell clusters lay the basic foundation of downstream analysis of single-cell data analysis, such as finding differential expression genes (DEGs) and cell function annotation. Here, we selected a mouse bone marrow dataset (45), 'Quake_10x_bone_marrow', to find out whether our method is beneficial for downstream analysis. The original authors first collected 14 cell types from cell ontology (48), including a small fraction of the progenitor of hematopoietic stem cells (HSCs) and most of their progeny cells in three HSC differential branches. Then, they applied the standard annotation method for the sequencing data, i.e. using cluster-specific gene expression of known markers and genes to assign cell type annotations to each cluster. In this paper, we first used our method scziDesk to get 14 cell clusters and showed the comparison of UMAP visualization plot with the given true label in Supplementary Figure S13.

In addition, we applied function 'FindAllMarkers' with default parameters in Seurat (22) package to find out the DEGs of each cluster. Moreover, we used the completely same way to obtain the DEGs under the golden standard clusters with cell type annotation, and also under the Seurat standard analysis pipeline with resolution parameter equal to 0.3 to get 14 clusters. For each method, we select top 100 DEGs of each cluster and compare their overlap to find out whether we can annotate every cluster correctly to a known cell type in golden standard. The total similarity heat maps of marker genes' list are shown in Figure 11, where similarity = number of overlapped DEGs/number of total selected DEGs. The rows stand for 14 clusters determined by clustering methods and columns stand for known cell types that are regarded as golden standard. From Figure 11A, we can see that for every cell type we have only one cluster that holds the highest similarity in differentially expressed genes' list. Actually, most similarities are >0.9, such as the cluster 0 with monocyte, and only three clusters give similarities of ∼0.7. We can annotate each cluster determined by our method to a specific cell type by finding out the most similar cell type. However, for Seurat clustering in Figure 11B, although most clusters can match to unique cell type, there is no cluster match with the 14th cell type, Fraction A pre-pro B cell. Therefore, by comparing the differentially expressed genes, our clusters have high similarities with the golden standard cell types, which makes the cell type annotation easy and correct. Additional feature plots of some important identifier marker genes for cell types can be seen in Supplementary Figure S14. For example, *Csf1r* is an important marker gene to distinguish monocyte or monocyte progenitor from other types of cells (49). From the feature plot, cells that highly express *Csf1r* get together in cluster 13 of our clustering results and this cluster can be annotated as monocyte; that is, our clustering results match with background biology knowledge. In summary, applying our clustering method does contribute to the scRNA-seq data downstream analysis.

When utilizing deep learning technology to analyze scRNA-seq data, compared to traditional MSE-based autoencoder, modeling with a reasonable parameter distribution like ZINB can often learn more cluster-friendly latent space and achieve better clustering performance. A large part of the reason is that the MSE-based autoencoder fails to learn the gene expression dispersion and cannot interpret the sparsity of data. Although traditional statistical methods can also estimate the parameters of these distributions, they usually add some additional assumptions, such as conjugate Bayesian priors or specific forms of generalized linear models, which cannot fully capture and portray the complex dependencies among parameters. The neural network can effectively solve the model on a wider domain and capture the potential low-dimensional manifold of the data with

complex dependencies at the same time. This method of estimating parameters is fascinating and inspiring. In addition, although assuming that latent space follows the Gaussian mixture model (31) has perfect statistical interpretation significance, in practice, clustering algorithms based on distance measures like soft *K*-means are often more effective and robust since the component centers of the Gaussian mixture model are not always consistent with the embedding of latent representation (50). Besides, complex low-dimensional manifolds cannot be characterized by mixture Gaussian distributions at all times. In manifold learning, retaining affinity relationships between similar samples is critical to preserve global and local structure of the data. Since the raw data of single-cell transcriptome are highly noisy, the pairwise distance relationship in the high-dimensional space is not so reliable; thus, we consider characterizing the constraints between the pairwise similarities in the low-dimensional latent space. The experimental results fully illustrate that our self-learning strategy effectively captures the data structure and assists the clustering process. Overall, perfect data representation learning with an enhanced soft clustering algorithm achieves excellent performance.

In recent years, multi-task learning has attracted increasing attention in the field of deep learning. How to balance the training process of multiple objectives is also an essential and worth exploring issue. In this paper, we need to judge and weigh the importance of multiple objective functions for data fitting and model optimization. Without any preference, we hope that the contribution of the three loss functions to the gradients is at the same level; that is, by adjusting the weight parameters γ and λ, their values are comparable and are on the same order of magnitude. In the specific algorithm implementation, L_1 is averaged over all cells and genes, while L_2 and L_3 are averaged over all cells, so L_2 and L_3 are naturally larger than L_1. Without weighted recoordination, simply superimposing the three loss functions together will most likely destroy the global structure of gene expression data. Therefore, in the 'Analysis of real data' section, we set the hyperparameters γ and λ in the scziDesk model to 0.001 for all datasets by default. Actually, when we take the grid search method to find the hyperparameter with the best clustering performance for each of the 20 real datasets, we can discover that the optimal hyperparameters for some datasets may not be the default value. Specifically, we assume that γ and λ are both selected from (0.1, 0.01, 0.001); the parameter combinations that achieve the highest ARI and NMI values are shown in Table 1 and Supplementary Table S2. It can be seen that taking the results of default parameters as a reference, on average, the ARI and NMI values of 20 real datasets under the optimal parameters have increased by nearly 0.07, and individual datasets have even improved by up to 0.2. The *P*-values of one-sided pairwise *t*-test are 2.2×10^{-4} (ARI) and 9.0×10^{-7} (NMI) (see Supplementary Figure S11B), respectively, which fully illustrates that optimal hyperparameter search can improve the results significantly. From the perspective of the optimal parameter matching pattern, we recommend that users can set the value of the parameter λ not greater than γ as much as possible.

Determining the number of clusters is an open problem. Although many statistical methods have been proposed to solve this problem, such as gap statistic (51), silhouette score and so on, the results are not always satisfactory. For some datasets, the estimated cluster number by gap statistics can be the same as the true cluster number (as shown in Supplementary Figure S15), while it is overestimated most of the time. In the 'Results' section, we have validated that our method is very robust and stable against perturbations of cluster numbers. Moreover, in our opinion, the true cluster numbers should be given by biologists. People may like to find more clusters if their interest is to study the fine-grained heterogeneity, or may like to reduce the number of clusters if they do not want to care too much about details. Therefore, it is hard to determine the exact cluster number *K* without background knowledge. Maybe it is more reasonable to combine it with some other downstream analysis during scRNA-seq data analysis, such as marker gene identification and cellular ontology annotation.

A controversial topic is whether scRNA-seq data are zero-inflated. Recently, Svensson (52) pointed out that scRNA-seq data based on droplet experiments are not zero-inflated, which means that NB distribution without zero inflation is sufficient to characterize these data. Twelve of the real datasets we used are based on droplet experiments, including inDrop, Drop-seq, 10x, etc. From their experimental results, scDesk and scziDesk modeling each win half and the *P*-values of the two-sided pairwise *t*-test for ARI and NMI values are 0.2340 and 0.4498 (see Supplementary Figure S11B), respectively, which demonstrates that the difference between them is not significant. Thus, for droplet-based datasets, we suggest that scDesk can also be considered. In addition, our model does not consider non-biological variation such as batch effect biases. For datasets with strong batch effects, we recommend that users can use some existing batch effect correction methods like pagoda2 (53) to remove these non-biological variations before using our clustering method.

All in all, we validated our method in simulation and real datasets by comparing the median ARI and NMI values under different seeds with five existing methods. Our simulation experiments used a recently developed R package 'splatter' to generate data similar to reality. We considered as many situations as we may encounter in real datasets, including 'dropout' events, different cluster number, variable cluster size and so on. Our method, scziDesk, performs better than other methods in almost all situations. For real dataset analysis, we tested 20 datasets from different species, organs and sequence platforms. Our model ranks in the top two in most datasets among the eight methods and never falls into the last three. Other existing methods may fail to discover genuine clusters in some datasets and sometimes have very poor results with ARI or NMI < 0.2. Contrary to most deep learning-based methods with high dependence on parameters, our method demonstrated its robustness and stability in our extra experiments, including downsampling and dropout. We also verified that our method is not too sensitive to the neural network structure. In the scalability experiments, we showed that our method can finish clustering in the shortest time with the best performance compared to other methods. When varying the cluster number within reasonable limits, our method would not cause clustering performance to collapse. Most importantly,

our method helps improve differential expression analysis and facilitates cell annotation. All in all, our method is superior to most existing methods, considering accuracy, robustness and efficiency, thus satisfying the growing demand of scRNA-seq data analysis.

Nowadays, fast-developing single-cell sequencing can profile genetic, epigenetic, spatial, proteomic and lineage information in individual cells. It is both an opportunity and a challenge for integrative single-cell analysis that learns across multiple types of data. In the future, we are interested in extending our clustering method to multi-omics research.

DATA AVAILABILITY

All real datasets and the source codes for this study are available at GitHub: https://github.com/xuebaliang/scziDesk.

SUPPLEMENTARY DATA

Supplementary Data are available at NARGAB Online.

ACKNOWLEDGEMENTS

We thank Frank Huang from Cincinnati Children's Hospital Medical Center for providing computer and server for part of computational experiments.
Authors' contribution: L.C. and M.D. conceived and designed the scziDesk and scDesk models. W.W. implemented the simulation study and real dataset analysis. Y.Z. wrote the experiment codes. Lastly, L.C., W.W. and M.D. wrote the whole manuscript. All authors read and approved the final manuscript.

FUNDING

National Key Research and Development Program of China [2016YFA0502303]; National Key Basic Research Project of China [2015CB910303]; National Natural Science Foundation of China [31871342].
Conflict of interest statement. None declared.

REFERENCES

1. Shekhar,K., Lapan,S.W., Whitney,I.E., Tran,N.M., Macosko,E.Z., Kowalczyk,M., Adiconis,X., Levin,J.Z., Nemesh,J., Goldman,M. *et al.* (2016) Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics. *Cell*, **166**, 1308–1323.
2. Ben-Dor,A., Shamir,R. and Yakhini,Z. (1999) Clustering gene expression patterns. *J. Comput. Biol.*, **6**, 281–297.
3. Shapiro,E., Biezuner,T. and Linnarsson,S. (2013) Single-cell sequencing-based technologies will revolutionize whole-organism science. *Nat. Rev. Genet.*, **14**, 618–630.
4. Kolodziejczyk,A.A., Kim,J.K., Svensson,V., Marioni,J.C. and Teichmann,S.A. (2015) The technology and biology of single-cell RNA sequencing. *Mol. Cell*, **58**, 610–620.
5. Wang,Y. and Navin,N.E. (2015) Advances and applications of single-cell sequencing technologies. *Mol. Cell*, **58**, 598–609.
6. Hebenstreit,D. (2012) Methods, challenges and potentials of single cell RNA-seq. *Biology*, **1**, 658–667.
7. Grün,D., Kester,L. and Van Oudenaarden,A. (2014) Validation of noise models for single-cell transcriptomics. *Nat. Methods*, **11**, 637–640.
8. Svensson,V., Natarajan,K.N., Ly,L.-H., Miragaia,R.J., Labalette,C., Macaulay,I.C., Cvejic,A. and Teichmann,S.A. (2017) Power analysis of single-cell RNA-sequencing experiments. *Nat. Methods*, **14**, 381–387.
9. Zheng,G.X., Terry,J.M., Belgrader,P., Ryvkin,P., Bent,Z.W., Wilson,R., Ziraldo,S.B., Wheeler,T.D., McDermott,G.P., Zhu,J. *et al.* (2017) Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.*, **8**, 14049.
10. Macosko,E.Z., Basu,A., Satija,R., Nemesh,J., Shekhar,K., Goldman,M., Tirosh,I., Bialas,A.R., Kamitaki,N., Martersteck,E.M. *et al.* (2015) Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, **161**, 1202–1214.
11. Kharchenko,P.V., Silberstein,L. and Scadden,D.T. (2014) Bayesian approach to single-cell differential expression analysis. *Nat. Methods*, **11**, 740–742.
12. Finak,G., McDavid,A., Yajima,M., Deng,J., Gersuk,V., Shalek,A.K., Slichter,C.K., Miller,H.W., McElrath,M.J., Prlic,M. *et al.* (2015) MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. *Genome Biol.*, **16**, 278.
13. Ji,Z. and Ji,H. (2016) TSCAN: pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic Acids Res.*, **44**, e117.
14. Qiu,X., Mao,Q., Tang,Y., Wang,L., Chawla,R., Pliner,H.A. and Trapnell,C. (2017) Reversed graph embedding resolves complex single-cell trajectories. *Nat. Methods*, **14**, 979–982.
15. Kumar,M.P., Du,J., Lagoudas,G., Jiao,Y., Sawyer,A., Drummond,D.C., Lauffenburger,D.A. and Raue,A. (2018) Analysis of single-cell RNA-seq identifies cell–cell communication associated with tumor characteristics. *Cell Rep.*, **25**, 1458–1468.
16. Vento-Tormo,R., Efremova,M., Botting,R.A., Turco,M.Y., Vento-Tormo,M., Meyer,K.B., Park,J.-E., Stephenson,E., Polański,K., Goncalves,A. *et al.* (2018) Single-cell reconstruction of the early maternal–fetal interface in humans. *Nature*, **563**, 347–353.
17. Aibar,S., González-Blas,C.B., Moerman,T., Imrichova,H., Hulselmans,G., Rambow,F., Marine,J.-C., Geurts,P., Aerts,J., van den Oord,J. *et al.* (2017) SCENIC: single-cell regulatory network inference and clustering. *Nat. Methods*, **14**, 1083–1086.
18. Dai,H., Li,L., Zeng,T. and Chen,L. (2019) Cell-specific network constructed by single-cell RNA sequencing data. *Nucleic Acids Res.*, **47**, e62.
19. Lin,P., Troup,M. and Ho,J.W. (2017) CIDR: ultrafast and accurate clustering through imputation for single-cell RNA-seq data. *Genome Biol.*, **18**, 59.
20. Wang,B., Zhu,J., Pierson,E., Ramazzotti,D. and Batzoglou,S. (2017) Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning. *Nat. Methods*, **14**, 414–416.
21. Levine,J.H., Simonds,E.F., Bendall,S.C., Davis,K.L., El-ad,D.A., Tadmor,M.D., Litvin,O., Fienberg,H.G., Jager,A., Zunder,E.R. *et al.* (2015) Data-driven phenotypic dissection of AML reveals progenitor-like cells that correlate with prognosis. *Cell*, **162**, 184–197.
22. Satija,R., Farrell,J.A., Gennert,D., Schier,A.F. and Regev,A. (2015) Spatial reconstruction of single-cell gene expression data. *Nat. Biotechnol.*, **33**, 495–502.
23. Grün,D., Muraro,M.J., Boisset,J.-C., Wiebrands,K., Lyubimova,A., Dharmadhikari,G., van den Born,M., van Es,J., Jansen,E., Clevers,H. *et al.* (2016) *De novo* prediction of stem cell identity using single-cell transcriptome data. *Cell Stem Cell*, **19**, 266–277.
24. Kiselev,V.Y., Kirschner,K., Schaub,M.T., Andrews,T., Yiu,A., Chandra,T., Natarajan,K.N., Reik,W., Barahona,M., Green,A.R. *et al.* (2017) SC3: consensus clustering of single-cell RNA-seq data. *Nat. Methods*, **14**, 483–486.
25. Yang,Y., Huh,R., Culpepper,H.W., Lin,Y., Love,M.I. and Li,Y. (2018) SAFE-clustering: single-cell aggregated (from ensemble) clustering for single-cell RNA-seq data. *Bioinformatics*, **35**, 1269–1277.
26. LeCun,Y., Bengio,Y. and Hinton,G. (2015) Deep learning. *Nature*, **521**, 436–444.
27. Eraslan,G., Avsec,Ž., Gagneur,J. and Theis,F.J. (2019) Deep learning: new computational modelling techniques for genomics. *Nat. Rev. Genet.*, **20**, 389–403.
28. Hinton,G.E. and Salakhutdinov,R.R. (2006) Reducing the dimensionality of data with neural networks. *Science*, **313**, 504–507.

29. Ding,J., Condon,A. and Shah,S.P. (2018) Interpretable dimensionality reduction of single cell transcriptome data with deep generative models. *Nat. Commun.*, **9**, 2002.
30. Eraslan,G., Simon,L.M., Mircea,M., Mueller,N.S. and Theis,F.J. (2019) Single-cell RNA-seq denoising using a deep count autoencoder. *Nat. Commun.*, **10**, 390.
31. Grønbech,C.H., Vording,M.F., Timshel,P.N., Sønderby,C.K., Pers,T.H. and Winther,O. (2020) scVAE: variational auto-encoders for single-cell gene expression data. *Bioinformatics*, btaa293.
32. Tian,T., Wan,J., Song,Q. and Wei,Z. (2019) Clustering single-cell RNA-seq data with a model-based deep learning approach. *Nat. Mach. Intell.*, **1**, 191–198.
33. Xie,J., Girshick,R. and Farhadi,A. (2016) Unsupervised deep embedding for clustering analysis. In: *International Conference on Machine Learning*, pp. 478–487.
34. Guo,X., Gao,L., Liu,X. and Yin,J. (2017) Improved deep embedded clustering with local structure preservation. In: I*nternational Joint Conference on Artificial Intelligence*, pp. 1753–1759.
35. Li,X., Lyu,Y., Park,J., Zhang,J., Stambolian,D., Susztak,K., Hu,G. and Li,M. (2020) Deep learning enables accurate clustering and batch effect removal in single-cell RNA-seq analysis. *Nat. Commun.*, **11**, 1–14.
36. Amodio,M., van,D.D., Srinivasan,K., Chen,W.S., Mohsen,H., Moon,K.R., Campbell,A., Zhao,Y., Wang,X., Venkataswamy,M. *et al.* (2019) Exploring single-cell data with deep multitasking neural networks. *Nat. Methods*, **16**, 1139–1145.
37. Wolf,F.A., Angerer,P. and Theis,F.J. (2018) SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.*, **19**, 15.
38. Vlasblom,J. and Wodak,S.J. (2009) Markov clustering versus affinity propagation for the partitioning of protein interaction graphs. *BMC Bioinformatics*, **10**, 99.
39. van der Maaten,L. and Hinton,G. (2008) Visualizing data using *t*-SNE. *J. Mach. Learn. Res.*, **9**, 2579–2605.
40. Santos,J.M. and Embrechts,M. (2009) On the use of the adjusted Rand index as a metric for evaluating supervised classification. In: *International Conference of Artificial Neural Networks*. Springer, Berlin, pp. 175–184.
41. Vinh,N.X., Epps,J. and Bailey,J. (2010) Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.*, **11**, 2837–2854.
42. Herman,J.S. and Grün,D. (2018) FateID infers cell fate bias in multipotent progenitors from single-cell RNA-seq data. *Nat. Methods*, **15**, 379–386.
43. Zhu,L., Lei,J., Devlin,B. and Roeder,K. (2019) Semi-soft clustering of single cell data. *Proc. Natl Acad. Sci. U.S.A.*, **116**, 466–471.
44. Zappia,L., Phipson,B. and Oshlack,A. (2017) Splatter: simulation of single-cell RNA sequencing data. *Genome Biol.*, **18**, 174.
45. Schaum,N., Karkanias,J., Neff,N.F., May,A.P., Quake,S.R., Wyss-Coray,T., Darmanis,S., Batson,J., Botvinnik,O., Chen,M.B. *et al.* (2018) Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris: the Tabula Muris Consortium. *Nature*, **562**, 367–372.
46. McInnes,L., Healy,J. and Melville,J. (2018) UMAP: uniform manifold approximation and projection for dimension reduction. arXiv doi: https://arxiv.org/abs/1802.03426, 09 February 2018, preprint: not peer reviewed.
47. Park,J., Shrestha,R., Qiu,C., Kondo,A., Huang,S., Werth,M., Li,M., Barasch,J. and Suszták,K. (2018) Single-cell transcriptomics of the mouse kidney reveals potential cellular targets of kidney disease. *Science*, **360**, 758–763.
48. Bakken,T., Cowell,L., Aevermann,B.D., Novotny,M., Hodge,R., Miller,J.A., Lee,A., Chang,I., McCorrison,J., Pulendran,B. *et al.* (2017) Cell type discovery and representation in the era of high-content single cell phenotyping. *BMC Bioinformatics*, **18**, 559.
49. Baccin,C., Al-Sabah,J., Velten,L., Helbling,P.M., Grünschläger,F., Hernández-Malmierca,P., Nombela-Arrieta,C., Steinmetz,L.M., Trumpp,A. and Haas,S. (2020) Combined single-cell and spatial transcriptomics reveal the molecular, cellular and spatial bone marrow niche organization. *Nat. Cell Biol.*, **22**, 38–48.
50. Xiong,L., Xu,K., Tian,K., Shao,Y., Tang,L., Gao,G., Zhang,M., Jiang,T. and Zhang,Q.-F. (2019) SCALE method for single-cell ATAC-seq analysis via latent feature extraction. *Nat. Commun.*, **10**, 4576.
51. Tibshirani,R., Walther,G. and Hastie,T. (2001) Estimating the number of clusters in a data set via the gap statistic. *J. R. Stat. Soc. B: Stat. Methodol.*, **63**, 411–423.
52. Svensson,V. (2020) Droplet scRNA-seq is not zero-inflated. *Nat. Biotechnol.*, **38**, 147–150.
53. Jean,F., Neeraj,S., Rui,L., Gwendolyn,E.K., Yun,C.Y., Joseph,L.H., Fiona,K., Jian-Bing,F., Kun,Z., Jerold,C. *et al.* (2016) Characterizing transcriptional heterogeneity through pathway and gene set overdispersion analysis. *Nat. Methods*, **13**, 241–244.