Briefings in Bioinformatics, 23(1), 2022, 1–19

https://doi.org/10.1093/bib/bbab345
Review

A comparison of deep learning-based pre-processing and clustering approaches for single-cell RNA sequencing data

Jiacheng Wang, Quan Zou and Chen Lin

Corresponding authors: Quan Zou, Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China, and Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou, Zhejiang, China. Tel.: 170-9226-1008. E-mail: zouquan@nclab.net; Chen Lin, School of Informatics, Xiamen University, Xiamen, China. Tel.: 13606938620. E-mail: chenlin@xmu.edu.cn

Abstract

The emergence of single cell RNA sequencing has facilitated the studied of genomes, transcriptomes and proteomes. As available single-cell RNA-seq datasets are released continuously, one of the major challenges facing traditional RNA analysis tools is the high-dimensional, high-sparsity, high-noise and large-scale characteristics of single-cell RNA-seq data. Deep learning technologies match the characteristics of single-cell RNA-seq data perfectly and offer unprecedented promise. Here, we give a systematic review for most popular single-cell RNA-seq analysis methods and tools based on deep learning models, involving the procedures of data preprocessing (quality control, normalization, data correction, dimensionality reduction and data visualization) and clustering task for downstream analysis. We further evaluate the deep model-based analysis methods of data correction and clustering quantitatively on 11 gold standard datasets. Moreover, we discuss the data preferences of these methods and their limitations, and give some suggestions and guidance for users to select appropriate methods and tools.

Key words: single-cell RNA-seq; deep learning; pre-processing steps; clustering analysis

Introduction

The significant advances of single cell RNA sequencing (scRNA-seq) technology have made it possible for researchers to explore the genome and transcriptome at the level of individual cells. It is considered an effective way to elucidate molecular heterogeneity, pedigree path analysis, cell lineage relationships, random gene expression and rare cell-type identifying, and the gold standard for defining cell states and phenotypes as of 2021 [1–4]. In recent years, scRNA-seq technology has been widely used in the development of biology, neuroscience, oncology, microbiology and other important research fields. Since the first scRNA-seq experiment was reported [5], a large volume of scRNA-seq data from multiple platforms have been released [6]. With large volumes of scRNA-seq data available, how to effectively analyze and mine complex relationships and potential regulation patterns between cells has become the key to the success of downstream research.

The computational analysis of scRNA-seq data consists of preprocessing of raw data, visualization and downstream analysis, and depends on bioinformatics tools. The preprocessing is divided into processes such as quality control, normalization, data correction and integration, feature selection and

Jiacheng Wang is a doctoral student at the Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China. His research interests include machine learning and bioinformatics. His email is jiachengwang@std.uestc.edu.cn.
Quan Zou is a professor at the University of Electronic Science and Technology of China. He is a senior member of Institute of Electrical and Electronics Engineers (IEEE) and Association for Computing Machinery (ACM). He won the Clarivate Analytics Highly Cited Researchers in 2018–2020. He majors in bioinformatics, machine learning and algorithms. His email is zouquan@nclab.net.
Chen Lin is a professor at Xiamen University. She majors in social network and bioinformatics. Her email is chenlin@xmu.edu.cn.
Submitted: 26 June 2021; **Received (in revised form):** 22 July 2021

dimensionality reduction, and visualization. Downstream analysis includes tasks such as clustering, clustering annotations, cell composition analysis, trajectory variables and differential gene expression [7, 8].

ScRNA-seq data have the characteristics of scale, noisy [9], high-dimensionality and heterogeneity [10], etc. These characteristics present more challenging to perform the analysis of scRNA-seq data compared to RNA sequencing data from bulk cell populations [11]. Several traditional computational methods from bulk RNA-seq are applied to scRNA-seq analysis [12, 13]. However, the specificity of scRNA-seq led to the poor performances [14]. To adapt the characteristics of scRNA-seq data, computational analysis requires updating traditional methods or developing new methods. In the past ten years, deep learning technology, a powerful component of machine learning, has received great attention and made significant progress [15]. Deep learning approaches employ multiple neural layers to approximate the nonlinear transformation that can learn the potential information and patterns in rather complex data sets. Unlike traditional machine learning methods, deep learning is independent of prior knowledge and can automatically learn features from input data. Its end-to-end learning scheme and outstanding performance of representation learning allow researchers to tap its great potential in computer vision [16], natural language processing [17], and DNA sequence alignment [18], etc. Deep learning technology is not sensitive to the type of noise and demonstrates promising power to denoise. Besides, deep learning methods have strong scalability and ability to process complex and multi-modality data. These advantages make it very compatible with single-cell RNA sequencing data [7]. Recently, deep learning algorithms have been widely applied to a variety of scRNA-seq analysis tasks, including data correction, dimensionality reduction and visualization, and clustering. In addition to methods that focus on a particular analysis task, some pipelines form a complete analysis process to fit data, preprocess the data, and perform downstream analysis within the model framework. It can retain accurate estimates of data changes, while incorporating the influence of noise and substitution effects into downstream statistical analysis models.

Although considerable deep learning-based scRNA-seq analysis algorithms and pipelines have been proposed, a barrier to researchers adopting these tools is the lack of a comprehensive guidance for description and comparison of them. In this paper, we present a review for scRNA-seq computational analysis tools based on deep learning algorithms. We descripted and detailed these popular available tools according to the various steps of scRNA-seq analysis. For several crucial and most studied steps such as data correction and clustering analysis, we performed these deep model-based scRNA-seq analysis algorithms on eleven publicly available datasets of different species, sizes and complexity to evaluate their performance. In addition, we discussed their preferences and limitations from multiple perspectives according to the assessment results and offer some advice on how to select appropriate tools. Finally, we concluded the review with discussions of two considerable challenges and future expectation in single-cell RNA-seq analysis based on deep learning as technology continues to evolve.

Deep learning-based methods for scRNA-seq analysis

In this paper, we reviewed 22 scRNA-seq analysis methods based on deep learning models. These methods cover the preprocessing procedure: quality control (1), normalization (1), batch correction (7), denoising and imputation (9), dimension reduction and visualization (5), and downstream clustering analysis (8). To better overview these deep model-based analysis methods, a workflow of scRNA-seq analysis covering them is provided (Figure 1). Among them, there are several multifunctional tools, such as scVI [19], SAUCIE [20], scScope [21]. A summary of these methods and tools is also given in Table 1, including information of their language, functions, applied underlying deep models, suitable datasets and download URLs. Deeper-level approaches of downstream analysis are described and summarized by Zhang et al. [22].

Quality control

The initial step in scRNA-seq is typically to isolate samples of biological tissue to profile the mRNAs in individual cells. There are two techniques for single cell dissociation, one based on a flat plate and the other on a microfluidic droplet [42, 43]. For both of dissociation techniques, it is possible for multiple cells to be captured together (doublet) [44] and for no cells to be captured at all (empty droplets). These doublets violate the basic premise of scRNA-seq technology and have serious implications for the inferences of downstream analysis. To detect and remove these exception captures, quality control is performed for reads data generated by sequencing prior to analysis. Quality control is to determine whether the capture is abnormal by examining the distribution of three indicators: count depth, the total number of genes detected, and the proportion of transcription from mitochondrial genes [45, 46]. A captured sample with unexpectedly high counts and abundant genes detected is likely to be considered doublets. Several recently released tools provide a better insight and robustness for detecting doublets (DoubletDecon [47], Scrublet [48], Doublet Finder [49]). However, the linear embeddings extracted from original count data by them do not fit well the complex interrelationships between cells.

Solo

Recently, Bernstein et al. proposed Solo, a tool for identifying doublets based on semi-supervised deep learning [23]. Following previous computational doublet detection approaches, Solo assumes that most captures of reads data generated by sequencing are singlet and simulated doublets can be generated in silicon to approximate the distribution of doublets based on the observed data. Given observed data, Solo randomly selected several single cells and integrated them into a simulated doublet. By repeating this process for n times, a simulated doublet set N_d was derived. Unlike the previous machine learning-based doublet detection methods with linear embeddings derived by Principal component analysis (PCA) [50] and singular value decomposition (SVD) [51], Solo employed the variational autoencoder (VAE) [19], a nonlinear embedding method, to learn the representations of cells. VAE encodes gene expression x to compressed representation y by learning a map function $f\left(y|x\right)$ and decodes the compressed representation back to the original space by learning a reverse transform $g\left(x|y\right)$. By minimizing the probability of the original count matrix under a negative binomial distribution, Solo learned a representation of observed cells. Following the cells embeddings, Solo appended a neurol network to classify the doublet status. Simulated doublets and observed data were used to train the discriminative classifier, and experimental datasets with doublet were employed to evaluate the model.

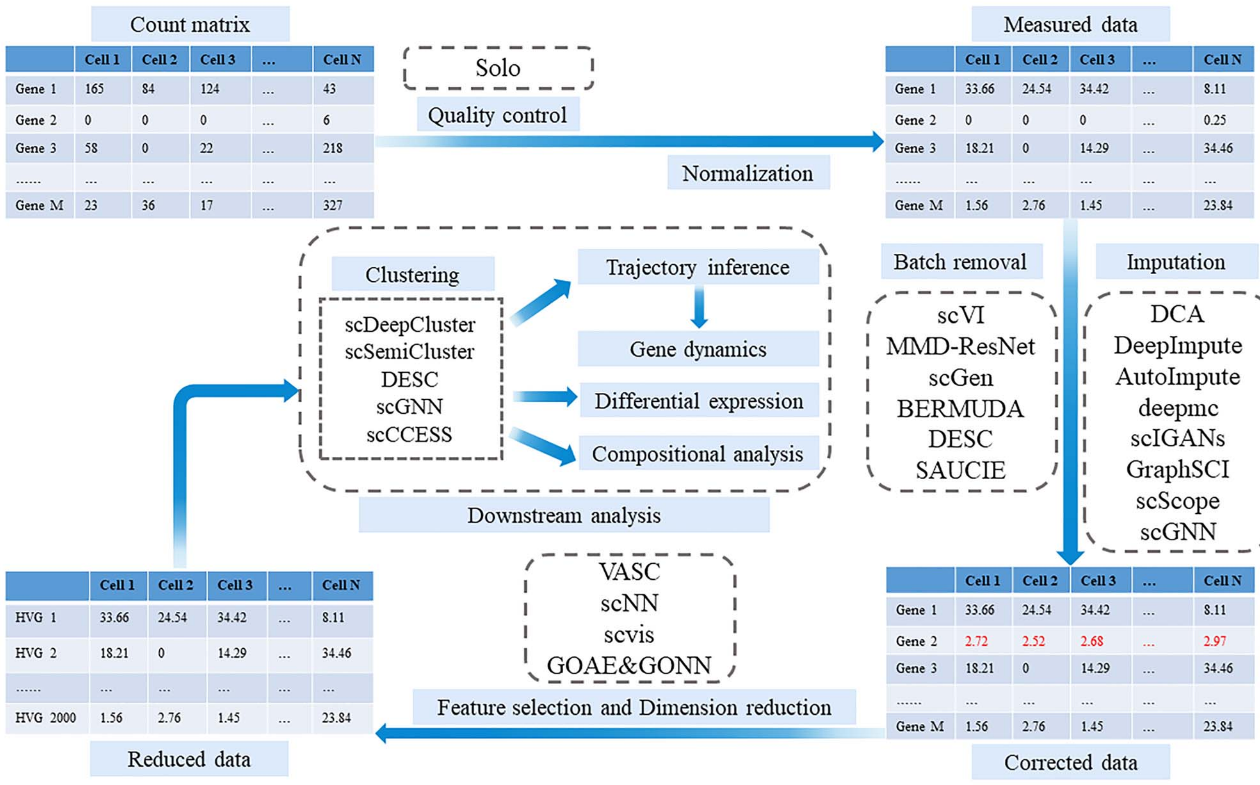

Figure 1. Workflow of typical single-cell RNA-sequencing analysis that covering deep learning-based algorithms.

Normalization

The analysis tasks of scRNA-seq data like clustering can uncover the heterogeneity between different cell populations. However, the variations between cells are not only come from biological differences, but also may come from technical factors [52]. Due to the variability inherent in each step of sequencing, the sequencing depth between cells may vary greatly, even if the same cell is sequenced twice and the counting depth obtained may vary. Normalization is a necessary step in the analysis of single-cell RNA sequencing data in order to remove the impact of technical variations in the analysis count data, while preserving as much as possible the biological differences between cells. Several bulk RNA sequencing analysis methods have been applied to scRNA-seq analysis, the most classic approach for normalization is ‘counts per million’ (CPM). It removes the technical biases by applying size factor that is proportional to the count depth of each sample. However, the differences in gene length may cause bias. To address this issue, many variations of CPM normalization are proposed. Among them, three most commonly adopted variations include Reads per kilobase million [53] (RPKM), fragments per kilobase million [54] (FPKM) and transcripts per million [55] (TPM). Both RPKM and FPKM are divided by the sequencing depth and then by the gene length. The only difference is that the former is for single-end sequencing and the latter is for pair-end sequencing. The normalization order of TPM is opposite to that of RPKM and FPKM, making it easier to compare samples directly and more recommended for RNA-seq data. However, the characteristics of single cell data (sparsity, fault zero value, etc.) require normalization methods for scRNA-seq data specifically [56, 57].

Data correction

The normalized scRNA-seq data may still contain factors that are irrelevant to the analysis. The primary purpose of data correction is to further remove technical factors and biological confounders that are not relevant to the study. And the correction for technical variations may be crucial to reveal potential biological signals. Among them, one of the most prominent technical factors is the batch. Different conditions and processes experienced by the cells during the experimental operation may create batch effects, such as different chips for sequencing, different sequencing channels and different time points of data collection. Batch effect correction usually handles samples or cells in the same experiment, while data integration usually handles data from different laboratories. In the several years, many batch correction and data integration methods have been proposed to address this issue, such as Mutual Nearest Neighbors (MNN) [58], Canonical Correlation Analysis (CCA) [59], Scanorama [60], Harmony [61] and BBKNN [62]. These methods are based on pairwise analysis and are less time-efficient, as only two batches of data sets are corrected at a time. Moreover, the final result of correction will vary due to the order of the batches. Recently, deep learning techniques are adopted to improve the accuracy and time efficiency of correction. Several deep model-based batch correction methods have been developed and drawn wide attentions.

scVI

Lopez et al. proposed a scalable multitasking tool, single-cell variational inference (scVI) [19], for learning low-dimensional

Table 1. Summary of deep model-based scRNA-seq analysis methods

Algorithm	Functions	Language	Underlying deep models	Accesses	Suitable datasets
Solo [23]	Doublets identification	Python	Neural networks and Variational autoencoder	https://github.com/calico/solo	Large
scVI [19]	Batch correction/Normalization	Python	Hierarchical Bayesian model and variational autoencoder	https://github.com/YosefLab/scVI	Noisy, large
MMD-ResNet [24]	Batch correction	Python	Deep residual network	https://github.com/ushaham/BatchEffectRemoval	Noisy
scGen [25]	Batch correction	Python	Variational autoencoder	https://github.com/theislab/scGen	High-dimension
BERMUDA [26]	Batch correction	Python, R	Autoencoder, transfer learning	https://github.com/txWang/BERMUDA	Noisy, large
DESC [27]	Batch correction/Clustering	Python	Autoencoder	https://eleozzr.github.io/desc/	Noisy, large
SAUCIE [20]	Batch correction/Imputation/Visualization/Clustering	Python	Autoencoder	https://github.com/KrishnaswamyLab/SAUCIE/	Noisy, large
DCA [28]	Denoising and Imputation	Python	Autoencoder	https://github.com/theislab/dca	Noisy, large
DeepImpute [29]	Imputation	Python	Multiple neural networks	https://github.com/lanagarmire/DeepImpute	Dense
Deepmc [30]	Imputation	Python	Deep matrix factorization	–	Dense
AutoImpute [31]	Imputation	Python, R	Autoencoder	https://github.com/divyanshu-talwar/AutoImpute	Sparse, small
scIGANs [32]	Imputation	Python, Shell, R	Generative adversarial networks	https://github.com/xuyungang/scIGANs	Noisy, small
GraphSCI [33]	Imputation	Python	Graph convolution network and Autoencoder	https://github.com/GraphSCI/GraphSCI	Sparse, small
scScope [21]	Imputation/Batch correction/Clustering	Python	Recurrent networks and Autoencoder	https://github.com/AltschulerWu-Lab/scScope	Noisy, large
scGNN [34]	Imputation/Clustering	Python	Graph neural networks and Autoencoder	https://github.com/juexinwang/scGNN	Noisy, large
VASC [35]	Dimension reduction and Visualization	Python	Variational autoencoder	https://github.com/wang-research/VASC	Noisy, large
scNN [36]	Dimension reduction	Python	Neural networks	http://sb.cs.cmu.edu/scnn/	Sparse
GOAE&GONN [37]	Dimension reduction/Clustering	Python	Neural networks and Autoencoder	–	Large
scvis [38]	Dimension reduction and Visualization	Python	Neural networks	https://bitbucket.org/jerry00/scvis-dev/	High-dimension
scDeepCluster [39]	Clustering	Python, R	Autoencoder	https://github.com/ttgump/scDeepCluster	Noisy, small
scSemiCluster [40]	Clustering and annotation	Python	Autoencoder	https://github.com/xuebaliang/scSemiCluster	High-dimension
scCCESS [41]	Clustering	R	Autoencoder	https://github.com/gedcom/scCCESS	High-dimension

'-' indicates that the authors did not release the code of corresponding algorithm.

representations and analysis of scRNA-seq data based on hierarchical Bayesian and deep learning. For raw scRNA-seq data with batch ID, scVI uses an autoencoder based on neural networks to model the observed expression. The autoencoder architecture consists of a encode layer and a decode layer. First, the encoder compressed the expression of single cells to a low-dimensional representation by a non-linear mapping. Then the decoder transformed the low-dimensional embedding back to the original space of the single cell expression data by another non-linear mapping. The difference between the input and the output of the autoencoder was minimized to learn the distributional parameters by using variational inference and a scalable stochastic optimization procedure. The compressed representation derived from the autoencoder can provide insight for visualization and clustering of scRNA-seq data. In addition, another neural network followed the autoencoder projects the compressed representation into the parameters of the zero-inflated negative binomial (ZINB) [63] distribution, which makes scVI competent for batch effect removal and normalization with great performance and efficiency.

MMD-ResNet

Shaham et al. developed a batch effects removal method MMD-ResNet [24] based on the deep residual network for mass cytometry and scRNA-seq data. It assumed that the moderate difference between the data distributions from two or multiple batches can be correct through an approximate transforming function. Residual neural network is a typical deep learning method and is popular in the field of image recognition [64]. ResNet consists of multiple blocks that are sequentially connected, and each block containing a residual item and a target item can learn a map approximate to the target. In MMD-ResNet, datasets of two batches are designated the source and the target. Then MMD-ResNet is fed with the source sample and is trained to learn a map function that corrects the distribution of the source to be similar to the that of the target by minimizing the Maximum Mean Discrepancy (MMD) between two batches.

scGen

Lotfollahi et al. proposed a deep transfer learning model that integrates variational autoencoder and the vector arithmetics, namely scGen, to correct the batch effects of scRNA-seq data [25]. A variational autoencoder have the same network structure as classical autoencoders, but it maximizes the possibilities for each sample to approximate the distribution in the original space. scGen starts by inputting the expression matrix of genes into the encoder and projecting the original space into a low-dimensional space. Then a vector arithmetic is employed to learn a vector that can perform linear correct of batch effects in the low-dimensional space. After that, scGen applies a decoder layer to project the low-dimensional variable back to the original expression matrix of scRNA-seq data with batch effects removal.

Bermuda

Wang et al. were inspired from the advances of deep transfer learning for the domain adaption issues, and proposed a batch effect removal approach, BERMUDA [26], based on deep autoencoders. BERMUDA aligns the cell clusters from different batches to remove batch effects based on the similarities between cell populations. Specifically, graph-based clustering was applied to separate the cell populations for each batch. And the similarity score of each pair of clusters among different was evaluated based on their correlation. Then, the unaligned expression data after clustering were input into an autoencoder. The difference between the input and output of the autoencoder were defined as the reconstruction loss, and the distribution difference in the middle node of autoencoder between each pair of similar clusters was computed and defined as the transfer loss. By combining the transfer loss and the reconstruction loss, BERMUDA obtained a low-dimensional representation of raw data with batch effect corrected. Furthermore, a regularization parameter is added to the loss function to prevent overfitting.

DESC

An unsupervised embedding and clustering method based on deep learning, DESC, can accurately cluster scRNA-seq data and remove batch effects by iteratively self-learning processes [27]. Based on the assumption that the batch effect is considered to be removed as long as the technological variations are smaller than the biological differences, DESC firstly initializes the parameters of clustering by a stacked autoencoder network. Each layer of the stacked autoencoder network is an autoencoder and its input is the output from the previous layer. And the output of the bottleneck layer is inputted to the iterative clustering neural network. DESC applies Louvain's clustering method [65] to initialize the cluster centers and transfers each cell through an iterative process to its nearest cluster centroid, which can gradually reduce the batch effect of cells. After the iterative procedure and optimizing objective function of clustering, the stacked autoencoder network can learn a low-dimensional representation from the expression data of scRNA-seq with batch effect removal. In addition, DESC allows users to set the number of hidden layers and nodes depending on different situations.

SAUCIE

One challenge presented by deep learning is how to interpret the mechanism of model and the reasons of the significant performance. To address this issue, Amodio et al. developed SAUCIE [20], a deep neural network with customized regularization items on multilayers, providing multilevel interpretable analysis of scRNA-seq data. Similarly, SAUCIE applied the autoencoder neural network to learn a unified low-dimensional representation of gene expression data of single cell. At first, the raw data were fed into the autoencoder and were performed nonlinear dimensionality reduction gradually through the network. In the deep autoencoder neural network, different layers focused on different aspects of embedding, including batch effect correction, imputation and clustering. To make the learned representation interpretable, SAUCIE introduced two novel regularizations to the framework. For clustering, the information dimension regularization was applied on the sparse encoding layer and facilitates near-binary activations. For batch correction, maximal mean discrepancy regularization makes the probability distribution of the bottleneck layer of the sample similar, and removes complex nonlinear batch effects. Besides, SAUCIE also provides denoising and imputation, visualization and clustering of scaled scRNA-seq data.

Denoising and imputation

Another type of technical difference is noise or missing values, one of main sources is the dropout event, also referred to 'false zero' [9, 63]. In fact, in some scRNA-seq data sets without effective quality control, more than 80% of expression values are zero [14]. These zeros fall into two categories, one is 'true' zero where the gene of a cell is not expressed. The other is 'false' zero, where gene expression values are not observed due to low capture rates, inadequate sequencing depth, and other technical factors [66]. However, it is quite challenging to distinguish the 'false' zero from the 'true' zero, as it has no dominant characteristic. Despite continued advances in sequencing techniques, the dropout events are still common in scRNA-seq data and introduce technical variability and high noise that affect multiple downstream bioinformatics analysis tasks. One of the main approaches to address this issue is imputation or denoising. The imputation approach was developed to recovery the missing values of bulk-RNA-seq or microarray data. And many imputation methods have been applied to address the dropout events in scRNA-seq data. Kim et al. proposed a local least-squares imputation method named LLSimpute [67], which imputes each missing value with a linear combination of similar genes. However, scRNA-seq data is more sparse, random and fluctuated compared with bulk-RNA seq data. These challenges have led to the emergence of numerous imputation approaches based on different principles designed specifically for scRNA-seq data. These methods can be divided into three types: model-based, smoothing, low-rank matrix. Prabhakaran et al. proposed BISCUIT [68], an iterative imputing method based on the Dirchlet process mixture framework, to learn and infer the missing values of gene expression of single cells. MAGIC [69] is a graph imputation approach based on Markov affinity. It constructs a Markov transition matrix to impute the false zeros of the cells by applying diffusion geometry. However, it may impute the true zeros where genes are not expressed and remove the biological variations that can be used to explore the heterogeneity of cell populations. scImpute [70] is developed to address the overcorrection and only imputes those 'false' zeros caused by the dropout events. scImpute first estimates the dropout probability distribution of genes based on a mixture model. Then the genes with a high dropout probability of each cell are imputed according to the expression values of same gene in similar cells with a low dropout probability, which is considered gold-standard data. However, the estimate of dropout probabilities for genes of each cell depends on many factors [9], and a slightly biased estimation can make a big difference to the imputation. Kwak et al. proposed an ensemble approach, DrImpute [71], to recovery expression of genes. DrImpute performs consensus clustering and averages the expression values of genes in similar cells. Then the average expression values are used to impute the missing values of genes. Based on the assumption that the counts of single cells follow a Poisson-gamma mixture distribution, SAVER combines the weighted average expression values and observed expression of other genes in the same cell to recovery the true expression values of scRNA-seq data [72]. However, these statistical and machine learning methods not only are computationally intensive and not scalable for those datasets with millions of cells, but also may fail to extract nonlinear structures from the count matrix of scRNA-seq data.

DCA

By extending the autoencoder network with two noise models, Eraslan et al. developed DCA [28], a denoising and imputation method specifically for scRNA-seq data. Many previous imputing and denoising approaches assume that scRNA-seq data followed a zero-inflation negative binomial distribution. However, different from the sequencing technology based on reads, the count data based on unique molecular identifier (UMI) is more likely to follow the negative binomial distribution. To provide a more comprehensive analysis for scRNA-seq data, DCA applied two noise models specifically for different count technologies. The autoencoder learns a compressed representation of the scRNA-seq data by minimizing the reconstruction loss between the input and output data. The compressed representation is a mapping from a high-dimensional feature space to a low dimensional one, capturing the underlying data structure. In addition, DCA infers the distribution parameters based on the compressed representation, and are further applied to denoise and impute the gene expression matrix of cells.

DeepImpute

To accurate impute the scRNA-seq data, Arisdakessian et al. proposed a divide-and-conquer imputation algorithm DeepImpute [29] based on deep neural-network models with dropout layers and loss functions. To reduce the complexity of imputing the datasets of large scale, DeepImpute starts by constructing multiple neural networks. Each neural network consists of four layers: an input layer, a fully connected layer, a dropout layer for preventing overfitting, and an output layer. DeepImpute randomly selected genes to impute at a rate of 0.5 mean variance. A weighted mean square error (MSE) loss function is employed to update the weights of neural networks. After training, DeepImpute imputes the zero values of the expression matrix by the output of the networks. Moreover, DeepImpute are also scalable for dataset with large volumes, as its divide-and-conquer manner.

Deepmc

Aim to recovery the expression matrix of genes due to the insufficient capturing rate and other technology factors, MONGIA et al. considered imputation as a standard matrix factorization issue. Motivated by the success of deep learning, MONGIA et al. developed deepMc [30], a deep matrix factorization method, to recovery the gene expression data of single cells. deepMc only retains the top 1000 most dispersed genes. Then the preprocessed count matrix of scRNA-seq data is input into a deep feedbackward neural network consisting of four layers. Formally, the deep neural network can be defined as followed:

$$Y = L_1 L_2 L_3 X. \tag{1}$$

where Y denotes the estimated expression matrix after imputation, L represents the different layers of basis, and X represents the final layer of coefficients. The gene expression matrix can be recovered through the coefficient layer and three basis layers.

AutoImpute

Inspired by the excellent performance of deep learning technologies in bulk RNA sequencing data, Talwar et al. proposed AutoImpute [31], an autoencoder-based imputation approach for sparse expression matrix of scRNA-seq data. Autoencoder can derive the latent distribution from original data space and recover the unobserved expression values while preserving biological differences. AutoImpute starts with screening out the bad genes from the raw gene expression matrix of scRNA-seq data. After normalization, the preprocessed expression matrix is input into the overcomplete autoencoder model. The encoder of the model projects the expression matrix into a low-dimension embedding and then the decoder maps the low-dimension embedding back to the original space. AutoImpute learns these two transformation functions by minimizing the reconstruction loss, and applies the autoencoder to reconstruct the imputed gene expression matrix of scRNA-seq data.

scIGANs

Motivated by the success of generative adversarial networks (GANs) [73] in generating realistic images and realistic scRNA-seq data, Xu et al. developed scIGANs [32] which derived the non-linear dependencies between genes from complex, multiple type cells to impute the missing values of scRNA-seq data based on the GANs. scIGANs reshapes the expression matrix of each cell into an image, and feeds them into the GANs. On the one hand, the generative model projects a random 100-dimensional representation into the expression matrix of genes to generate simulated images. On the other hand, the discriminator model determines whether the images are genuine or simulated. These two models play and compete with each other to improve the performance of both. Once the discriminator fails to distinguish the simulated images generated by the generator from the authentic ones, scIGANs employs the generative model to produce the simulated gene expression matrix of defined cell type. With the simulated data, scIGANs further applies the k-nearest neighbors (KNN) [74] method to impute the missing values of the gene expression matrix of real scRNA-seq data.

GraphSCI

Rao et al. considered gene–gene relationships as an important factor in imputation for scRNA-seq data that ignored by existing expression recovery methods, and proposed an effective and robust imputation approach, GraphSCI [33]. Taking full advantages of the latent embeddings of similar cells and the interactions between genes, GraphSCI combines graph convolution network (GCN) [75] and Autoencoder neural networks to impute the missing values in gene expression matrix of scRNA-seq data. Different from other deep learning-based imputation methods, GraphSCI feeds genes' interactions and expression matrix into the GCN and autoencoder network, respectively. By transforming the expression recovery into the node recovery task in the undirected gene graph, GCN learns the low-dimensional representations of interactions between genes. Meanwhile, the autoencoder neural network of GraphSCI derives the nonlinear dependencies and structures between cells from the gene expression matrix of scRNA-seq data. Moreover, a generative model combines the learned Gaussian distributions and ZINB distributions to reconstruct the gene interactions and recovery the gene expression matrix.

scScope

Based on the fact that subsets of genes are usually correlated, Deng et al. assume the co-expression patterns of genes can be explored by sufficient single cells and develop scScope [21] to impute the dropout measurements of scRNA-seq data. scScope iteratively imputes the missing values of gene expression matrix by adopting a recurrent network architecture. The self-learning model consists of three layers including an encoder, a decoder and an imputation layer. scScope starts with the batch effects removal of the expression matrix of scRNA-seq data. Then the batch corrected expression matrix is fed into the encoder for learning a low-dimensional representation. The decoder is applied to project the representation back to the original space, which can further denoise the data. After decoder, scScope uses the self-correcting imputation layer to recover the missing values. The output of the imputation layer is used to merge the corrected expression matrix for imputing the dropout measurements. Furthermore, scScope repeats this procedure to gradually improve the imputation performance and eventually learn a low-dimensional representation that can be used for several downstream analysis tasks.

scGNN

With the powerful graph neural network (GNN) to learn representations and node relationships of the whole graph in a global insight, Wang et al. proposed scGNN [34] by applying multi-modal autoencoders to formulate and capture heterogeneous biological relationships of cells for scRNA-seq data. scGNN is a hypothesis-free framework and focuses on imputation and cell-type clustering analysis. In detail, it constructed a graph for cells based on the model of left-truncated mixture Gaussian to simulate regulatory signals for a specific cell type, which can reduce the noise in the signals. Then scGNN built a GNN of cells with dynamic pruning to model the biological cell relationships and their underlying gene regulation patterns from the bottom up. scGNN further learned a graph embedding with low dimensions by pooling the entire graph to represent the latent topological relationships of cells. The learned graph embedding can be applied to perform various preprocessing procedure and analysis tasks, such as imputation, clustering and annotation, cell trajectory inference, etc.

However, any imputation and denoising approaches are not perfect and may result in insufficient or excessive noise correction in the data. Furthermore, it has been reported that imputation methods for the missing values may introduce the wrong related signals [76]. Since it is very challenging to evaluate whether missing values are filled properly in practical applications, users should choose specific methods carefully according to their own requirements and characteristics of data.

Feature selection and dimension reduction

There are more than 25 000 genes in the scRNA-seq dataset of human, most of which do not express or provide effective information [7]. Even with quality control, an scRNA-seq dataset that filters out zero-count genes may have more than 15 000 dimensions in the feature space. Computational analysis of datasets with such amount dimensions is time-consuming and memory intensive. Moreover, the high dimensionality causes small differences between cells and thus make it difficult to infer cells types, that is known as 'curse of dimensionality' [77]. To reduce noise in the data and the computational cost of downstream analysis tasks, and facilitate data visualization [78], dimension reduction is usually performed on the dataset. Feature selection is the initial step of dimension reduction of scRNA-seq dataset to screen out informative highly variable genes (HVG) that contribution to variability of data [79]. Depending on the analysis and the complexity of the data set, 1000 to 5000 HVGs are typically screened for downstream analysis. After feature selection, dimension reduction algorithms are usually applied to further compress the expression matrix of scRNA-seq data into a low-dimensional representation while capturing all the information [80]. The compressed representations are used to perform data visualization, which attempts to optimally present the structure and distribution of scRNA-seq data in two or three dimensions. In general, many approaches provide services by combing dimension reduction and data visualization. Principal component analysis (PCA) [50] is one of the most common linear dimension reduction and data visualization methods. It is widely applied in bulk RNA sequencing analysis and has recently become popular in scRNA-seq data analysis. PCA projects a set of potentially correlated variables into a set of linearly unrelated variables through orthogonal transformation, which are called the principal components. Compared with the nonlinear methods that capture more information of the original data through fewer dimensions, PCA has a consistent interpretation of the distance in the reduced dimension space in all regions of the space. In fact, it is usually a preprocessing step before dimension reduction approaches based on nonlinear structures, and lays the foundation for many downstream analysis tasks such as clustering and trajectories inference. Diffusion maps [81] are a common dimension reduction and visualization method for nonlinear data. It converts spatial distance into a probability of state transition to determine the direction of the random walk, and is therefore usually used to determine the trajectory of cell development. For single cell expression profiles, diffusion maps method is helpful to highlight the heterogeneity of cell populations. t-SNE [82] is a common dimension reduction and visualization method for nonlinear scRNA-seq data or manifold data with high dimensions. It converts the similarity of data points in the original space and embedded space into Gaussian joint probability and random T-distribution, respectively. Then, the Kullback–Leibler (KL) divergence of the joint probabilities in the two spaces is used to evaluate the visualization. t-SNE focuses on preserving local structure while sacrificing global structure, thus ignoring latent relationships between groups, which can be mitigated by initializing the points with PCA. However, the computational complexity of t-SNE is high and can take several hours in millions of sample data sets, while PCA can be completed in seconds or minutes. Another nonlinear dimension reduction and visualization method Uniform Manifold Approximation and Projection (UMAP) [83] is arguably one of the best ways to show the underlying topology of scRNA-seq data, and has faster speeds and the ability to be applied to larger data scales than t-SNE. It first computes the distances between points in a higher-dimensional space, projects them into a lower-dimensional space, and calculates the distances between points in that lower-dimensional space. It then uses stochastic gradient descent to minimize the difference between these distances.

In recent years, many researches have focused on data visualization by applying neural networks. The activation functions of neural network can project the data from the original space into the latent space with low dimensions, where is more suitable for visualization. In this section, we introduce four advanced dimension reduction and visualization tools for scRNA-seq data based on neural networks.

VASC

Wang et al. developed VASC [35], a deep generative model based on variational autoencoder, to facilitate the dimension reduction and visualization for scRNA-seq data. The aim of VASC is to model a posterior distribution $P(\upsilon|X)$ from the original space to the compressed embedding space. However, the best distribution $P(\upsilon|X)$ is usually incomputable. To address this issue, VASC applies the variational autoencoder to approximate the posterior distribution by the variational distribution $Q(\upsilon|X)$. The architecture of VASC mainly consists of five parts, including dropout layer, encoder, sample layer, decoder and zero-inflated layer. The dropout layer randomly adding noise to the expression matrix of scRNA-seq data to prevent the model from overfitting. The encoder model contains three layers of fully connected neural network, which is aim to generate the posterior parameters of a Gaussian distribution for $Q(\upsilon|X)$. The first layer of encoder network applies PCA transformation to preserve the underlying global structure. The sample layer then models the low-dimensional representations by the variational distribution generated by the encoder network. Another three-layer fully connected network decoder projects the compressed representations back to the original space. Immediately following the decoder is an additional ZI layer, which uses the double-exponential distribution to simulate the dropout events. Moreover, the back-propagation is performed in ZI layer based on the Gumbelsoftmax distribution [84]. After training, VASC can obtain a two-dimension representation for visualization.

scNN

Considering that unsupervised methods cannot directly perform discriminant analysis, Lin et al. proposed four types of deep neural network models for dimension reduction [36]. All models contain the input layer, one hidden layer and the output layer, while one model also contains an additional hidden layer, one model contains TF and PPI nodes that are only connected to their corresponding genes, and one model contains both an additional hidden layer and TF and PPI nodes. In these models, the hidden layers are applied to encode the expression matrix of scRNA-seq data, and the TF and PPI nodes can provide sufficient prior biological knowledge for cell type identification. In addition, the number of parameters of the neural network can be greatly reduced so that the model can be simplified and overfitting can be avoided, with TF and PPI nodes just linked their corresponding genes. Finally, the output layers of these models transform the representations into the possibilities for each cell type, that can be used to perform further visualization, cell type classification and functional analysis.

GOAE&GONN

Inspired by the hierarchical structure of NN, Peng et al. developed two deep neural-based methods for dimension reduction on supervised way and unsupervised way, named GONN and GOAE [37], respectively. Similar to NN, both GONN and GOAE incorporate with gene ontology (GO), which provides sufficient and reliable prior biological knowledge to excavate the underlying biological mechanism. In GOAE and GONN, the GO terms in biological process and molecular function are partially connected to their corresponding genes and constructed as a neural network. GOAE is an unsupervised method for dimension reduction by applying an autoencoder. The GO neural network and a followed fully connected hidden layer make up the encoder. At the same time, this hidden layer and mirrored GO neural network form the decoder. After training, GOAE can learn a latent embedding with low dimensions for several downstream analysis. GONN is a supervised method for dimension reduction and clustering. Similar to GOAE, GONN also constructs the GO neural network, and add an another fully connected hidden layer before the output layer. After training, a low-dimension representation can also be generated by GONN for clustering analysis.

scvis

To capture the latent variables of expression matrix of scRNA-seq data, Ding et al. proposed a deep generative model, scvis [38], for interpretable dimensionality reduction. Based on the assumption that the expression matrix of scRNA-seq data is a random vector of a low-dimensional representation, scvis constructs a feedforward neural network to learn the low-dimension representation of the scRNA-seq data. The neural network consists of the input layer, multiple fully connected hidden layers and the output layer. The expression matrix of scRNA-seq data is fed into the input layer, and the output of scvis is the parameters of the variational distribution that used to sample and generate the low-dimension representation. Generally speaking, scvis can preserve the global structures of the scRNA-seq data while providing an efficient way to interpret the biological mechanism.

It is worth noting that these data visualization methods can only be used for theoretical exploration, but not as a standard to verify the correctness of the theory.

Downstream clustering analysis

Clustering is usually the initial step for downstream analysis of scRNA-seq data and the key step in identifying cell types. It is an unsupervised method and provides the insights for understanding of the biological mechanism [85]. Cells are divided into several clusters by similarity of their gene expression values, which is generated by measuring the distance of low-dimension representations after dimension reduction. Euclidean distance is a commonly used distance metric, which is calculated by the expressed value of principal components. Other available distance metrics including Pearson's correlation, Spearman's correlation and cosine similarity focus on the relative distance of values, making them more scalable for libraries of different sizes [14]. With considerable progress has been made in recent years, various clustering methods are available. These algorithms can be divided into three main categories including K-means clustering, hierarchical clustering and consensus clustering. As the most widely used clustering algorithm, k- means starts with randomly selecting K objects as the initial centroids. By calculating the distance between each object and each centroid as the similarity, k-means assigns each object to the centroid closest to it. As each sample is allocated, the clustering center is updated according to the existing clustering points. This process is repeated until some termination condition is met [86]. In addition, the initial category number of the k-means algorithms, k, needs to be manually specified, that is often unknown in advance and needs to be determined heuristically. Lloyd's method [87] is the most typical k-means algorithm, and is scalable to the size of the applied dataset. However, it does not necessarily find the global minimum, but falls into the local minimum because of its greedy strategy. Moreover, k-means algorithms tend to generate clusters with equal size and will fail to identify rare cell types. To solve this issue, Grün et al. proposed a K-means-based clustering method for detecting rare cell types, RaceID [88], while SIMLR [89] develops the K-means algorithm by learning an adaptive distance measure for each dataset. Hierarchical clustering is another common clustering algorithm for scRNA-seq data. There are two types of hierarchical clustering algorithms, namely, divisive and agglomerative methods. Divisive hierarchical clustering starts by clustering all cells into a cluster, and then iteratively divides a cluster into multiple clusters according to certain criteria until each cell is a cluster. Conversely, in the Agglomerative hierarchical clustering algorithm, each cell starts as a cluster. Then it iteratively merges the two closest clusters into a new cluster according to certain criteria until all cells belong to a cluster. Hierarchical clustering algorithms are inevitably time and memory expensive, which makes them inappropriate for large data sets. BISCUIT is a Bayesian clustering method based on the hierarchical Dirichlet model [68]. It iteratively performs clustering process with technical variation removal. CIDR [90] is a stable hierarchical clustering method with an additional implicit imputation model specified for dropout events. To speed up the clustering process, various approaches [91–93] adapt the hierarchical clustering methods by performing dimension reduction after each iterative process. Consensus clustering [94] algorithm integrates the results of multiple clustering algorithms to find a consensus cluster, also known as cluster aggregation or cluster integration. It depends on multiple iterations of the selected clustering method over the subsamples of the dataset. Consensus clustering algorithm induces sampling variability by subsampling data sets several times, which provides clustering stability. In general, it can determine the number of possible clusters in a data set and provide quantitative evidence for cluster members. SC3 [95] provides robust clustering performance by applying multiple k-means process with different initializations. SAFE-clustering [96] is another consensus method based on aggregation clustering. It starts by applying four popular to cluster scRNA-seq data, and then the results of four algorithms are aggregated to generate consensus cluster labels. To overcome the limitations of K-means and the high cost of hierarchical clustering, various community detection methods have been developed to apply to scRNA-seq data. They partition data based on graph representation, which is usually generated by KNN algorithm. The obtained representations contain the underlying topology information of expression profiles of single-cells. Community detection methods identify clusters of nodes tightly connected instead of nodes with close distance, which greatly improves the speed of clustering process [65]. For more details and comprehensive comparisons of classic clustering methods, readers are encouraged to consult several overviews that focused on clustering analysis [97–99].

The 'curse of dimensionality' leads to poor performance of traditional clustering methods on a high dimension. Deep learning technology provides a solution to address the challenge by transforming the original scRNA-seq data with high dimensionality into a low-dimension representation.

scDeepCluster

Tian et al. developed a deep embedded clustering algorithm specified for scRNA-seq data, scDeepCluster [39], by applying the autoencoder integrated with ZINB model. scDeepCluster starts by feeding the expression matrix of scRNA-seq data into the encoder to learn a nonlinear transformation, which can project the original data space into a latent low-dimensional representation. In addition, the encoder integrates a random Gaussian noise model to denoise and impute the missing values, which can further improve the clustering performance. Then, the decoder incorporated with the ZINB loss model maps the latent variables back to the original expression matrix as its output. The ZINB loss model consists of three fully connected layers, namely mean, dispersion and dropout layer. Moreover, scDeepCluster uses the learned low-dimensional representation to carry out clustering process based on Kullback–Leibler divergence between two distributions. The outperforming computation efficiency provides scDeepCluster great scalability for the analysis of large scRNA-seq data.

scSemiCluster

To obtain more compact and pure clusters, Chen et al. developed a semi-supervised clustering and annotation model, namely scSemiCluster [40]. Similar to scDeepCluster, the autoencoder integrated with ZINB distribution model is also employed to map the original expression matrix into a latent space in scSemiCluster. Unlike other clustering methods, it employs both reference data and target data to train the model. For the learned low-dimensional representation, scSemiCluster performs both discriminative and generative clustering models on the target data. Then it minimizes the cross-entropy of the two clustering models to explore the inherent heterogeneity of the target data. Furthermore, scSemiCluster uses the structural similarity between the target cells and the reference cells to constrain the clustering of target data with the structural regularization of the original data.

scCCESS

Inspired by the powerful performance of ensemble learning algorithms and consensus clustering methods, Geddes et al. developed a clustering ensemble model namely scCCESS [41], based on the autoencoder artificial neural network. scCCESS starts by randomly sampling N times from the original expression matrix of scRNA-seq dataset to generate N sub datasets. Then, each sub dataset is fed into the individual autoencoder and is mapped to a low-dimensional representation by a nonlinear transformation. Subsequently, scCCESS employs an ensemble clustering model containing the standard k-means and kernel-based SIMLR to cluster each compressed representation. At last, the clustering results are integrated based on the fixed-point method to generate the ensemble output.

Performances of deep model-based methods on scRNA-seq datasets

Datasets

In this paper, 11 experimentally derived scRNA-seq datasets were applied to evaluate the performances of the imputation, visualization and clustering methods based on deep models. Data source, cell counts and clusters statistics of these datasets are summarized in Table 2. Among them, the first six datasets are considered the gold standard as they were benchmark labeled, and the last five datasets labeled computationally are considered the silver standard.

Biase's [100] dataset consists of 9 zygotes, 10 2-cell and 5 4-cell mouse embryos, with a total of 49 samples and 25 737 genes. The gene expression values are normalized by extracting the Fragments Per Kilobase Million (FPKM) with the option of upper quartile normalization. Yan's [101] dataset includes the transcriptomes of 90 individual human pre-implantation embryos, and the embryonic stem cells sequenced by Hiseq2000. The gene expression values are normalized by applying RPKM method, and 20 214 expression genes are used with their RPKM is greater than 0.1. The transcriptomes of Goolam's [102] dataset were derived from all blastomeres of 28 embryos at 2-cell, 4-cell and 8-cell stages, and from 12 individual cells at 16-cell and 32-cell stage embryos, with a total of 124 samples and 41 480 genes. Deng's [103] dataset consists of 268 individual cells from zygote, the late blastocyst and the adult liver. The transcriptomes are normalized by using RPKM method to generate read counts. Pollen's [104] dataset includes the transcriptomes of 301 single cells from 11 distinct populations in both of low-coverage and high coverage. The normalization of gene expression values was performed by applying transcripts per million (TPM) for all samples. The single-cell transcriptomes of Kolodziejczyk's [105] dataset contain 250 serum cells, 295 2i cells and 159 a2i cells derived from mouse embryonic stem cells across three different culture conditions respectively, with a total 704 individual cells and 38 653 genes. And batch effects derived from different experiments exist in three conditions. Treutlein's [106] dataset contains 80 single-cell transcriptomes with 23 271 genes derived from mouse lung epithelial cells at four different stages, and the gene expression levels were quantified by FPKM method. The transcriptomes of Ting's [107] dataset consist of 75 single cells circulating in mouse blood enriched for circulating tumor cells from 5 mice, 12 single cells from a mouse embryonic fibroblast cell line, 16 single cells from the nb508 mouse pancreatic cancer cell line, 12 single mouse white blood cells, and 34 dilutions to 10 or 100 picograms of total RNA from mouse primary pancreatic tumors from 4 mice. Moreover, the

Table 2. Summary of eleven scRNA-seq datasets for benchmark

Dataset	# of cells	# of genes	# of clusters	Normalization Type	Cell Resource	Standard	Suitable methods
Biase [100]	49	25 737	3	FPKM	Two and four-cell Mouse embryo	Gold	scGNN/scDeepCluster
Yan [101]	90	20 214	7	RPKM	Human preimplantation embryos and embryonic stem cells	Gold	DeepImpute/scSemiCluster
Goolam [102]	124	41 480	5	UMI	Four-cell mouse embryos	Gold	scIGANs/scSemiCluster
Deng [103]	268	22 457	10	RPKM	Mouse preimplantation embryos	Gold	scIGANs/scSemiCluster
Pollen [104]	301	23 730	11	TPM	Human	Gold	scScope/scGNN
Kolodziejczyk [105]	704	38 653	3	UMI	Mouse embryonic stem cell	Gold	scIGANs/scSemiCluster
Treutlein [106]	80	23 271	5	FPKM	Human lung epithelium	Silver	scIGANs/scSemiCluster
Ting [107]	149	29 018	7	RPM	Human pancreatic circulating tumor cells	Silver	scGNN/scSemiCluster
Usoskin [108]	622	25 334	11	RPM	Human neuron	Silver	scGNN/scGNN
Klein [109]	2717	24 175	4	UMI	Human embryonic stem cells	Silver	scGNN/scDeepCluster
Zeisel [91]	3005	19 972	9	UMI	Mouse cortex	Silver	scIGANs/scDeepCluster

Note that 'Suitable methods' in table represents imputation and clustering methods, respectively, for recommendation, and '#' denotes number.

gene expression values were normalized in reads per million (RPM). Klein's [109] dataset includes 2717 transcriptome samples derived from mouse embryonic stem (mES) cells with 24 175 genes. The gene expression values of dataset present unique molecular identifier (UMI)-filtered counts per cell without normalization after matching and mapping. The transcriptomes of Usoskin's [108] dataset contain 622 dissociated single cells dissected from the mouse lumbar dorsal root ganglion distributed over a total of nine 96-well plates. The expression levels were quantified in reads per million (RPM) values for 25 334 genes. Zeisel's [91] dataset includes the unique molecular identifier (UMI) counts from 19 972 genes in 3005 individual cells from the somatosensory cortex and hippocampus CA1 of mouse. Transcripts with gene expression values less than one were considered not expressed and dropped from further analysis for all datasets.

Data processing

The standard quality control (QC) process was performed at all datasets introduced above. For data filtering, only genes expressed in at least ten cells, and cells with at least one gene detected were kept. For Goolam's dataset, Kolodziejczyk's dataset, Klein's dataset and Zeisel's dataset that contain UMI counts, we applied the Scanpy [110] package of Python to perform the normalization of counts per million (CPM) on them. Moreover, the normalized or quantified expression values of all datasets were transformed to the log2 space.

Performances of imputation methods on scRNA-seq datasets

Note that it is difficult to evaluate imputation methods directly due to the challenge of defining the ground truth for datasets [28]. In this paper, we first simulated a scRNA-seq dataset by applying Splatter [111] to directly assess the imputation performance of deep model-based denoising and imputation algorithms. The simulated scRNA-seq dataset consists of 2000 cells with 2000 genes, and distributes in 6 groups. Five data recovery methods (DCA, DeepImpute, scScope, scIGANs and scGNN) were applied to impute the counts with dropout, respectively. Then we computed Pearson Correlation coefficient (PCC) and rooted mean squared error (RMSE) between the imputed counts and the true counts (Figure 2d). It can be observed that scGNN significantly outperforms others on both PCC and RMSE. DCA and DeepImpute have competitive PCCs at the cost of a very large RMSE. In contrary, scIGANs and scScope performs well on RMSE but relatively poor on PCC. It suggests that scGNN has excellent denoising and imputation performance, while other methods either ignore correlations between genes or introduce imputation bias into dataset.

To evaluate their imputation impact and improvements for downstream analysis, we then performed unsupervised clustering task on the imputed expression matrix derived from eight deep-based expression recovery methods respectively. Unsupervised clustering is usually the initial step for downstream analysis of scRNA-seq data and the key step in identifying cell types with similar expression profiles. In this paper, Louvain clustering algorithm that is one of the most common community detection methods is applied to identify clusters of datasets. Furthermore, the adjusted Rand index (ARI) [112], normalized mutual information (NMI) [113] and the Silhouette coefficient [114] are used to evaluate the unsupervised clustering performance. Silhouette coefficient is often used to interpret and validate the consistency of clustering results of a dataset. The value of Silhouette coefficient presents the tightness of a sample with its cluster assigned and separation to other distant clusters. The range of Silhouette coefficient is between $[-1, 1]$. The sample matches well to its cluster assigned and poorly matches to other distant clusters if the index is high. In particular, the silhouette value 0 means the sample is assigned to another cluster closest to one it belongs to; however, it is the same distance to both clusters. In this paper, we calculate the overall average Silhouette coefficient for an imputed scRNA-seq dataset. The adjusted Rand index measures the similarity between true and predicted clustering by comparing all pairs of samples that are whether assigned in the same or different clusters. The range of ARI also lie between $[-1.0, 1.0]$, where a value 1.0 presents perfect clustering and 0.0 stands for random labeling. Mutual information represents the increase of clusters information or the reduction of its uncertainty under the premise of given ground truth information. However, MI score tends to be higher than the actual amount of mutual information between the clustering information and the ground truth information as the number of clusters increases. Normalized Mutual Information (NMI) is developed to normalize clustering results with a large number of clusters by some generalized mean of predicted labels and ground truth labels, and scale the MI score between 0 (which means no mutual information) and 1 (which means perfect correlation).

Seven imputation methods (DCA, DeepImpute, AutoImpute, scIGANs, GraphSCI, scScope, scGNN; The authors of deepmc didn't share the codes.) were applied to above datasets, respectively, and then performed Louvain clustering algorithm on the imputed expression matrix. To explore whether imputation methods will improve the downstream analysis tasks (clustering results e.g.), we further applied Louvain clustering on the eleven datasets without imputation process. The values of ARI, NMI and Silhouette coefficient of eleven imputed matrixes are shown in Figure 2a-c. A table including results for ARI, NMI and Silhouette coefficient is also provided in Supplementary Data 1. It is worth noting that AutoImpute and GraphSCI consume too much memory to carry out, even on the Biase's dataset with smallest number of cells. Among the seven imputation methods, scIGANs and scGNN are the most prominent, having better performances on almost all datasets, and can achieve the great improvement of clustering results on several datasets, such as Usoskin's and Zeisel's. Besides, scGNN works most stably across all datasets in terms of the three metrics, as refer to the results without imputation process. Moreover, scIGANs is dominant on both Treutlein's and Zeisel's datasets, demonstrating its excellent performance of imputation and noise reduction on some datasets, as well as its instability across all datasets. In fact, none of the methods can guarantee that their imputation process will improve the clustering performance. Each imputation method has a different preference for the dataset. DeepImpute and scScope carried out better results on the 'small' datasets (with small number of cells) than on the datasets with large number of cells, due to their simple model structures. And for those methods with complex model and network structure, such as scGNN, more samples are required for training to optimize the hyperparameters. Therefore, scGNN presents outstanding imputation performance on the larger datasets including Usoskin, Klein and Zeisel. Moreover, all the five methods improved the clustering performance significantly for the Pollen's dataset. For datasets of Biase, Goolam, Kolodziejczyk and Klein; however, the clustering results after imputation process are even worse. This phenomenon suggests that

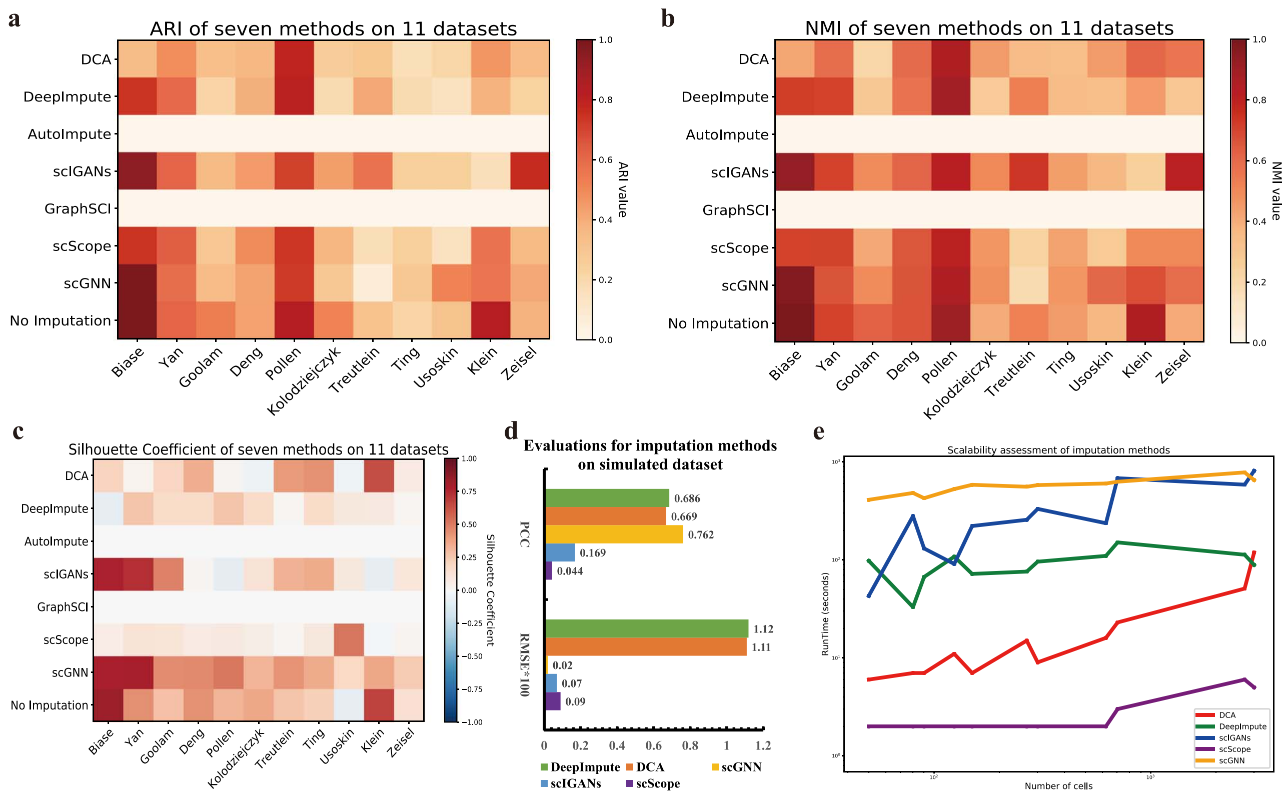

Figure 2. Imputation and Clustering performance comparison for 11 datasets after using seven imputation methods. (**a**) depicts the Heatmap of ARI. (**b**) presents the Heatmap of NMI. (**c**) presents the Heatmap of Silhouette coefficient. (**d**) depicts PCC and RMSE of simulated scRNA-seq dataset after imputing by five imputation methods. e shows the runtimes for five imputation methods on 11 real scRNA-seq datasets with different number of cells ranging from 50 to 3005. The closer to red means the bigger value, and the closer to white means the smaller value. 'No imputation' represents results of datasets without imputation process. Note that AutoImpute and GraphSCI consume too much memory to carry out, even on the Biase's dataset with smallest number (50) of cells.

expression recovery by imputation process may introduce some false correlation signals into the expression matrix. With the increasing number of cells in single-cell experiments, scalability has become an increasingly important evaluation criterion for single-cell data analysis algorithms. In this paper, we further measure the runtime of these imputation methods on datasets with different number of cells to evaluate their scalability (Figure 2e). The size of datasets ranges from 50 to 3005. scScope has best scalability across all datasets, and took 2 seconds to impute datasets within 1000 cells. Even for Zeisel's dataset with 3005 cells, scScope just took 5 seconds while other methods took several minutes. DCA also has excellent scalability, the runtime of it is almost linear with the number of cells. The other three deep model-based imputation methods DeepImpute, scIGANs and scGNN took much more time. The runtime of DeepImpute is less affected by the number of cells and more affected by the number of sub-network and predictors of it. Among these methods, scIGANs has the worst scalability for those datasets with large number of cells, mainly because it requires constant GANs execution to impute the expression matrix. scGNN took a long time to run on every dataset, and even a data set with only 50 cells took more than 400 seconds. This is mainly related to its complex algorithm structure. Except that, scGNN also has good scalability for datasets with large number of cells.

To further test whether these imputation methods could capture cell population structure in some complex scRNA-seq datasets, we applied the above five methods (DCA, DeepImpute, scScope, scIGANs, scGNN) to impute several datasets including Kolodziejczyk et al., Usoskin et al., Klein et al. and Zeisel et al. with 600 cells and 19 900 genes at least. After imputation procedure, we first conduct the principle comment analysis (PCA) to extract the latent features with 15 dimensions. Then we performed the uniform manifold approximation and projection (UMAP) on the latent features and drew a scatter plot on the 2-dimension UMAP space (Figure 3). For Kolodziejczyk's dataset (Figure 3a), it is observed that original dataset is dispersed into five clusters, and there is not enough separation between clusters. The expression matrixes imputed by DeepImpute, DCA, scGNN and scScope are clustered more compactly, and scGNN also tends to make the clusters relatively more separated. However, cells of one cell type are dispersed into several sub-clusters after imputing by DCA, DeepImpute and scGNN. Among the five methods, scIGANs has the best imputation performance and significantly improved the clustering results, keeping the cells within clusters compact and the cells between clusters as far apart as possible. For Klein's dataset (Figure 3b), DeepImpute, scScope and scIGANs imputation procedure make some cells more similar, leading to the poorer clustering performance than others. It suggests that intercellular heterogeneity may be masked by some false signals introduced by the imputation process, contrary to the purpose for which the imputation method (even the expression recovery process) was invented. For Usoskin's dataset (Figure 3c), scGNN has improved the clustering result and found several subtypes of cells from one cluster of original data. Zeisel's dataset has the largest number of cells and is fairly sparse. For this dataset (Figure 3d), scIGANs has outstanding imputation performance and significantly improved the clustering results. It not only helps to find the subtypes of

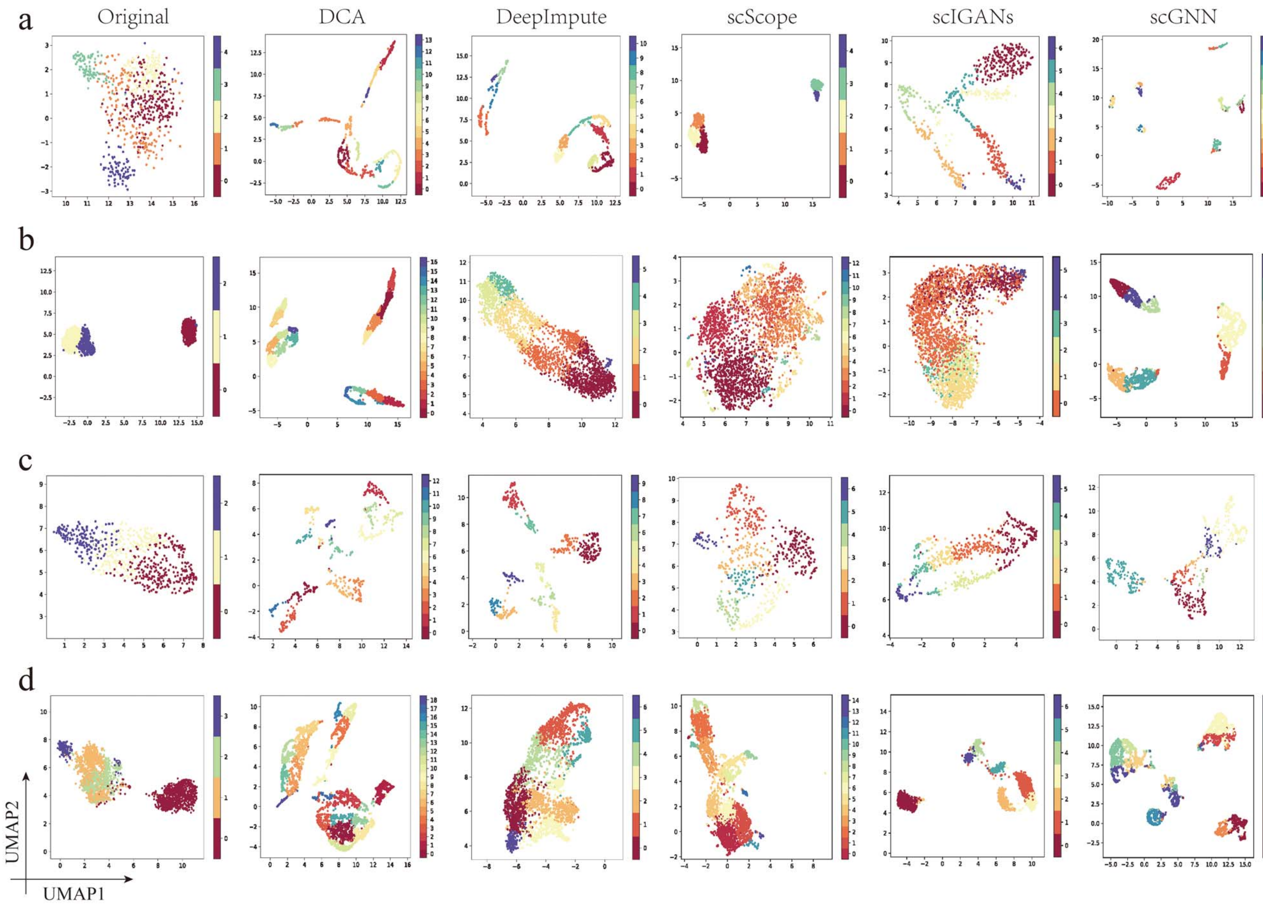

Figure 3. The UMAP visualization for four scRNA-seq datasets a, Kolodziejczyk et al. b, Klein et al. c, Usoskin et al. d, Zeisel et al. after imputation process from five methods, including DCA, DeepImpute, scIGANs, scScope and scGNN. 'Original' represents the results of four datasets without imputation process.

cells from one cluster of the original data, but also keeps the cells between the clusters separated from each other. At the same time, scGNN offers a visible lift for clustering, such as potential subtypes of cells.

Evaluation performances of clustering methods on scRNA-seq datasets

In this section, we compared the clustering results of four deep learning-based clustering algorithms (scDeepCluster, scSemiCluster, scGNN and DESC) that focused on clustering analysis and implemented by Python. To evaluate their clustering performance, we carried out them on elven datasets described in this paper (Table 2), respectively. Note that we used the default parameters provided by these methods, and the same batch size for all four methods. For scGNN, we applied the default clustering method Louvain. Similarly, ARI, NMI and Silhouette coefficient were applied to evaluate the clustering results of these methods (Figure 4a-c). To further compare the deep model-based clustering methods with traditional clustering algorithm, we directly performed Louvain clustering algorithm and FEATS [115], a multifaceted tool for batch correction and downstream analysis, on 11 scRNA-seq datasets applied in this paper and evaluate their clustering results. For ARI and NMI metrics, scSemiCluster performs almost dominant on almost all datasets, especially achieves ARIs of 0.979, 0.908 and 0.75, NMIs of 0.965, 0.88 and 0.728 for three datasets with large number of cells (Kolodziejczyk et al., Klein et al. and Zeisel et al.). It suggests that the cluster labels assigned by scSemiCluster match the ground truth of cells very well. scDeepCluster also performs excellent compared to traditional clustering algorithm Louvain, with significant improvements on several datasets, such as Yan, Goolam, Deng, Pollen and Zeisel. And scGNN performs most stable on almost all datasets in terms of ARI, NMI and silhouette coefficient, except on dataset of Treutlein. For silhouette coefficient, scGNN outperforms scDeepCluster, scSemiCluster, DESC and Louvain, which means bigger separation between clusters and better tightness of cells within clusters assigned by it. DESC also has great stability, especially on the metrics of ARI and NMI. Two traditional clustering tools Louvain and FEATS also perform well on several scRNA-seq datasets, such as Biase, Yan, Pollen and Klein. Besides, FEATS outperforms other clustering methods on dataset of Usoskin. Moreover, we evaluated the scalabilities of three deep model-based clustering methods as well as two traditional clustering algorithms by assessing their runtime carried on datasets with different number of cells (Figure 4d). It could be observed that deep model-based clustering methods are more time-consuming than classical clustering algorithm Louvain, due to their complex structure and deep layers. Among them, DESC shows excellent and linear scalability and took just 36 seconds when the number of cells increases to 3000, while other methods took hundreds of seconds. At the same time, scDeepCluster and scSemiCluster took tens of seconds when the number of cells is less than 1000. However, the runtime

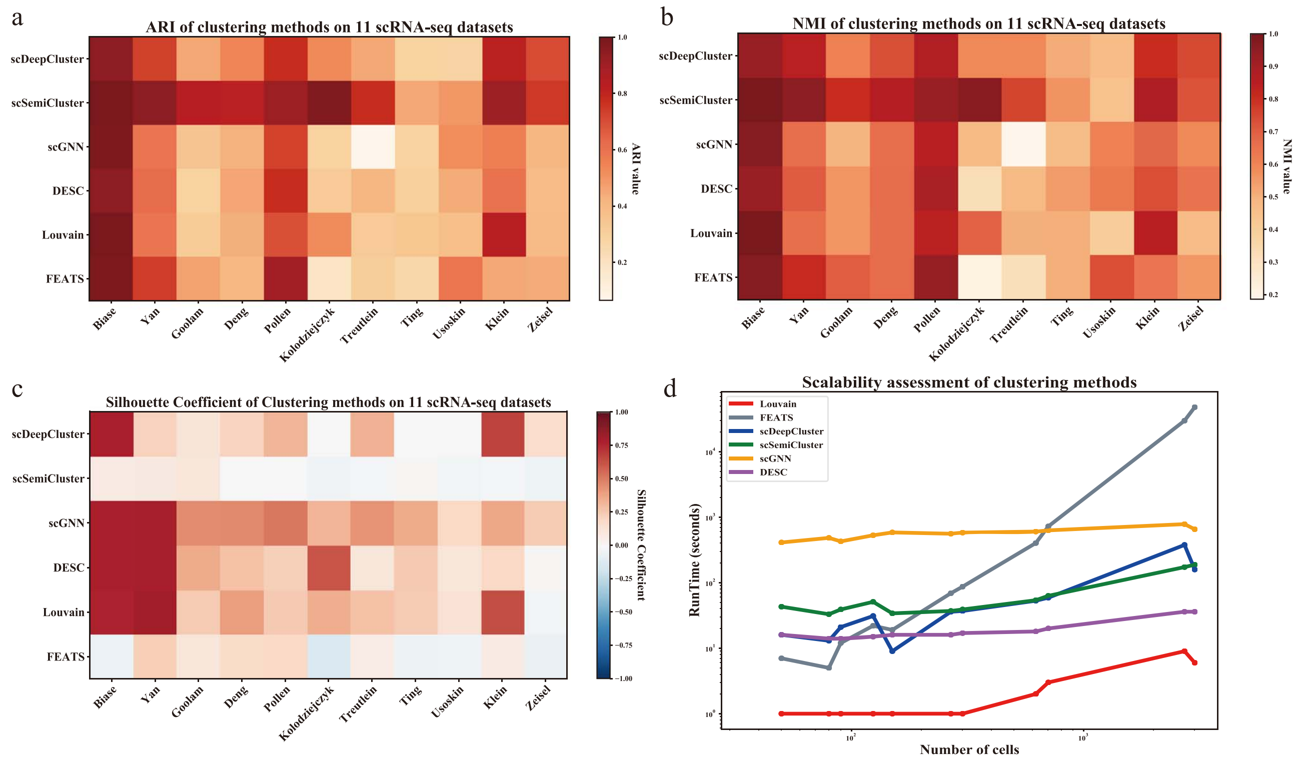

Figure 4. The quantitative assessments of three deep model-based clustering methods scDeepCluster, scSemiCluster, scGNN, DESC and classical clustering algorithm Louvain. The figure shows the clustering results' heatmap of ARI (**a**), NMI (**b**), Silhouette coefficient (**c**), and the tendency of the runtime as the number of cells increases (**d**). The number of cells ranges from 49 to 3005. Note that the range of ARI in this figure is between 0 and 1 as the ARI values of the clustering results of four methods are greater than or equal to 0.

of them increases greatly when the number of cells is greater than 1000. scGNN took hundreds of seconds even when the number of cells is 50, because of its pruning cell graph process and graph autoencoder structure. And the runtime of it grows slowly as the number of cells increases, which suggests that scGNN also has good scalability for those datasets with large number of cells. The runtime of another traditional clustering tool FEATS increases exponentially as the number of cells in the dataset increase. It suggests that FEATS has relatively poor scalability, especially when handling datasets with larger number of cells. In general, scSemiCluster and scDeepCluster are more appropriate for users who have high requirements for clustering performance, while scGNN is suitable for users who consider the stability of clustering algorithm. And users could select DESC when they require both great performance while less runtime. After all, Louvain is recommended to users for initial clustering step before a comprehensive clustering process due to its fast speed.

We further visualized the clustering results of four complex datasets with large number of cells (Kolodziejczyk et al., Klein et al., Usoskin et al. and Zeisel et al.) on UMAPs, as well as the cells with ground truth of these four datasets (Figure 5a-d). It is worth to note that the number of clusters to be clustered of scDeepCluster and scSemiCluster need to be given manually in advance, and scGNN will learn it approximately by applying the k-nearest-neighbors method. scSemiCluster tends to have an excessive tightness and separation of clusters for all four datasets, while it predicts the type of cells accurately. For dataset of Kolodziejczyk (Figure 5a), it can be observed from the original visualization that there exist batch effects in three conditions derived from different experiments. scDeepCluster gets rid of a little batch effect of dataset, while fails to cohere the cells within the clusters and separate the cells between clusters. scGNN divides the dataset into 11 clusters, which suggests it fails to remove the batch effect and may have mistaken batch information for biological differences to distinguish cells. DESC performs better than scGNN as it merges several sub-clusters into a bigger one. However, it also fails to correct the batch effect and divides cells belonging to one cluster into three parts. For dataset of Klein (Figure 5b), the clustering results of scDeepCluster are consistent with ground truth, while scSemiCluster has a better tightness within cluster and an excessive separation between clusters than the ground truth. It is interesting to note scGNN predicts more clusters than the ground truth, which may mean that there are some subtypes of cells within the clusters, and it is worth for researchers to perform a further clustering annotation. For dataset of Usoskin (Figure 5c), scGNN improves the tightness of clusters and illustrates a clear but not excessive separation between clusters than baseline with ground truth, while DESC coheres the cells but makes the relationships between different types of cells disappear. And scDeepCluster fails to tighten the cells that belong to the same cluster. For dataset of Zeisel (Figure 5d), scDeepCluster gives a most consistent clustering result and visualization with the ground truth, and there are even better shapes of some clusters. For several clusters, scGNN divides the cells belonging to one cluster into two parts, although it predicts the same number of clusters as the ground truth. In general, it is almost impossible for a single clustering method to perform excellent on all situations, and each method has its own preferences, which should be considered by users when selecting an appropriate clustering algorithm.

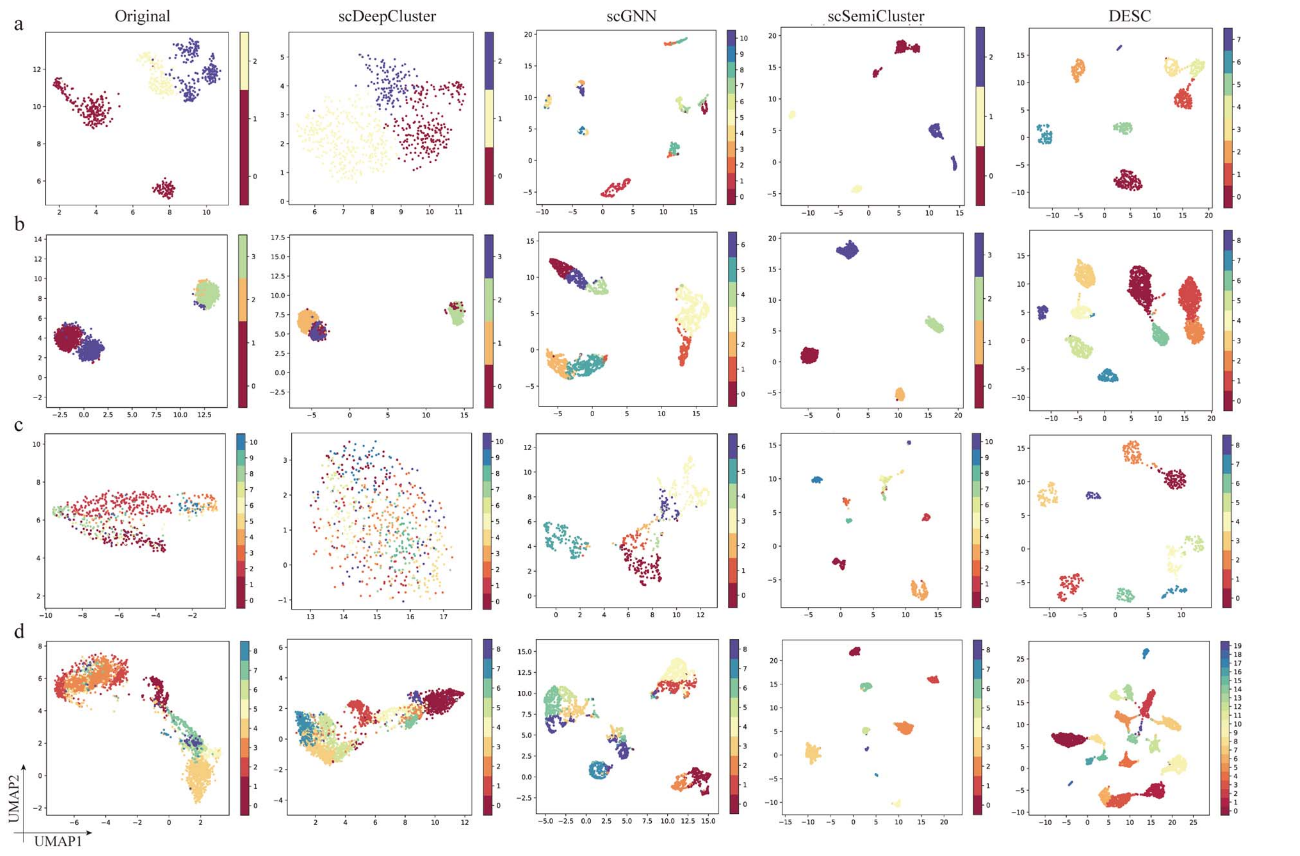

Figure 5. The 2D visualization of UMAPs for clustering results of scDeepCluster, scGNN, scSemiCluster, DESC and cells with ground truth (Original), on datasets of (**a**) Kolodziejczyk et al. (704 cells), (**b**) Klein et al. (2717 cells), (**c**) Usoskin et al. (622 cells) and (**d**) Zeisel et al. (3005 cells), respectively.

Discussion and conclusions

The emergence of scRNA-seq technology has provided deep insights into genomics and transcriptomics studies at the level of individual cells. The availability of amounts of scRNA-seq datasets has facilitated the development of a number of analysis algorithms and tools. However, the high-dimensional, high-throughput and high-noise characteristics of scRNA-seq data remain major challenges for these analysis methods and tools. Deep learning technologies are developed for noisy data with high dimensions, and their models are driven by large amounts of data. These advantages make them match the characteristics of scRNA-seq data very well. In this paper, we reviewed the deep model-based scRNA-seq analysis methods covering the procedure of pre-processing and initial downstream analysis task. The underlying deep models of most scRNA-seq analysis methods are autoencoders, which demonstrates its powerful ability of unsupervised learning and great potential in extracting latent patterns from the original data, especially in dimensionality reduction and visualization task. At the same time, graph neural network has attracted growing attention in recent years. The strengths of its aggregating and propagating mechanisms of graph make it available to capture and learn the heterogeneous cell–cell relationships with a bottom-up pattern.

Moreover, the denoise/imputation methods and clustering methods based on deep learning models were evaluated quantitatively on 11 datasets with gold standard with multi-dimension metrics. The assessment results suggest that neither of these deep model-based scRNA-seq analysis methods is perfect in all situations. Among denoising and imputation methods, scGNN and scIGANs are significantly outperforming other methods in terms of imputation effect and improvements bringing to downstream analysis tasks. However, they also took hundreds of seconds even on the data with smallest number of cells, due to their complex model structures, which contribute to the excellent performance. In contrary, scScope is a fast tool with significant scalability for datasets with large number of cells. But it is not stable enough and performs poorly in terms of Silhouette coefficient. In addition, another important assessment criterion is the availability of methods and tools. However, AutoImpute and GraphSCI perform poorly in this criterion, consuming too much memory to carry out the programs even on a dataset with 50 cells. After all, attention must be paid when performing imputation process as it has been reported to introduce some false signals into the original count data. For clustering methods, scSemiCluster predicts the labels of cells accurately, and performs dominant in terms of ARI and NMI, but fails to cohere cells within clusters and differentiate the cells between clusters. scDeepCluster also has good performance in ARI and NMI, but a poor Silhouette coefficient for datasets with large number of cells. scGNN has a relatively incompetent clustering result, but performs more stable in terms of ARI, NMI and Silhouette coefficient. However, these deep model-based clustering methods are all time assuming, and takes tens or hundreds of times longer than classical clustering algorithm Louvain. DESC is best and linearly scalable and appropriate for large datasets due to its fast speed and fairish clustering performance. In contrary, scDeepCluster and scSemiCluster scale poorly when the number of cells is large. In addition, the number of clusters of scDeepCluster and scSemiCluster to be clustered needs to be manually input in advance as it is unknown in most situations, while scGNN will learn it approximately by applying k-nearest neighbor algorithm. As above deep model-based clustering methods are implemented by Python, scCCESS is a choice for researchers who use R language and corresponding platforms. Besides, users could also choose the appropriate scRNA-seq analysis methods according to the size of the dataset to be analyzed and their hardware conditions.

While the scRNA-seq analysis methods based on deep learning technology continue to emerge, there still remains considerable challenges to this area. As the downstream analysis gets deeper, how could deep model-based scRNA-seq analysis methods maintain their robustness while covering more downstream analysis tasks is a key issue. At the same time, the tools with an acceptable speed and good scalability are expected. Another challenge and expected development are multi-omics integration of scRNA-seq data [116]. As the availability of scRNA-seq data containing genome, transcriptome and proteome, it is not enough for deep model-based analysis methods and tools to use only a single omics data, but rather to integrate multiple omics scRNA-seq data and conduct systematic biological analysis to interpret the complex and dynamic interrelationships across genes and cells.

Key Points

- This paper presented a comprehensive review for deep learning-based single-cell RNA-sequencing (scRNA-seq) analysis approaches, covering processing steps of quality control, normalization, data correction, dimension reduction and visualization, and downstream clustering analysis.
- Detail information of 22 deep model-based scRNA-seq analysis tools were provided, involving their programing languages, functions, underlying deep models and published year.
- We further evaluated deep learning methods for processing step of imputation and clustering analysis that are crucial and most studied on 11 gold standard scRNA-seq datasets, and performed quantitative assessments from multiple prospective. In addition, the descriptions and download URLs of 11 scRNA-seq datasets were provided in this paper.
- According to quantitative assessments, we discussed the advantages, data preferences and limitations of these approaches clearly, and provided recommendations and guidance for non-specialists to select appropriate tools.

Supplementary Data

Supplementary data are available online at *Briefings in Bioinformatics*.

Data availability

This article has not produced any new data. All data used in this paper can be freely downloaded with their corresponding references.

Funding

The National Natural Science Foundation of China (nos 62131004, 61 922 020, 61 771 331); the Sichuan Provincial Science Fund for Distinguished Young Scholars (2021JDJQ0025); the Special Science Foundation of Quzhou (2020D003).

References

1. Tammela T, Sage J. Investigating Tumor Heterogeneity in Mouse Models. *Annu Rev Cancer Biol*, 2020;**4**:99–119.
2. Briggs J, Weinreb C, Wagner DE, *et al*. The dynamics of gene expression in vertebrate embryogenesis at single-cell resolution. *Science* 2018;**360**:980–+.
3. Montoro DT, Haber A, Biton M, *et al*. A revised airway epithelial hierarchy includes CFTR-expressing ionocytes. *Nature* 2018;**560**:319–24.
4. Plasschaert LW, Zilionis R, Choo-Wing R, *et al*. A single cell atlas of the tracheal epithelium reveals the CFTR-rich pulmonary ionocyte. *Nature* 2018;**560**:377–81.
5. Tang F, Barbacioru C, Wang Y, *et al*. mRNA-Seq whole-transcriptome analysis of a single cell. *Nat Methods* 2009;**6**:377–82.
6. Rozenblatt-Rosen O, Stubbington M, Regev A, *et al*. The human cell atlas: from vision to reality. *Nature* 2017;**550**:451–3.
7. Luecken MD, Theis F. Current best practices in single cell RNA seq analysis: a tutorial. *Mol Syst Biol* 2019;**15**.
8. Zhang Z, Cui F, Wang C, *et al*. Goals and approaches for each processing step for single-cell RNA sequencing data. *Brief Bioinform* 2020;**22**(4):bbaa314.
9. Kharchenko P, Silberstein L, Scadden D. Bayesian approach to single-cell differential expression analysis. *Nat Methods* 2014;**11**:740–2.
10. Buettner F, Natarajan K, Casale FP, *et al*. Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. *Nat Biotechnol* 2015;**33**:155–60.
11. Stegle O, Teichmann S, Marioni J. Computational and analytical challenges in single-cell transcriptomics. *Nat Rev Genet* 2015;**16**:133–45.
12. Dillies M, Rau A, Aubert J, *et al*. A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis. *Brief Bioinform* 2013;**14**(6):671–83.
13. Pachter L. Models for transcript quantification from RNA-Seq. *arXiv: Genomics*, 2011;1104.3889.
14. Kiselev V, Andrews T, Hemberg M. Challenges in unsupervised clustering of single-cell RNA-seq data. *Nat Rev Genet* 2019;**20**:273–82.
15. Goodfellow I, Bengio Y, Courville AC. Deep learning. *Nature* 2015;**521**:436–44.
16. Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. *Communications of the ACM* 2012;**60**:84–90.
17. Collobert R, Weston J. A unified architecture for natural language processing: deep neural networks with multitask learning. In: *ICML '08*, 2008;160–67.
18. Alipanahi B, Delong A, Weirauch M, *et al*. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat Biotechnol* 2015;**33**:831–8.
19. Lopez R, Regier J, Cole M, *et al*. Deep generative modeling for single-cell transcriptomics. *Nat Methods* 2018;**15**:1053–8.

20. Amodio M, Dijk DV, Srinivasan K, *et al*. Exploring single-cell data with deep multitasking neural networks. *Nat Methods* 2019;**16**:1139–45.
21. Deng Y, Bao F, Dai Q, *et al*. Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning. *Nat Methods* 2019;**16**:311–4.
22. Zhang Z, Cui F, Lin C, *et al*. Critical downstream analysis steps for single-cell RNA sequencing data. *Brief Bioinform* 2021;bbab105. Epub ahead of print.
23. Bernstein N, Fong NL, Lam I, *et al*. Solo: doublet identification in single-cell RNA-Seq via semi-supervised deep learning. *Cell Systems* 2020;**11**:95–101.e5.
24. Shaham U, Stanton KP, Zhao J, *et al*. Removal of batch effects using distributioŽ matching residual networks. *Bioinformatics* 2017;**33**:253Y 2546.
25. Lotfollahi M, Wolf FA, Theis F. scGen predicts single-cell perturbation responses. *Nat Methods* 2019;**16**:715–21.
26. Wang T, Johnson T, Shao W, *et al*. BERMUDA: a novel deep transfer learning method for single-cell RNA sequencing batch correction reveals hidden high-resolution cellular subtypes. *Genome Biol*. 2019;**20**.
27. Li X, Wang K, Lyu Y, *et al*. Deep learning enables accurate clustering with batch effect removal in single-cell RNA-seq analysis. *Nat Commun*. 2020;**11**.
28. Eraslan G, Simon L, Mircea M, *et al*. Single-cell RNA-seq denoising using a deep count autoencoder. *Nat Commun*. 2019;**10**.
29. Arisdakessian C, Poirion OB, Yunits B, *et al*. DeepImpute: an accurate, fast, and scalable deep neural network method to impute single-cell RNA-seq data. *Genome Biol*. 2019;**20**.
30. Mongia A, Sengupta D, Majumdar A. deepMc: deep matrix completion for imputation of single-cell RNA-seq data. *Journal of Computational Biology: A Journal of Computational Molecular cell Biology* 2019;**27**:1011–1019.
31. Talwar D, Mongia A, Sengupta D, *et al*. AutoImpute: autoencoder based imputation of single-cell RNA-seq data. *Sci Rep* 2018;**8**(1):16329.
32. Xu Y, Zhang Z, You L, *et al*. scIGANs: single-cell RNA-seq imputation using generative adversarial networks. *Nucleic Acids Res* 2020;**48**:e85–5.
33. Rao J, Zhou X, Lu Y, *et al*. Imputing single-cell RNA-seq data by combining graph convolution and autoencoder neural networks. *bioRxiv*. iScience, 2021;**24**(5):102393.
34. Wang J, Ma A, Chang Y, *et al*. scGNN is a novel graph neural network framework for single-cell RNA-Seq analyses. *Nat Commun*. 2021;**12**.
35. Wang D, Gu J. VASC: dimension reduction and visualization of single-cell RNA-seq data by deep Variational autoencoder, genomics. *Proteomics & Bioinformatics* 2018;**16**: 320–31.
36. Lin C, Jain S, Kim H, *et al*. Using neural networks for reducing the dimensions of single-cell RNA-Seq data. *Nucleic Acids Res* 2017;**45**:e156–6.
37. Peng J, Wang X, Shang X. Combining gene ontology with deep neural networks to enhance the clustering of single cell RNA-Seq data. *BMC Bioinformatics* 2019;**20**: 284.
38. Ding J, Condon A, Shah S. Interpretable dimensionality reduction of single cell transcriptome data with deep generative models. *Nat Commun* 2018;**9**:2002.
39. Tian T, Wan J, Song Q, *et al*. Clustering single-cell RNA-seq data with a model-based deep learning approach. *Nature Machine Intelligence* 2019;**1**:191–8.
40. Chen L, He Q, Zhai Y, *et al*. Single-cell RNA-seq data semi-supervised clustering and annotation via structural regularized domain adaptation. *Bioinformatics* 2021;**37**: 775–784.
41. Geddes TA, Kim T, Nan L, *et al*. Autoencoder-based cluster ensembles for single-cell RNA-seq data analysis. *BMC Bioinformatics* 2019;**20**:660.
42. Ziegenhain C, Vieth B, Parekh S, *et al*. Comparative analysis of single-cell RNA sequencing methods. *Mol Cell* 2017;**65**(4):631–643.e634.
43. Macosko EZ, Basu A, Satija R, *et al*. Highly parallel genome-wide expression profiling of individual cells using Nanoliter droplets. *Cell* 2015;**161**:1202–14.
44. Stoeckius M, Zheng S, Houck-Loomis B, *et al*. Cell hashing with barcoded antibodies enables multiplexing and doublet detection for single cell genomics. *Genome Biol* 2018;**19**:224.
45. Ilicic T, Kim JK, Kolodziejczyk AA, *et al*. Classification of low quality cells from single-cell RNA-seq data. *Genome Biol* 2016;**17**(1):29.
46. Griffiths J, Scialdone A, Marioni J. Using single ... cell genomics to understand developmental processes and cell fate decisions. *Mol Syst Biol* 2018;**14**(4):e8046.
47. DePasquale EAK, Schnell DJ, Valiente-Alandí Í, *et al*. DoubletDecon: Cell-state aware removal of single-cell RNA-seq doublets. *Cell rep*, 2019;**29**(6):1718–1727.e8.
48. Wolock SL, Lopez R, Klein AM. Scrublet: computational identification of cell doublets in single-cell transcriptomic data. *Cell systems* 2019;**8**(4):281–291.e289.
49. McGinnis CS, Murrow LM, Gartner Z. DoubletFinder: doublet detection in single-cell RNA sequencing data using artificial nearest Neighbors. *Cell systems* 2019;**8**(4):329–337.e324.
50. Jolliffe IT. Principal Component Analysis. In *Encyclopedia of Statistics in Behavioral Science*. wiley online library, 2002.
51. Golub G, Reinsch C. Singular value decomposition and least squares solutions. *Numerische Mathematik* 2007;**14**:403–20.
52. Hicks SC, Townes W, Teng M, *et al*. Missing data and technical variability in single ... cell RNa sequencing experiments. *Biostatistics* 2018;**19**:56R 578.
53. Mortazavi A, Williams BA, McCue K, *et al*. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat Methods* 2008;**5**:621–8.
54. Lee S, Seo C, Lim B, *et al*. Accurate quantification of transcriptome from RNA-Seq data by effective length normalization. *Nucleic Acids Res* 2011;**39**:e9–9.
55. Li B, Ruotti V, Stewart R, *et al*. RNA-Seq gene expression estimation with read mapping uncertainty. *Bioinformatics* 2010;**26**:493–500.
56. Lun ATL, Bach K, Marioni J. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biol* 2016;**17**:75.
57. Vallejos C, Risso D, Scialdone A, *et al*. Normalizing single-cell RNA sequencing data: challenges and opportunities. *Nat Methods* 2017;**14**:565–71.
58. Haghverdi L, Lun A, Morgan M, *et al*. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat Biotechnol* 2018;**36**:421–7.
59. Butler A, Hoffman PJ, Smibert P, *et al*. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat Biotechnol* 2018;**36**:411–20.
60. Hie B, Bryson B, Berger B. Efficient integration of heterogeneous single-cell transcriptomes using Scanorama. *Nat Biotechnol*, 2019;**37**:685–691.

61. Korsunsky I, Millard N, Fan J, *et al*. Fast, sensitive, and accurate integration of single cell data with harmony. *Nat Methods* 2019;**16**:1289–96.
62. Krzysztof Polański, Matthew D Young, Zhichao Miao, Kerstin B Meyer, Sarah A Teichmann, Jong-Eun Park. BBKNN: fast batch alignment of single cell transcriptomes. *Bioinformatics*, 2020;**36**:964–965.
63. Grün D, Kester L, Oudenaarden A. Validation of noise models for single-cell transcriptomics. *Nat Methods* 2014;**11**: 637–40.
64. Xie S, Girshick RB, Dollár P *et al*. Aggregated residual transformations for deep neural networks, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE: NEW YORK, NY 10017 USA, 2017:5987–95.
65. Blondel V, Guillaume J-L, Lambiotte R, *et al*. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008;**2008**:10008.
66. Bacher R, Kendziorski C. Design and computational analysis of single-cell RNA-sequencing experiments. *Genome Biol* 2016;**17**:63.
67. Kim H, Golub G, Park H. Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics* 2005;**21**(2):187–98.
68. Prabhakaran S, Azizi E, Carr A, *et al*. Dirichlet process mixture model for correcting technical variation in single-cell gene expression data. *JMLR Workshop and Conference Proceedings* 2016;**48**:1070–9.
69. Dijk DV, Sharma R, Nainys J, *et al*. Recovering gene interactions from single-cell data using data diffusion. *Cell* 2018;**174**:716–729.e727.
70. Li W, Li J. An accurate and robust imputation method scImpute for single-cell RNA-seq data. *Nat Commun*. 2018;**9**.
71. Gong W, Kwak I-Y, Pota P, *et al*. DrImpute: imputing dropout events in single cell RNA sequencing data. *BMC Bioinformatics* 2018;**19**(1):220.
72. Huang M, Wang J, Torre EA, *et al*. SAVER: gene expression recovery for single-cell RNA sequencing. *Nat Methods* 2018;**15**:539–42.
73. Goodfellow I, Pouget-Abadie J, Mirza M, *et al*. *Generative Adversarial Nets*. Montreal, CANADA: NIPS, 2014;**27**: 2672–2680.
74. Potamias M, Bonchi F, Gionis A, *et al*. K-nearest neighbors in uncertain graphs. *Proceedings of the VLDB Endowment* 2010;**3**:997–1008.
75. Kipf T, Welling M. Semi-supervised classification with graph convolutional networks. *ArXiv*, 2017, abs/1609.02907.
76. Andrews T, Hemberg M. False signals induced by single-cell imputation. *F1000Research* 2018;**7**:1740.
77. Bellman R. Dynamic programming. *Science* 1966; **153**(3731):34–7.
78. Pierson E, Yau C. ZIFA: dimensionality reduction for zero-inflated single-cell gene expression analysis. *Genome Biol*. 2015;**16**.
79. Brennecke P, Anders S, Kim JK, *et al*. Accounting for technical noise in single-cell RNA-seq experiments. *Nat Methods* 2013;**10**:1093–5.
80. Heimberg G, Bhatnagar R, El-Samad H, *et al*. Low dimensionality in gene expression data enables the accurate extraction of transcriptional programs from shallow sequencing. *Cell systems* 2016;**2**(4):239–50.
81. Coifman R, Lafon S, Lee AB, *et al*. Geometric diffusions as a tool for harmonic analysis and structure definition of data: diffusion maps. *Proc Natl Acad Sci U S A* 2005; **102**(21):7426–31.
82. Maaten LVD, Hinton GE. Visualizing data using t-SNE. *Journal of Machine Learning Research* 2008;**9**:2579–605.
83. McInnes L, Healy J, Saul N, *et al*. UMAP: uniform manifold approximation and projection. *J Open Source Softw* 2018;**3**:861.
84. Jang E, Gu S, Poole B. Categorical Reparameterization with Gumbel-Softmax. *ArXiv*, 2017;abs/1611.01144.
85. Jain A. Data clustering: 50 years beyond K-means. *Pattern Recognit Lett* 2010;**31**:651–66.
86. MacQueen J. *Some methods for classification and analysis of multivariate observations*, 1967;**1**:281–297.
87. Lloyd SP. Least squares quantization in PCM. *IEEE Trans Inf Theory* 1982;**28**:129–36.
88. Grün D, Lyubimova A, Kester L, *et al*. Single-cell messenger RNA sequencing reveals rare intestinal cell types. *Nature* 2015;**525**:251–5.
89. Wang B, Zhu J-J, Pierson E, *et al*. Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning. *Nat Methods* 2017;**14**:414–6.
90. Lin P, Troup M, Ho J. CIDR: ultrafast and accurate clustering through imputation for single-cell RNA-seq data. *Genome Biol* 2017;**18**(1):59.
91. Zeisel A, Muñoz-Manchado A, Codeluppi S, *et al*. Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. *Science* 2015;**347**:1138–42.
92. Zurauskiene J, Yau C. pcaReduce: hierarchical clustering of single cell transcriptional profiles. *BMC Bioinformatics* 2016;**17**(1):140.
93. Tasic B, Menon V, Nguyen T, *et al*. Adult mouse cortical cell taxonomy by single cell Transcriptomics. *Nat Neurosci* 2016;**19**:335–46.
94. Argelaguet R, Velten B, Arnol D, *et al*. Multi omics factor Analysi a framework for unsupervised integration of multi omics data sets. *Mol Syst Biol* 2018;**14**(6):e8124.
95. Kiselev V, Kirschner K, Schaub MT, *et al*. SC3-consensus clustering of single-cell RNA-Seq data. *Nat Methods* 2017;**14**:483–6.
96. Yang Y, Huh R, Culpepper HW, *et al*. SAFe clustering: single cell aggregated (from ensemble) clustering for single cell RNA seq data. *Bioinformatics* 2019;**35**:126Y 1277.
97. Freytag S, Tian L, Lönnstedt I, *et al*. Comparison of clustering tools in R for medium-sized 10x genomics single-cell RNA-sequencing data. *F1000Research* 2018;**7**: 1297.
98. Qi R, Ma A, Ma Q, *et al*. Clustering and classification methods for single-cell RNA-sequencing data. *Brief Bioinform* 2020;**21**(4):1196–208.
99. Petegrosso R, Li Z, Kuang R. Machine learning and statistical methods for clustering single-cell RNA-sequencing data. *Brief Bioinform* 2020;**21**(4):1209–23.
100. Biase F, Cao X, Zhong S. Cell fate inclination within 2-cell and 4-cell mouse embryos revealed by single-cell RNA sequencing. *Genome Res* 2014;**24**(11): 1787–96.
101. Yan L, Yang M, Guo H, *et al*. Single-cell RNA-Seq profiling of human preimplantation embryos and embryonic stem cells. *Nat Struct Mol Biol* 2013;**20**:1131–9.
102. Goolam M, Scialdone A, Graham SJL, *et al*. Heterogeneity in Oct4 and Sox2 targets biases cell fate in 4-cell mouse embryos. *Cell* 2016;**165**:61–74.
103. Deng Q, Ramsköld D, Reinius B, *et al*. Single-cell RNA-Seq reveals dynamic, random Monoallelic gene expression in mammalian cells. *Science* 2014;**343**: 193–6.

104. Pollen AA, Nowakowski T, Shuga J, *et al*. Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. *Nat Biotechnol* 2014;**32**:1053–8.
105. Kolodziejczyk AA, Kim JK, Tsang JC, *et al*. Single cell RNA-sequencing of pluripotent states unlocks modular transcriptional variation. *Cell Stem Cell* 2015;**17**:471–85.
106. Treutlein B, Brownfield D, Wu A, *et al*. Reconstructing lineage hierarchies of the distal lung epithelium using single cell RNA-seq. *Nature* 2014;**509**:371–5.
107. Ting D, Wittner B, Ligorio M, *et al*. Single-cell RNA sequencing identifies extracellular matrix gene expression by pancreatic circulating tumor cells. *Cell Rep* 2014;**8**: 1905–18.
108. Usoskin D, Furlan A, Islam S, *et al*. Unbiased classification of sensory neuron types by large-scale single-cell RNA sequencing. *Nat Neurosci* 2015;**18**:145–53.
109. Klein AM, Mazutis L, Akartuna I, *et al*. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell* 2015;**161**:1187–201.
110. Wolf FA, Angerer P, Theis F. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol*. 2017;**19**.
111. Zappia L, Phipson B, Oshlack A. Splatter: simulation of single-cell RNA sequencing data. *Genome Biol* 2017; **18**(1):174.
112. Hubert L, Arabie P. Comparing partitions. *Journal of Classification* 1985;**2**:193–218.
113. Strehl A, Ghosh J. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J Mach Learn Res* 2002;**3**:583–617.
114. Rousseeuw P. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* 1987;**20**: 53–65.
115. Vans E, Patil A, Sharma A. FEATS: feature selection-based clustering of single-cell RNA-seq data. *Brief Bioinform* 2020;**22**:bbaa306.
116. Ma A, McDermaid A, Xu J, *et al*. Integrative methods and practical challenges for single-cell multi-omics. *Trends Biotechnol* 2020;**38**(9):1007–22.