Bioinformatics, 38(13), 2022, 3377–3384
https://doi.org/10.1093/bioinformatics/btac357
Advance Access Publication Date: 26 May 2022
Original Paper

Gene expression

scPreGAN, a deep generative model for predicting the response of single-cell expression to perturbation

Xiajie Wei[1,2,†], Jiayi Dong[1,2,†] and Fei Wang [1,2,*]

[1]Shanghai Key Lab of Intelligent Information Processing, Shanghai, China and [2]School of Computer Science and Technology, Fudan University, Shanghai, China

*To whom correspondence should be addressed.

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.

Associate Editor: Peter Robinson

Received on September 14, 2021; revised on April 29, 2022; editorial decision on May 18, 2022; accepted on May 20, 2022

Abstract

Motivation: Rapid developments of single-cell RNA sequencing technologies allow study of responses to external perturbations at individual cell level. However, in many cases, it is hard to collect the perturbed cells, such as knowing the response of a cell type to the drug before actual medication to a patient. Prediction in silicon could alleviate the problem and save cost. Although several tools have been developed, their prediction accuracy leaves much room for improvement.

Results: In this article, we propose scPreGAN (Single-Cell data Prediction base on GAN), a deep generative model for predicting the response of single-cell expression to perturbation. ScPreGAN integrates autoencoder and generative adversarial network, the former is to extract common information of the unperturbed data and the perturbed data, the latter is to predict the perturbed data. Experiments on three real datasets show that scPreGAN outperforms three state-of-the-art methods, which can capture the complicated distribution of cell expression and generate the prediction data with the same expression abundance as the real data.

Availability and implementation: The implementation of scPreGAN is available via https://github.com/JaneJiayiDong/scPreGAN. To reproduce the results of this article, please visit https://github.com/JaneJiayiDong/scPreGAN-reproducibility.

Contact: wangfei@fudan.edu.cn

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Rapid developments of single-cell RNA sequencing (scRNA-seq) technologies (Metzker, 2010; Nawy, 2014; Shapiro *et al.*, 2013) allow the analysis of transcriptional data at the individual cell level and provide more information about the heterogeneity of tissues and organs than the bulk RNA sequencing technologies (Papalexi and Satija, 2018; Patel *et al.*, 2014). With the increasing amount of scRNA-seq data, researchers could analyze cellular responses to external perturbations (Lotfollahi *et al.*, 2019b; Schubert *et al.*, 2018), which greatly promote research in personalized medication, such as cancer. Cancer heterogeneity is so high that targeted drugs are widely used.

While it remains challenging to obtain data on cellular responses to different perturbations, e.g. it is almost impossible to evaluate therapy response of each cancer subtype in wet-lab experiments. One reason is that the space of all possible treatments and their combination is super huge, so it is almost impossible to exhaustively explore the responses to perturbations for each cell type. Another reason is that scRNA-seq data in certain conditions sometimes are difficult to be collected (Rampášek *et al.*, 2019), which limits the sample size and reduces the credibility of the biological results. Besides, wet experiments are cost-intensive. Predicting cellular perturbation responses in silicon could alleviate these problems, which not only save cost and labor but also avoid the difficulties that biological experiments meet. What is more important, it can be used to analyze drug effects (Karczewski and Snyder, 2018), therapeutic efficacy (Hagai *et al.*, 2018), gene knockdown (Datlinger *et al.*, 2017) and to guide personalized treatment (Rampášek *et al.*, 2019).

In recent years, deep-learning methods especially generative models have developed rapidly in various fields. Though these models, such as the variational autoencoders (Kingma and Welling, 2013) and generative adversarial networks (GANs) (Goodfellow *et al.*, 2014), have achieved great success in the field of computer vision like predicting images related to specific conditions, there are still many challenges in predicting perturbation response of scRNA-seq data. One reason is that scRNA-seq data are sparse and suffered from dropout events, i.e. the real non-expressed genes and

the non-sequenced parts cannot be distinguished. In addition, there is a lot of technical noise in scRNA-seq data. These features make the distribution of scRNA-seq data hard modeled (Korthauer *et al.*, 2016).

At present, generative models for perturbation-response prediction of scRNA-seq data can be roughly divided into three categories. The first category is based on vector arithmetic, such as scGen (Lotfollahi *et al.*, 2019b), Tybalt (Way and Greene, 2018), CPA (Lotfollahi *et al.*, 2021), etc. These methods are essentially hard-coded, which means that the responses to perturbations of all cell types are assumed to be the same. This kind of methods cannot distinguish the different responses to perturbations among several cell types and lead to deviation of prediction.

The second category relies on the conditional variables of a conditional variational autoencoder (CVAE) (Sohn *et al.*, 2015), such as trVAE (Lotfollahi *et al.*, 2019a) and scAlign (Johansen and Quon, 2019). The inputs of CVAE contain condition vectors related to the conditional information. These methods simulate perturbation responses by changing the condition vectors. Whether the conditional vectors can accurately capture the responses of different cell types to perturbations depends on the way of how to regularize them.

The third category switches to adversarial training, which promotes generator networks to generate the prediction data, with the stronger fitting ability (Russkikh *et al.*, 2020; Targonski *et al.*, 2020). The performance of this category relies on how to apply adversarial training. For instance, stVAE (Russkikh *et al.*, 2020) places a discriminator on the latent space and trains it against the encoder to make part of latent vectors disentangled with specific perturbations. The rest part of the latent vectors can be altered to generate the desired data. This category of methods can make the generated data distribution conform to the real data distribution; therefore, they are more promising in perturbation-response prediction compared with vector arithmetic or CVAE-based methods. However, stVAE applies the adversarial network to the latent vectors instead of the generated data. Although latent vectors can extract information unrelated to the perturbations, they cannot guarantee that the data generated by the generator fit the real data distribution. TSPG (Targonski *et al.*, 2020) is another instance of this category. Though the discriminator of TSPG is trained against the generator, outputs of the generator are the predicted vector difference between the unperturbed data and the perturbed data, which are not the direct inputs of the discriminator. It may result in limited gradient information provided by the discriminator, consequently the model training is difficult to converge.

In this article, we propose scPreGAN (Single-Cell data Prediction base on GAN), a generative model that designs for predicting the perturbation response of scRNA-seq data. ScPreGAN uses one encoder to obtain latent vectors as the common information of the perturbed and unperturbed data, and two generators to fit the data distribution under each condition, respectively. From the common latent vectors, one generator reconstructs the unperturbed data, and the other predicts the corresponding perturbed data. The predicted data we want are the output of neural networks instead of relying on conditional variables associated with certain perturbations. This end-to-end prediction way makes the transformation ability of scPreGAN stronger. Besides, we add two discriminators to the generated data space and let them train against the generators, which makes the generated distribution more realistic. What is more, we choose an appropriate loss function to make scPreGAN trained efficiently and stably. We have tested scPreGAN on three real datasets and compared it with three state-of-the-art methods. The experimental results show that scPreGAN performs much better than other methods.

2 Materials and methods

2.1 GAN with autoencoder

In the field of deep learning, generative models are usually used for data simulation or data augmentation (Antoniou *et al.*, 2017; Calimeri *et al.*, 2017; Marouf *et al.*, 2020). One of the most widely used generative model is GAN. GAN is composed of two neural networks: a generator network simulates the real data distribution and generates data from random noise; a discriminator network tries to distinguish the generated data from real data. The training goal of the generator is to confuse the discriminator and to make the discriminator believe that its outputs are real data. The generator and the discriminator are trained against each other. When reaching the Nash balance, the generated data would look real. Although the trained generator holds a strong generating ability, inputs of the generator are usually Gaussian random noise with a lot of uncertainty, which leads to the lack of controllability of the generated data. Later proposed conditional GANs (CGANs) (Mirza and Osindero, 2014) can generate data matching a specific condition, and the premise is that the data of the specific condition must exist in the training set. Auxiliary classifier GAN (Odena *et al.*, 2017) has an improved version of the discriminator, which can distinguish between different classes.

Autoencoder (AE) is a type of artificial neural network used to learn efficient codings of data. It consists of two neural networks, namely an encoder and a decoder. The encoder maps the input data to low-dimensional embeddings. The decoder receives the embeddings and reconstructs the high-dimensional data. The low-dimensional embeddings given by the trained encoder maximize the information of real data. Therefore, AEs are mainly used for dimensionality reduction. We can input the low-dimensional embeddings of AE to the generator of GAN, so that the generator can receive useful information to generate data that match specific conditions. Many models combining AEs and GANs have been developed in the computer vision field (Dai Yang *et al.*, 2021; Makhzani *et al.*, 2015; Zhou *et al.*, 2019). For example, BranchGAN (Zhou *et al.*, 2019) with a single encoder and dual decoder structure was used for image style transfer. Inspired by this idea, we propose scPreGAN for predicting perturbation response of scRNA-seq data, in which special treatments are carried out based on characteristics of scRNA-seq data.

2.2 Overview of scPreGAN

The exact description of cellular perturbation-response prediction is as follows: for the scRNA-seq data of a cell type i with only the unperturbed condition (control group), we would like to use the available unperturbed and perturbed scRNA-seq data of other cell types to predict the perturbed scRNA-seq data of cell type i (case group).

In this article, the real perturbed data means the real data sequenced from cells after perturbation, the real unperturbed data means the real data sequenced from cells without perturbation, the predicted perturbed data means the prediction from an algorithm.

The architecture of scPreGAN is shown in Figure 1. The model consists of an encoder network E, two generator networks G_1 and G_2 and two discriminator networks D_1 and D_2. E maps the input of both the unperturbed data X_1 and the perturbed data X_2 to a common low-dimensional latent space. The goal of E is to extract the common features of both X_1 and X_2, i.e. the latent vector z contains common information of the data in two conditions. G_1 and G_2 generate data corresponding to the unperturbed and the perturbed conditions respectively from the latent vector z. G_1 and D_1, G_2 and D_2 form two paired GANs. We input the data generated by G_1 together with X_1 into D_1. The training goal of D_1 is to distinguish real data from the generated data correctly, and the training goal of G_1 is to fool D_1. On the other hand, multi-task learning is used in D_1, i.e. D_1 needs to cope with discriminating and classification tasks. D_1 and G_1 hope to improve the accuracy of classification tasks. The same training scheme is applied to G_2 and D_2. The five neural networks are trained together at the same time. We can obtain the reconstructed data from G_1 and the transformed prediction data from G_2 after training finished. In a word, the first half of scPreGAN is based on AE architecture, and the second half is based on GAN architecture. For perturbation-response prediction, we use E to encode the unperturbed data into latent space and then send the latent vectors into G_2. The outputs of G_2 are the prediction data.

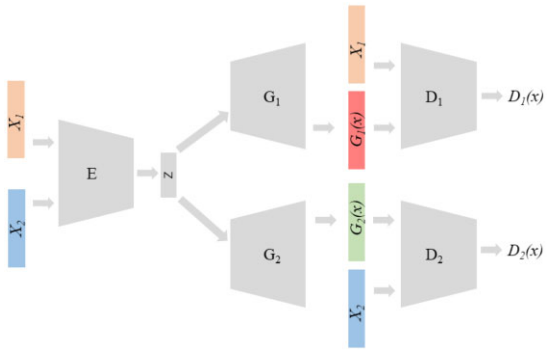

Fig. 1. Overview of scPreGAN. The model consists of an encoder network E, two generator networks G_1 and G_2 and two discriminator networks D_1 and D_2. E maps the input containing both the unperturbed data X_1 and the perturbed data X_2 to a common low-dimensional latent space. G_1 generates the unperturbed data $G_1(x)$ and G_2 generates the perturbed data $G_2(x)$ both from the latent vector z. G_1 and D_1, G_2 and D_2 form two paired GANs. The output of $D_1(D_2)$ is $D_1(x)(D_2(x))$ including the probability of whether x is real $P(s|x)$ and the probability that x belongs to different classes $P(c|x)$. For perturbation-response prediction, we use E to encode X_1 into latent space and then send the latent vector z into G_2. The outputs $G_2(x)$ are the prediction data

2.3 Loss function

The loss function of scPreGAN is designed according to the characteristics of scRNA-seq data. We consider the information hidden in scRNA-seq data from three aspects: (i) *condition information*, which mainly refers to whether the data are currently perturbed; (ii) *high-level abstract information*, mainly the biological information of the data including which class the data belong to; and (iii) *individual cell information*, mainly the detailed information of an individual cell in the data. Accordingly, adversarial loss, encoding loss and reconstruction loss are used to constrain scPreGAN to learn these three types of information. They play different roles in model training. We will explain them in detail as follows.

2.3.1 Adversarial loss

Adversarial loss is the loss function of the GAN architecture part in scPreGAN. Two generators G_1 and G_2 need to fit the distribution of the unperturbed and the perturbed data and capture their features respectively. These features are the condition information mentioned earlier. Two discriminators D_1 and D_2 need to distinguish between the real data and the generated data as well as among different cell types. Therefore, the objective function has two parts: the log-likelihood of the correct source, $L_S = \mathbb{E}[\log P(S|X)]$ and the log-likelihood of the correct class $L_C = \mathbb{E}[\log P(C|X)]$ according to the two parts of output $P(S|X)$ and $P(C|X)$. Specifically, the loss function of D_1 is defined in Equation (1)

$$\begin{aligned} L_{D_1} = {} & \mathbb{E}[\log P(S = real|X_1)] \\ & + \mathbb{E}[\log P(C = c|X_1)] \\ & + \mathbb{E}[\log P(S = fake|(G_1(E(X_2))))] \\ & + \mathbb{E}[\log P(C = c|(G_1(E(X_2))))], \end{aligned} \tag{1}$$

where X_1 and X_2 represent the distribution of unperturbed data and perturbed data, respectively. $E(x)$ means outputs of E and $G_1(x)$ means outputs of G_1.

The first and the second terms of L_{D_1} calculate the log-likelihood of the outputs obtained by inputting the unperturbed data into D_1 while the third and the last terms of L_{D_1} are the results obtained by inputting the data regenerated by G_1 from the encoded perturbed data into D_1. The optimization objective of D_1 is to maximize L_{D_1}. As L_{D_1} increases, D_1 has a stronger ability to distinguish the real data from the generated data under the condition of correct class. The loss of other one shown in Equation (2) is similar. Of note, the strategy of using the data regenerated by G_2 from the encoded unperturbed data can provide all types for classification task in the 'out-of-sample prediction' (Lotfollahi *et al.*, 2019b) (more details in Section 2.4).

$$\begin{aligned} L_{D_2} = {} & \mathbb{E}[\log P(S = real|X_2)] \\ & + \mathbb{E}[\log P(C = c|X_2)] \\ & + \mathbb{E}[\log P(S = fake|(G_2(E(X_1))))] \\ & + \mathbb{E}[\log P(C = c|(G_2(E(X_1))))]. \end{aligned} \tag{2}$$

What is more, G_1 is trained against D_1 with the L_S part of adversarial loss shown in Equation (1) and at the same time aims to improve the L_C part. The optimization objective of G_1 is to maximize L_{adv_1}. We can divide it into three different pairs of $L_C - L_S$. The first and second pairs in Equation (3) have the same meaning as in Equation (1). These two pairs can prompt G_1 to generate more realistic data. The third pair is to encode and decode the unperturbed data twice and then to input them into D_1, which can make the training of the entire model more stable through using two different decoders.

$$\begin{aligned} L_{adv_1} = {} & -\mathbb{E}[\log P(S = fake|(G_1(E(X_1))))] \\ & + \mathbb{E}[\log P(C = c|(G_1(E(X_1))))] \\ & - \mathbb{E}[\log P(S = fake|(G_1(E(X_2))))] \\ & + \mathbb{E}[\log P(C = c|(G_1(E(X_2))))] \\ & - \mathbb{E}[\log P(S = fake|(G_1(E(G_2(E(X_1))))))] \\ & + \mathbb{E}[\log P(C = c|(G_1(E(G_2(E(X_1))))))]. \end{aligned} \tag{3}$$

Correspondingly, the adversarial losses of G_2 are shown in Equation (4).

$$\begin{aligned} L_{adv_2} = {} & -\mathbb{E}[\log P(S = fake|(G_2(E(X_2))))] \\ & + \mathbb{E}[\log P(C = c|(G_2(E(X_2))))] \\ & - \mathbb{E}[\log P(S = fake|(G_2(E(X_1))))] \\ & + \mathbb{E}[\log P(C = c|(G_2(E(X_1))))] \\ & - \mathbb{E}[\log P(S = fake|(G_2(E(G_1(E(X_2))))))] \\ & + \mathbb{E}[\log P(C = c|(G_2(E(G_1(E(X_2))))))]. \end{aligned} \tag{4}$$

2.3.2 Encoding loss

The encoder maps the real data from high-dimensional space to low-dimensional space. We can consider it as a kind of information compression. The latent vectors of low-dimensional space retain the most important features of the real data. We describe the information contained in latent vectors as high-level abstract information. In our model, the encoder extracts the common features of both the unperturbed data and the perturbed data. In an ideal situation, the difference between the unperturbed data and the perturbed data only comes from external perturbation, we want the latent vectors to extract common high-level biological information excluding the perturbation element. In scPreGAN, encoding the unperturbed data and then using G_2 to decode the latent vectors will get the predicted perturbed data from the unperturbed data. In such case, the predicted perturbed data should switch perturbation information and retain the high-level abstract information of the original unperturbed data. If the reconstructed perturbed data are sent to the encoder, the resulting latent vectors should be the same as the latent vectors of the original perturbed data. Based on this assumption, we define the encoding loss of G_2 as shown in Equation (5).

$$\begin{aligned} L_{enc_2} = {} & \mathbb{E}_{x_2 \sim X_2}[||E(G_2(E(x_2))) - E(x_2)||^2] \\ & + \mathbb{E}_{x_1 \sim X_1}[||E(G_2(E(x_1))) - E(x_1)||^2]. \end{aligned} \tag{5}$$

In Equation (5), the first term is the mean square error of $E(G_2(E(x_2)))$ and $E(x_2)$. The second term is the mean square error of $E(G_2(E(x_1)))$ and $E(x_1)$. Under the constraints of these two terms, data generated by G_2 remain high-level abstract information unchanged. Similarly, the encoding loss of G_1 is shown in Equation (6).

$$L_{enc_1} = \mathbb{E}_{x_1 \sim X_1}[||E(G_1(E(x_1))) - E(x_1)||^2] + \mathbb{E}_{x_2 \sim X_2}[||E(G_1(E(x_2))) - E(x_2)||^2]. \quad (6)$$

2.3.3 Reconstruction loss

ScRNA-seq data contain unique information about cells. In order to promote G_1 and G_2 to better fit the data distribution and imitate the way how the specific information is generated, we design reconstruction loss to constrain the encoder and the generator. The reconstruction loss of G_1 shown in Equation (7) is used to constrain $G_1(E(x))$ generated by G_1 to be as close to the original data x as possible.

$$L_{rec_1} = \mathbb{E}_{x \sim X_1}[||G_1(E(x)) - x||^2], \quad (7)$$

in which, we use the mean square error as the reconstruction loss. Similarly, the reconstruction loss of G_2 is shown in Equation (8).

$$L_{rec_2} = \mathbb{E}_{x \sim X_2}[||G_2(E(x)) - x||^2]. \quad (8)$$

2.3.4 Optimization objectives

By incorporating Equation (1) ~ Equation (8), the optimization objective of the discriminators is to minimize Equation (9) and the optimization objective of the encoder and generators is to minimize Equation (10).

$$L_D = -L_{D_1} - L_{D_2}, \quad (9)$$

$$L_G = -\lambda_1(L_{adv_1} + L_{adv_2}) + \lambda_2(L_{enc_1} + L_{enc_2}) + \lambda_3(L_{rec_1} + L_{rec_2}). \quad (10)$$

2.4 Model training

Training scPreGAN includes the following steps: (i) *choosing model architecture and hyperparameters*. All neural networks are composed of two fully connected hidden layers, and the dimension of latent space of the encoder is 16. Hidden layers of the encoder and the generators apply layer normalization, and hidden layers of the discriminators apply spectral normalization (Miyato *et al.*, 2018), which can improve the stability of model training. The Adam optimizer is used to perform backpropagation. We tune the model in a random search way with the peripheral blood mononuclear cell (PBMC) dataset (see Section 3.2 for details). The architecture of scPreGAN is listed in Supplementary Table S1. The values of other hyperparameters can be found in Supplementary Table S2. (ii) *Training models*. To evaluate the generality of scPreGAN, we train models separately for all cell types in three datasets. Before training the model for a cell type, the perturbed data of this cell type are removed from the training data. The remaining unperturbed data of this cell type as well as the unperturbed and perturbed data of other cell types are used for training. In other words, the real perturbed data to be predicted are excluded from the training data, such a training scheme is also called 'out-of-sample prediction' (Lotfollahi *et al.*, 2019b). (iii) *Predicting perturbed data*. After training the model, we input the real unperturbed data of the cell type we want to predict to the encoder and then use G_2 to decode the latent vectors. The outputs of G_2 are the predicted perturbed data.

3 Baseline methods and datasets

3.1 Baseline methods

We choose three alternative methods, scGen, trVAE and stVAE, to compare with scPreGAN. These three methods are typical representative of vector arithmetic, CVAE and adversarial training, respectively. For each method, we use their default parameters to train the model and perform out-of-sample prediction. The detailed information about these methods is described as follows.

3.1.1 scGen

We install the python package of scGen from https://github.com/theislab/scgen and conduct experiments according to the examples provided in the tutorial. The input dimension is the number of genes in the training set. Model architecture and other hyperparameters apply the default settings provided by the author.

3.1.2 stVAE

We download stVAE's python package from https://github.com/NRsh-ka/stvae. The input dimension is the number of genes in the expression data, and other hyperparameters are set to the default values. The activation of stVAE is Mish and the outputs contain negative numbers. According to the author's prompt (Russkikh *et al.*, 2020), we replace all negative values in the outputs with 0.

3.1.3 trVAE

We install trVAE's python package according to the code address (https://github.com/theislab/trVAE) and configure the relevant experimental environment as required. According to the source code provided by the author, we set trVAE to accept 2000 highly variable genes. For data with more than 2000 genes, we use Scanpy (Wolf *et al.*, 2018) to select the top 2000 highly variable genes as the inputs of the model. The dimension of the bottleneck layer is 40 and the activation is ReLU. The maximum number of iterations is 1000 with the setting of early stopping. The rest of the hyperparameters use the default values of trVAE.

3.2 Datasets

Three real datasets with different characteristics are used for evaluation. For each dataset, we separately train a model for each cell type to perform out-of-sample prediction and finally combine the prediction data of all cell types to obtain the whole prediction data, which are used for further analysis. The specific information of these datasets is described below.

3.2.1 The PBMC dataset

The PBMC dataset (Haber *et al.*, 2017) includes seven cell types of control and interferon-beta (IFN-β)-stimulated human PBMCs. Original data can be obtained with accession number GSE96583. We got the preprocessed data from https://github.com/theislab/scgen-reproducibility/blob/master/code/DataDownloader.py (Lotfollahi *et al.*, 2019b) and retained the expression data with 2000 highly variable genes and 18 868 cells. The features of this dataset are that the number of samples is moderate, and the distribution is different between the two conditions.

3.2.2 The H.poly dataset

The H.poly dataset (Kang *et al.*, 2018) (accession number GSE92332) includes two conditions: 2711 parasitic helminth H.poly infected intestinal epithelial cells and 3240 control intestinal epithelial cells. Each condition includes eight cell types with 2000 highly variable genes. Original data were prepared as described in scGen paper (Lotfollahi *et al.*, 2019b). We downloaded it from https://github.com/theislab/scgen-reproducibility/blob/master/code/DataDownloader.py. This dataset has a small number of samples, and the distribution difference between the two groups of cells is obvious. It is mainly used to evaluate the performance of the models when the training sets are small.

3.2.3 The LPS dataset

The LPS dataset (Hagai *et al.*, 2018) (accession ID E-MTAB-6754) contains expression data on 6619 genes of four species (mouse, rat, rabbit and pig) with a total of 77 642 cells. The annotated dataset was taken from https://github.com/theislab/scgen-reproducibility/blob/master/code/DataDownloader.py. The data include two groups: control phagocytes and phagocytes perturbed by LPS for 6 h. For more preprocessing information about this dataset, please refer to scGen paper (Lotfollahi *et al.*, 2019b). The characteristic of

this dataset is that it contains different species, and the distribution between two groups of each species is quite different. It is mainly used to explore the performance of the models when predicting single-cell perturbation across species. In addition, we selected 2000 highly variable genes consistent with the above.

The details of these three datasets are listed in Supplementary Tables S3–S5.

4 Results

4.1 scPreGAN successfully predicts the perturbed data

In order to confirm whether the perturbed data predicted by scPreGAN conform to the distribution of real perturbed data, we combine the predicted perturbed data, the real unperturbed data and the real perturbed data to perform Uniform Manifold Approximation and Projection (UMAP) (McInnes *et al.*, 2018) dimensionality reduction, shown in Figure 2.

From Figure 2a that is results of the PBMC dataset, we can figure out that: for scPreGAN, the predicted data effectively being close to the real perturbed data; for scGen, the predicted data approximately overlap with the unperturbed data not the perturbed data, and it looks that the prediction task seemly is incomplete; for stVAE, its predicted data are quite different from the real data, which overlap neither with the real unperturbed nor with the perturbed data, indicating that the authenticity of the predicted data is insufficient; and for trVAE, the prediction overlaps the real unperturbed data much more than the real perturbed data.

For the H.poly dataset shown in Figure 2b, we observe similar results to the PBMC dataset, and scPreGAN gives the best prediction, which matches the target well. Since scGen predicts perturbation response following a hard-code way, the results reflect its problem of insufficient fitting ability. StVAE puts the discriminator in the latent space instead of the high-dimensional space, which can promote the model to get disentangled representations in latent space, but it is not enough to guarantee that the distribution of the generated data fits the real data distribution. TrVAE performs condition transformations based on condition variables, it is too hard to accurately capture responses to perturbation via the information provided only by condition vectors, which seriously hurts its performance. Compared with these three methods, prediction data of scPreGAN are more like the real perturbed data since it applies the discriminators to the high-dimensional space, which prompts the generated distributions to conform to the real distributions. In addition, scPreGAN performs end-to-end condition transformations on the data via the neural networks rather than relying on conditional variables. Such distribution-based transformations make the generated data more realistic.

The results of the LPS dataset are shown in Figure 2c. After dimensionality reduction, the predicted data of the four methods cannot overlap with the real perturbed data. Among them, the predicted data of scGen and trVAE are similar to the unperturbed data not the perturbed data, and the predicted data of stVAE still show a large difference from the real data, which confirms our analysis above. Although the predicted data of scPreGAN do not overlap with the real perturbed data after dimensionality reduction, they are roughly in the same subarea (the orange and green parts in Fig. 2c), indicating that the predicted distribution is closer to the real distribution. For the LPS dataset, due to the large differences in biological information between different species, we do not expect the model to accurately predict the perturbed data but hope to predict the expression trends of intertest cells based on the information provided by other species. The results of scPreGAN satisfy our expectations.

4.2 The prediction data of scPreGAN can preserve biological information

The predicted data should embody biological information of the real data. We explore it from the point of marker genes of each cell type, Scanpy (Wolf *et al.*, 2018) is employed to identify differentially expressed genes (DEGs).

The top 100 DEGs between the real unperturbed data and the real perturbed data, and the top 100 DEGs between the predicted perturbed data of each method and the real unperturbed data are given by Scanpy, respectively. Then, the number of common DEGs is counted between the former and the latter.

As shown in Figure 3, the common DEGs from scPreGAN are much more than its from the other three methods on all three datasets. For the PBMC dataset, about one-third to half of DEGs between the real unperturbed data and the real perturbed data are preserved by prediction data of scPreGAN. For the H.poly dataset, the common DEGs from scPreGAN are far more than its from other methods, which reach or exceed half of the total number on several cell types; scGen and stVAE seldomly present valid biological findings since the common DEGs for all cell types are almost zero, showing that the performance of scPreGAN is rather stable and excellent.

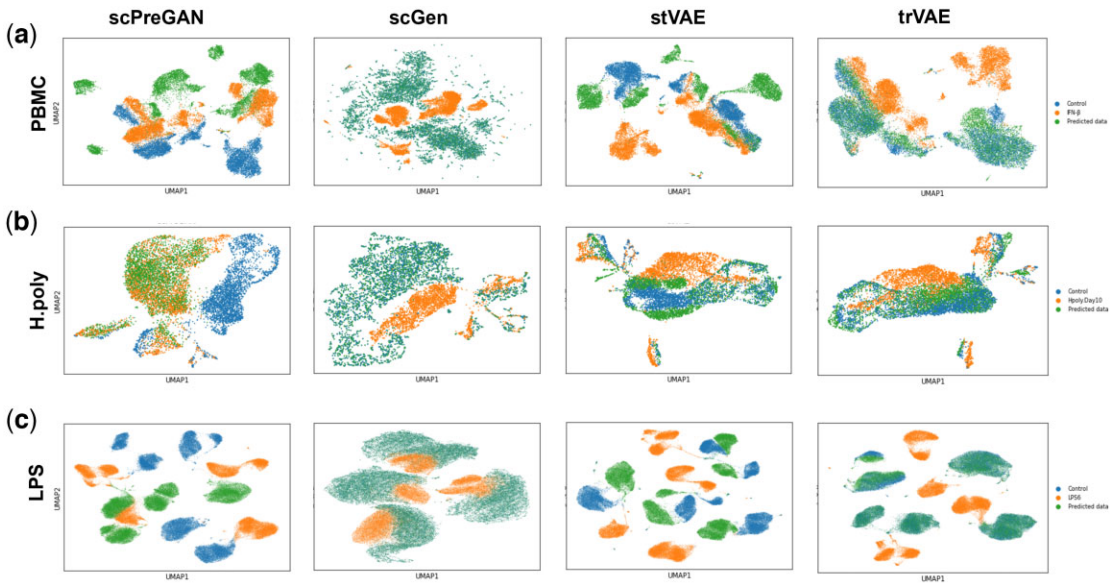

Fig. 2. UMAP dimensionality reduction of the real unperturbed data, the real perturbed data and the predicted perturbed data. (**a**) UMAP dimensionality reduction of the PBMC dataset. (**b**) UMAP dimensionality reduction of the H.poly dataset. (**c**) UMAP dimensionality reduction of the LPS dataset

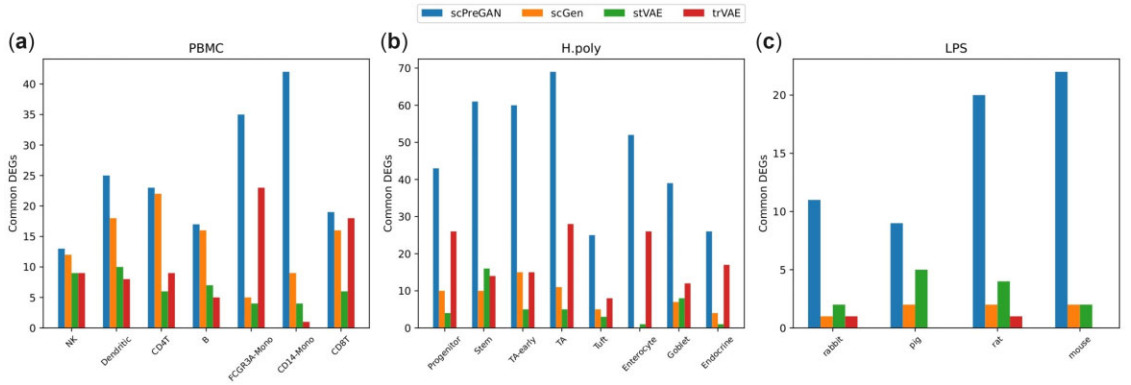

Fig. 3. The number of common DEGs of the top 100 DEGs from the real unperturbed data and the real perturbed data versus the top 100 DEGs from the predicted perturbed data versus the real unperturbed data. (**a**) Common DEGs results of the PBMC results. (**b**) Common DEGs results of the H.poly dataset. (**c**) Common DEGs results of the LPS dataset

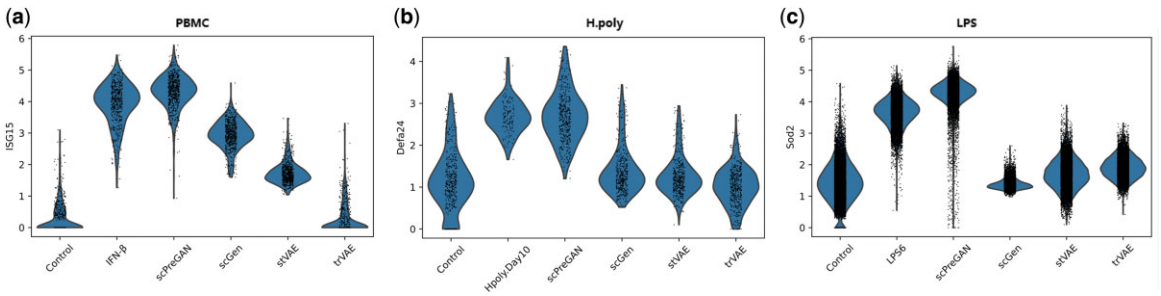

Fig. 4. Violin plots of the top DEGs in some cell types. The left to right plots are drawn from the unperturbed data, the perturbed data, the prediction of scPreGAN, scGen, stVAE and trVAE, respectively. (**a**) The violin plots of the *ISG*15 gene from CD14-Mono cells in the PBMC dataset. (**b**) The violin plots of the *Defa*24 gene of Enterocyte cells from the H.poly dataset. (**c**) The violin plots of the *Sod*2 gene from rat specie in the LPS dataset

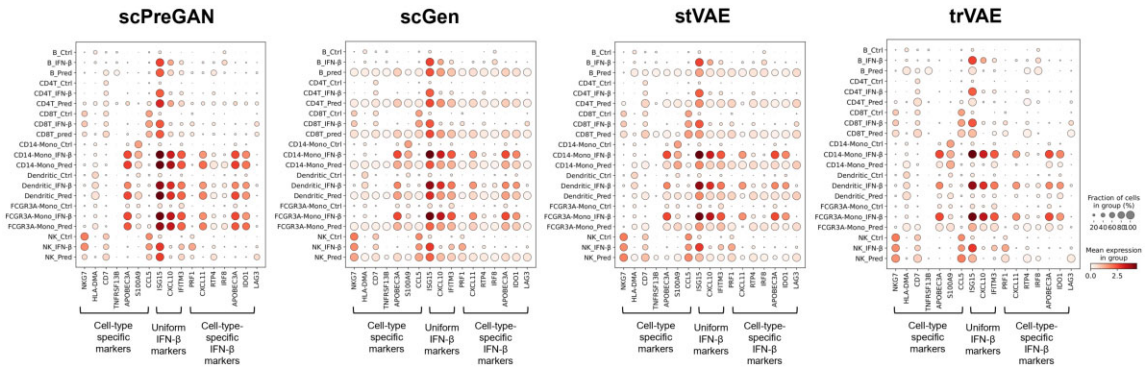

Fig. 5. Dot plots of some key genes in PBMC dataset. The left to right plots are the results of scPreGAN, scGen, stVAE and trVAE, respectively

The LPS dataset is across species, large gaps exist in the data among species, therefore, the number of common DEGs decreases. While, scPreGAN still pops out the most common DEGs. The results on rat and mouse are better than that on rabbit and pig, the reason may be that the rat and mouse species are more similar, and the model transmits more effective information between these two species.

To further explore the similarity of the generated distribution and the real distribution, we visualize the distribution of the top DEG in some cell types shown in Figure 4. As we can see, the generated distribution of scPreGAN is the most similar to the distribution of the real perturbed data. Except the prediction data of scPreGAN can give the same expression abundance, the predicted expression levels of other methods are much less than the real, the worst case is absent or very lowly expressed.

Besides, to test whether both cell type property and response information to perturbation are retained in prediction, we identify (i) cell markers of each unperturbed cell type, (ii) differentially expressed genes between the real unperturbed data and the real perturbed data, which are common response markers to perturbation and (iii) cell-type-specific markers to perturbation. All these markers are identified by Scanpy. For each cell type, a top cell type marker and a top cell-type-specific marker are selected. And three common response markers are selected. On these selected genes, shown in Figure 5, according to each cell type, we list the prediction of each method (end with 'Pred'), together with unperturbed data (end with 'Ctrl'), the real perturbed data (end with 'IFN-β') for clear comparison.

We could figure out that scPreGAN gives effective prediction, cell-type-specific information, common response to perturbation

and cell-type-specific response to perturbation are retained. ScGen performs well on common response to perturbation, while seems that cell-type-specific information is lost. Likely, cell-type-specific information is almost lost in prediction of stVAE. TrVAE fails to present the common response to perturbation.

4.3 Ablation study

ScPreGAN is composed of an encoder, two generators and two discriminators. The encoder and a generator constitute an AE model. The encoder aims to capture the common feature of unperturbed data and perturbed data. A generator is to regenerate the unperturbed data or the perturbed data, respectively, from the common latent representation given by the encoder. What is more important, the generator for perturbed data can predict the response to perturbation. A generator and a discriminator form a GAN model. The discriminator is to distinguish the prediction from the real data, the generator is trained against the discriminator to fool it.

In order to validate whether each part of scPreGAN is necessary, we have conducted the ablation study on the PBMC dataset. Specifically, we set the weight of each part in the loss function to zero so that it cannot take effect and there are seven different sub-models in total, shown in Table 1.

We evaluate these sub-models from whether the prediction could be classified as perturbed data and whether the prediction is similar as the real target. For the first metric, we train a *k*-nearest neighbors classifier based on the real perturbed data and the unperturbed data, and then calculate the accuracy that the predicted perturbed data are classified as perturbed data. For the second metric, we calculate the squared Pearson correlation (R^2) between the predicted perturbed data and the real perturbed data for each cell type and we show the mean and standard deviation.

As shown in Table 1, adversarial loss is necessary to make the prediction mixed with the real perturbed data. It is the key weapon for simulating the perturbed data. Reconstruction loss contributes most to retain the cell-type-specific property. And the combination of adversarial loss, reconstruction loss and encoding loss perform best with most stable.

5 Discussion

In this study, we propose scPreGAN, a novel method for perturbation-response prediction of scRNA-seq data. ScPreGAN combines the architecture of AE and GAN. An encoder is used to encode the scRNA-seq data, i.e. to extract the common information of the unperturbed and perturbed data. Two generators are used to generate the unperturbed and perturbed data, respectively, i.e. the latent vectors are remapped to the high-dimensional space. Besides, two discriminators are used to conduct adversarial training against the generators separately, which can promote the distributions of the generators more in line with the real distribution. We compared scPreGAN with three representative methods on several real datasets and evaluated their performances in the perspective of dimensionality reduction visualization, and biological information retention capability on several real datasets. The results show that scPreGAN performs best overall. The predicted perturbed data of scPreGAN not only are the most similar to the real perturbed data but also present the same expression abundance as the real data, which is hard for other methods. What is more, scPreGAN can accurately capture and regenerate the real distribution, even it is very complicated like bimodal.

ScPreGAN can be used to predict the effect of a drug or drug combination. Since the cell type is known in advance in our experiments, the adversarial loss part of scPreGAN (in Section 2.3.1) utilizes these cell-type labels, which represent as these terms of the L_C part in Equation (4). If the cell-type labels are unknown, these terms of the LC part in Equation (4) can be simply removed.

As shown in Figure 3, the performance of scPreGAN on rare cell type (Tuft of H.poly dataset, NK of PBMC dataset) looks not good as it on abundant cell type. The reason may be that it is not easy to get a good latent representation for a cell of rare cell type. In future work, we will explore to separate the training of AE part and GAN part of scPreGAN, and incorporate the control data of a cell type to be predicted into the training of AE part. If the cell types are known in advance, we could adjust the loss function of reconstruction part to amplify the reconstruction loss of rare cell type to make the encoder output effective latent representation of rare cell type.

By now, the prediction across species cannot reach the accuracy of prediction within a species. The reason lies in that the difference across species and the essential mechanism of perturbation have not been effectively disentangled. In future work, we could collect more scRNA-seq data across species, data after perturbation is not necessary, to capture and model their difference in the latent space first, and then tune the model on the paired data with and without perturbation to represent the essential mechanism of perturbation.

Table 1. Results for ablation study

Loss	Accuracy	R^2
$L_{adv} + L_{enc} + L_{rec}$	0.9727	0.8882 ± 0.0541
$L_{enc} + L_{rec}$	0.0395	0.7593 ± 0.106
$L_{adv} + L_{rec}$	0.9670	0.7380 ± 0.1798
$L_{adv} + L_{enc}$	0.8511	0.7652 ± 0.1427
L_{adv}	0.6946	0.1021 ± 0.0659
L_{enc}	0	0.0096 ± 0.0042
L_{rec}	0.3371	0.8068 ± 0.1455

Acknowledgements

The authors would like to thank Liuchuan Zhu and Haisha Xin for many helpful discussions and Yanjie Wang for useful suggestions on the codes and other kindly help. The authors also appreciate the anonymous reviewers for deeper discussions and great suggestion to polish this manuscript.

Funding

This work was supported by a grant from the National Natural Science Foundation of China. [61472086]. The funding source played no role in the design of the study and collection, analysis and interpretation of data and in writing the manuscript.

Conflict of Interest: none declared.

Data availability

All the datasets used for analysis in this study are public and published in other papers. We have referenced and described them in the manuscript.

References

Antoniou,A. *et al.* (2017) Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340. https://doi.org/10.48550/arXiv.1711.04340

Calimeri,F. *et al.* (2017) Biomedical data augmentation using generative adversarial neural networks. In: *International Conference on Artificial Neural Networks, Alghero, Sardinia, Italy*. Springer, pp. 626–634.

Dai Yang,K. *et al.* (2021) Multi-domain translation between single-cell imaging and sequencing data using autoencoders. *Nat. Commun.*, **12**, 1–10.

Datlinger,P. *et al.* (2017) Pooled CRISPR screening with single-cell transcriptome readout. *Nat. Methods*, **14**, 297–301.

Goodfellow,I. *et al.* (2014) Generative adversarial nets. In: *Advances in Neural Information Processing Systems*, *27*, *Montreal, Quebec, Canada*.

Haber,A.L. *et al.* (2017) A single-cell survey of the small intestinal epithelium. *Nature*, **551**, 333–339.

Hagai,T. *et al.* (2018) Gene expression variability across cells and species shapes innate immunity. *Nature*, **563**, 197–202.

Johansen,N. and Quon,G. (2019) ScAlign: a tool for alignment, integration, and rare cell identification from scRNA-seq data. *Genome Biol.*, **20**, 1–21.
Kang,H.M. *et al.* (2018) Multiplexed droplet single-cell RNA-sequencing using natural genetic variation. *Nat. Biotechnol.*, **36**, 89–94.
Karczewski,K.J. and Snyder,M.P. (2018) Integrative omics for health and disease. *Nat. Rev. Genet.*, **19**, 299–310.
Kingma,D.P. and Welling,M. (2013) Auto-encoding variational Bayes. arXiv preprint arXiv:1312.6114. https://doi.org/10.48550/arXiv.1312.6114
Korthauer,K.D. *et al.* (2016) A statistical approach for identifying differential distributions in single-cell RNA-seq experiments. *Genome Biol.*, **17**, 1–15.
Lotfollahi,M. *et al.* (2019a) Conditional out-of-sample generation for unpaired data using trVAE. arXiv preprint arXiv:1910.01791. https://doi.org/10.48550/arXiv.1910.01791
Lotfollahi,M. *et al.* (2019b) ScGen predicts single-cell perturbation responses. *Nat. Methods*, **16**, 715–721.
Lotfollahi,M. *et al.* (2021) Compositional perturbation autoencoder for single-cell response modeling. bioRxiv 2021.04.14.439903. https://doi.org/10.1101/2021.04.14.439903.
Makhzani,A. *et al.* (2015) *Adversarial autoencoders*. arXiv preprint arXiv:1511.05644. https://doi.org/10.48550/arXiv.1511.05644
Marouf,M. *et al.* (2020) Realistic in silico generation and augmentation of single-cell RNA-seq data using generative adversarial networks. *Nat. Commun.*, **11**, 1–12.
McInnes,L. *et al.* (2018) UMAP: uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426. https://doi.org/10.48550/arXiv.1802.03426
Metzker,M.L. (2010) Sequencing technologies—the next generation. *Nat. Rev. Genet.*, **11**, 31–46.
Mirza,M. and Osindero,S. (2014) Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784. https://doi.org/10.48550/arXiv.1411.1784
Miyato,T. *et al.* (2018) Spectral normalization for generative adversarial networks. arXiv preprint arXiv:1802.05957. https://doi.org/10.48550/arXiv.1802.05957
Nawy,T. (2014) Single-cell sequencing. *Nat. Methods*, **11**, 18.
Odena,A. *et al.* (2017) Conditional image synthesis with auxiliary classifier GANs. In: *International Conference on Machine Learning, Sydney, Australia*. pp. 2642–2651. PMLR.
Papalexi,E. and Satija,R. (2018) Single-cell RNA sequencing to explore immune cell heterogeneity. *Nat. Rev. Immunol.*, **18**, 35–45.
Patel,A.P. *et al.* (2014) Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma. *Science*, **344**, 1396–1401.
Rampášek,L. *et al.* (2019) Dr. VAE: improving drug response prediction via modeling of drug perturbation effects. *Bioinformatics*, **35**, 3743–3751.
Russkikh,N. *et al.* (2020) Style transfer with variational autoencoders is a promising approach to RNA-seq data harmonization and analysis. *Bioinformatics*, **36**, 5076–5085.
Schubert,M. *et al.* (2018) Perturbation-response genes reveal signaling footprints in cancer gene expression. *Nat. Commun.*, **9**, 20.
Shapiro,E. *et al.* (2013) Single-cell sequencing-based technologies will revolutionize whole-organism science. *Nat. Rev. Genet.*, **14**, 618–630.
Sohn,K. *et al.* (2015) Learning structured output representation using deep conditional generative models. *Adv. Neural Inf. Process. Syst.*, **28**, 3483–3491.
Targonski,C. *et al.* (2020) Cellular state transformations using deep learning for precision medicine applications. *Patterns*, **1**, 100087.
Way,G.P. and Greene,C.S. (2018) Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. *Pac. Symp. Biocomput.*, **23**, 80–91.
Wolf,F.A. *et al.* (2018) SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.*, **19**, 1–5.
Zhou,Y.-F. *et al.* (2019) BranchGAN: unsupervised mutual image-to-image transfer with a single encoder and dual decoders. *IEEE Trans. Multimedia*, **21**, 3136–3149.