scPerb: single-cell perturbation via style transfer-based variational autoencoder

Zijia Tang
Guangdong Experimental High School
Guangzhou, Guangdong, China
zijiatang2006@gmail.com

Zhengyong Hu
Guangdong Experimental High School
Guangzhou, Guangdong, China
hzy@gdsyzx.edu.cn

Qianqian Song
Department of Cancer Biology
Wake Forest University School of Medicine
Winston-Salem, NC, USA
qsong@wakehealth.edu

Abstract

Traditional methods for obtaining cellular responses after perturbation are usually labor-intensive and costly, especially when working with rare cells or under severe experimental conditions. Therefore, accurate prediction of cellular responses to perturbations is of great importance in computational biology. To address this problem, some methodologies have been previously developed, including graph-based approaches, vector arithmetic, and neural networks. However, these methods either mix the perturbation-related variances with the cell-type-specific patterns or implicitly distinguish them within black-box models. In this work, we introduce a novel framework, scPerb, to explicitly extract the perturbation-related variances and transfer them from perturbed data to control data. scPerb adopts the style transfer strategy by incorporating a style encoder into the architecture of a variational autoencoder. Such style encoder accounts for the differences in the latent representations between control cells and perturbed cells, which allows scPerb to accurately predict the gene expression data of perturbed cells. Through the comparisons with existing methods, scPerb presents improved performance and higher accuracy in predicting cellular responses to perturbations. Specifically, scPerb not only outperforms other methods across multiple datasets, but also achieves superior R^2 values of 0.98, 0.98, and 0.96 on three benchmarking datasets.

Keywords— Single-cell RNA sequencing, Perturbation, Style transfer, Variational auto-encoder

CONTENTS

1 INTRODUCTION

Single-cell RNA sequencing (scRNA-seq) is a revolutionary technology to profile gene expression of cells in heterogeneous tissue samples [1-3]. This technology can measure transcripts in thousands of single cells from multiple biological samples under different conditions [4-8]. Such breakthrough technology has inspired the development of tailored computational tools such as cell type annotations [9-12], identification of pseudo-time trajectories [13, 14], and rare cell type detection [15, 16], facilitating the biological insights into single-cell data [17, 18].

Although scRNA-seq technologies have led to a remarkable growth of single-cell data, it is still challenging to collect the matched pairs of control and perturbed samples for a particular perturbation. As current databases comprise a wide variety of single-cell data collected from samples at normal conditions, there is a critical need to leverage the existing data at normal conditions to generate and predict the single-cell data after a certain perturbation. To achieve this, an accurate and robust method is needed, with generalized capabilities in revealing gene expression patterns across different tissues, different platforms, and limited data size.

In recent studies, the gaps in perturbation tasks were filled using generative models like Generative Adversarial Networks (GAN) [19] and Variational Auto-Encoders (VAE) [20], to fill up the missing pieces in perturbation tasks. Specifically, GAN introduced a generator to construct fake perturbed data and trained an adversarial discriminator to determine whether the generated data was close to the real data or not. Such adversarial battle aimed to train a robust generator to infer high-quality data samples. However, the major drawback of GAN lay in the difficulty in balancing the adversarial training, leading to a useless collapsed generator that was very sensitive to the input data noise. sc-WGAN [21] transferred a more stable WGAN to the single-cell perturbation and style-transfer GAN (stGAN) [22] introduced the idea of style transferring that transferred multiple styles determined by the users to the generator. On the other hand, VAE generated data by sampling from a multivariate Gaussian distribution and used an encoder to estimate the mean and variance of the Gaussian distribution components of the original distributions, and new data observations are generated based on the estimated distribution using variational inference. For example, scGen [23] assumed a fixed linear gap between the control cells and the perturbed cells, calculated the latent difference from both datasets, and predicted the perturbed cell response using latent representation from the control cells and the perturbed cells. Conditional Variational Auto-Encoder (CVAE) [24] introduced more constraints to the neural network, allowing the prediction of perturbed data.

In this work, we presented a novel tool, i.e., scPerb, to predict single-cell gene expressions under specific conditions such as a dose [25], a treatment [26, 27], or a modification of genes [28-30] (**Fig.1**). Given two datasets generated under different conditions, for the same cell type, we denoted X_i^{ctrl} to represent the ith cell from the control condition, and X_j^{perb} for the jth cell from the perturbed dataset. scPerb solved the perturbation task by learning the latent features of cell types and the dataset-specific style vector. Inspired by the VAE architectures, scPerb first estimated the multi-variance normal distribution of the latent cell type feature c. Inspired by the stGAN [22], scPerb used a neural network to learn the style transformation matrix from the dataset. Compared with scGen, which adopts a constant vector to transfer the latent features from cells of the control dataset to perturbed cells, scPerb introduces learnable parameters and allows the neural network to learn both the style and content differences between the control and perturbed datasets. scPerb performs better and produces better prediction results when compared to other approaches.

2 METHODOLOGY

Inspired by the stGAN [22], we presented scPerb, a generative model to predict gene expression data after perturbation. We hypothesized the observations X^{ctrl} and X^{perb} from the control and perturbed datasets had two independent latent features: a cell type-related latent feature, denoted as "content" c; and a dataset-specific feature, denoted as "style" s. scPerb learned the contents Z_c^{ctrl} and Z_c^{perb} of the cell types from both the control and perturbed datasets, where c represented the content features of the cell types and transferred the style Z_s^{ctrl} from the control dataset to the perturbed dataset Z_s^{perb} , and s represented the dataset styles (**Fig. 1**).

scPerb was inspired by VAE [20] and stGAN [22]. Using an encoder, scPerb translated the input data into a probability distribution in the latent space. Specifically, it mapped the input data to a mean (μ) and a variance (σ) for each latent variable. We then projected the style vector s into the latent space and learned the transformation from the control dataset X^{ctrl} to the perturbed dataset X^{perb}, and the learned difference between X^{ctrl} and X^{perb} would be denoted as σ_s. Furthermore, we denoted $E_\mu^c(.)$ as the content encoder acquiring the cell-type awareness features, $E_\phi^s(.)$ as the style encoder projecting the random style vectors to the latent space, $E_\mu^c(.)$ and $E_\sigma^c(.)$ as the μ and σ estimation for the probability distribution generated by the encoders, and $D_\phi(.)$ as the decoder generating the perturbed data using the latent variables c and s. In the inference stage, given a specific cell type from the control dataset X^{ctrl}, scPerb would extract the cell type-related features Z_c^{ctrl}, generate the "fake" perturbed cell type $\hat{X}^{perb}$ based on Z_c^{ctrl} and σ_s, and minimize the differences between Z_s^{ctrl} and Z_s^{perb}.

2.1 Encoders

To extract common cell type content features, we projected both inputs (X^{ctrl}, X^{perb}) into the latent space. Followed by the setting of VAE, we assumed the content features were multivariate normal distributions, $N(\mu, \sigma)$, where μ and σ represented the mean and variance of multivariate normal distribution). The latent representation Z^{ctrl} of input data X^{ctrl} was obtained from the learned distribution

$$N(\mu^{ctrl}, \sigma^{ctrl}) : Z_c^{ctrl} \sim N(\mu^{ctrl}, \sigma^{ctrl})$$

,where $\mu^{ctrl} = E_\mu^c(E_\theta^c(X^{ctrl}))$ and $\sigma^{ctrl} = E_\sigma^c(E_\theta^c(X^{ctrl}))$.

Since the projection weights were shared between the two input datasets X^{ctrl} and X^{perb}, the latent representation Z^{perb} of input data X^{perb} was obtained from $Z_c^{perb} \sim N(\mu^{perb}, \sigma^{perb})$, , where $\mu^{perb} = E_\mu^c(E_\theta^c(X^{perb}))$ and $\sigma^{perb} = E_\sigma^c(E_\theta^c(X^{perb}))$. Followed by VAE settings, we used KL loss to estimate μ^{ctrl}, σ^{ctrl}, μ^{perb}, and σ^{perb}:

$$KLLoss^{ctrl} = KL(N(\mu^{ctrl}, \sigma^{ctrl}), N(0, I))$$
$$KLLoss^{perb} = KL(N(\mu^{perb}, \sigma^{perb}), N(0, I))$$

, where KL divergence was calculated by:

$$KL(P, Q) = \sum_{x \in X} P(x) log(\frac{P(x)}{Q(x)})$$

In this work, our task was to generate the "fake" perturbed cell types from the same cell types in the control dataset. Therefore, instead of learning the dataset styles explicitly, we applied a light-wise network to learn the transformation σ_s in the latent space. Our idea was inspired by the style transfer learnings [22], where randomly sampled style vector (s) and projected the latent space as the styles. In scPerb, we applied a style encoder $E_\phi^s(.)$, which can project the s into the latent space as the transformation variable to convert Z_c^{ctrl} to Z_c^{perb}:

$$\sigma_s = E_\phi^s(s)$$
$$\hat{Z}_c^{perb} = Z_c^{ctrl} + \sigma_s$$

Therefore, we had the following $StyleLoss$:

$$StyleLoss = SmoothL1Loss(Z_c^{perb}, \hat{Z}c^{perb})$$

While the $SmoothL1Loss$ was defined below:

$$SmoothL1loss(x, y) = \begin{cases} \frac{(x-y)^2}{2\beta} & if|x-y| < \beta \\ |x-y| - 0.5\beta & otherwise \end{cases}$$

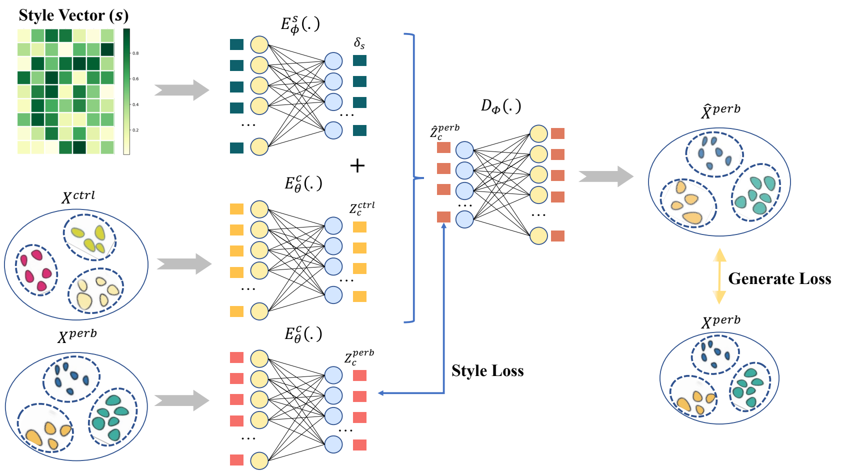

Fig. 1. scPerb predicts gene expressions of perturbed cells. scPerb was designed to predict gene expressions in perturbed cells and combines the principles of both style transfer and VAE. With the perturbed and control dataset as inputs, the content encoder projected the data into latent space. Differences between the latent representations of the perturbed dataset and the control dataset were captured by a Style Vector (s), which enabled transferring from the perturbed style to the control style. Such Style Vector was initiated with a random vector and updated via a style encoder, which learned the style of the perturbed dataset and transferred it to the control dataset by adding it to the latent representation of the control dataset. By minimizing the differences between both latent representations and gene expressions between predicted perturbed data and real perturbed data, scPerb transferred the control style to the perturbed style and predicted the gene expression of perturbed cells.

2.2 Decoder

In the decoder part, scPerb reparametrized the latent variable from the estimated posterior distribution $Z_c^{ctrl} \sim N(\mu^{ctrl}, \sigma^{ctrl})$ and $Z_c^{perb} \sim N(\mu^{perb}, \sigma^{perb})$. Unlike the standard VAE, which directly reconstructed the output $\hat{X}^{perb}$ from the latent variable Z_c^{ctrl} and Z_c^{perb} , scPerb converted the representation of the control data Z_c^{ctrl} to the latent representation $\hat{Z}_c^{perb}$, and generated the predicted perturbed data from decoder D_ϕ:

$$\hat{X}^{perb} = D_\phi(\hat{Z}_c^{perb})$$

Note that our task was to predict the perturbation of the cell types using the control dataset, instead of generating the samples from Z_c^{perb} and Z_c^{ctrl} as the original VAE, we only used $\hat{Z}_c^{perb}$ to generate $\hat{X}^{perb}$. Therefore, our $GeneratedLoss$ was:

$$GeneratedLoss = SmoothL1loss(X^{perb}, \hat{X}^{perb})$$

2.3 Loss function

The final objective function consisted of the $Generatedloss$, $StyleLoss$, and the KL regulation terms.

$$Loss = w_1 StyleLoss + w_2 KLLoss^{ctrl} + w_3 KLLoss^{perb} + w_4 Generatedloss$$

3 DATASETS AND PREPROCESS

We obtained the PBMC-Zheng dataset from Zheng et al. [31]. After removing the megakaryocyte cells that had uncertainly assigned labels, we log-transformed and normalized the data and selected the top 7,000 highly variable genes.

Kang et al. published a dataset from PBMCs including both control and perturbed cell types [25]. Among these data, we extracted the average of the top 20 cluster genes, which has 6,998 genes in total, from seven cell types, respectively: B cells, CD4-T cells, CD8-T cells, CD14 Mono cells, Dendritic cells, FCGR3A Mono cells, and NK cells, the same cell types as the PBMC-Zheng dataset.

Harber et al. presented a dataset using the responses of epithelial cells infected by Salmonella and H.poly [26]. In this dataset, there were 3,240 control cells, 2,711 H.poly-infected cells, and the rest 1,770 Salmonella-infected cells. The data were also normalized and log-transformed, and the top 7,000 highly variable genes were selected in this dataset.

In our model, we performed further data preprocessing to ensure consistency between control and perturbed cells within each cell type. Specifically, for each cell type, we randomly selected an equal number of cells from both the control and perturbed groups and used them to balance the dataset. This data preprocessing step helped us create a more robust and unbiased dataset, enabling accurate and fair comparisons between each cell type's control and perturbed conditions during subsequent analyses. By doing such data processing, we guaranteed that each pair of X^{ctrl} and X^{perb} have the same cell type, so the following style transfer process would be valid.

4 STATISTICS AND REPRODUCIBILITY

In scPerb, we evaluated the performance of our model under a fixed seed of 42 by using the square of the R value (R^2), calculated through $scipy.stats.linregress$ function [32]. This metric evaluated the degree to which the generated images and the real perturbed data were correlated. We computed the R^2 values for all genes' mean and variance and the top 100 Differential Expressed Genes (DEGs). To understand the model's results visually, we created scatter plots comparing the generated images to the corresponding ground truth data. This graph allowed us to observe how well the model's predictions aligned with the actual values.

Additionally, we used a violin plot to examine the discrepancies between the generated images and the real perturbed data for the top DEGs. The DEGs were identified using the $scanpy.tl.rank\_genes\_groups$ [33] function, employing the Wilcoxon method [34].

Through these analyses, we aimed to assess the accuracy and performance of our scPerb model based on the input gene expression data. The evaluation of R^2 values and the visualization of the scatter and violin plots provided valuable insights into the model's capabilities and highlighted any discrepancies between the generated and real perturbated data for further investigation.

5 RESULTS AND ANALASYS

5.1 SCPERB OUTPERFORMS OTHER BENCHMARKS

To demonstrate the performance of scPerb, we compared scPerb with currently existing methods, including scGen, CVAE, stGAN, and sc-WGAN [21-24]. Three datasets were used for benchmarking, including two published human peripheral blood mononuclear cell (PBMC) datasets, i.e., PBMC-Kang [25] and PBMC-Zheng [31] datasets, which were perturbed with interferon ($IFN - \beta$), and the intestinal epithelial cell dataset fetched by parasitic helminth H.poly [26], i.e., H.poly dataset.

Based on those three datasets, each method's performance was evaluated using the R^2 between predictions and real perturbed data. Specifically, we randomly selected a cell type to predict its gene expression data after perturbation, meanwhile using the rest of the cell types for model training. We repeated such process across all cell types and presented the average of the R^2 in **Fig. 2a**. In the PBMC-Zheng dataset, scPerb achieved the average R^2 score of 0.98, which was better than the performance of the competitors, including scGen (average R^2 = 0.94), CVAE (average R^2 = 0.93), stGAN (average R^2 = 0.39) and sc-WGAN (average R^2 = 0.10). Surprisingly, the GAN-based methods had much worse performance, as both GAN-based methods could not reach a R^2 value exceeding 0.5. Meanwhile, in the PBMC-Kang dataset, scPerb achieved the highest average R^2 score of 0.98, while the second-best and third-best approaches were scGen and CVAE which had 0.96 and

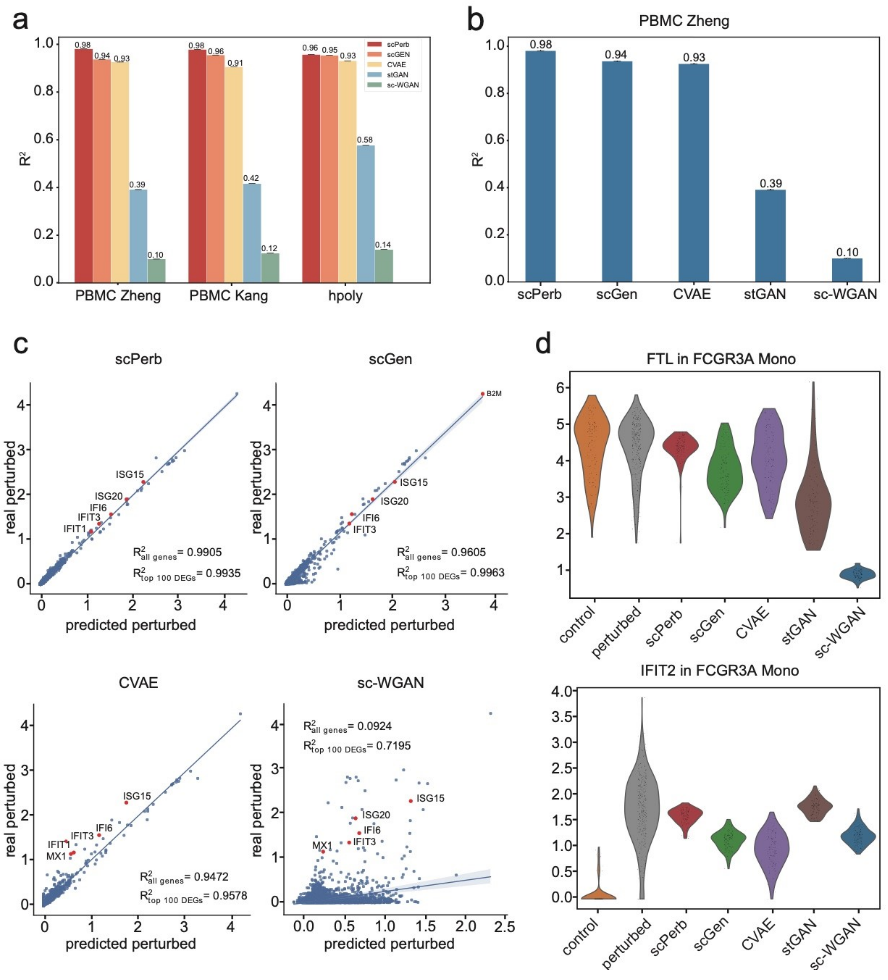

Fig. 2. The Overall result of scPerb. **a**: Comparison of R^2 values across all benchmarking methods; **b**: Bar plots showed the R^2 value of all methods in the PBMC-Zheng dataset [31]; **c**: Scatter plot showed the correlation between real and predicted gene expression of 7,000 genes by scPerb and other three benchmarking methods in CD4-T cells, and the five red dots represented the top five DEGs; **d**: The distribution of the control dataset, perturbed dataset, and the prediction of all methods in one of the least DEGs (*FTL*), and one of the top DEGs (*IFIT2*).

0.91. Similarly, the stGAN and sc-wGAN only had an average R^2 score of 0.42 and 0.12, respectively, in this dataset. Finally, we applied scPerb to the H.poly dataset and still got a 0.96 average R^2 score, followed by the scGen, CVAE, stGAN, and sc-wGAN with the average R^2 score of 0.95, 0.93, 0.58, 0.14. When comparing their results in a specific cell type, scPerb consistently outperformed other benchmarking methods (**Fig. 2b**). For example, in CD4-T cell type, one of the most numerous cell types in the PBMC-Zheng dataset, scPerb achieved a superior R^2 score of 0.99, which was much better than scGen, CVAE, stGAN, and sc-WGAN (R^2 score: 0.96, 0.95, 0.16, and 0.09) respectively.

In particular, we evaluated the performance of the proposed scPerb and the other benchmarking methods at the gene level. In **Fig. 2c**, we illustrated the prediction of our scPerb and the performance of the other three benchmarking methods in CD4-T cells from the PBMC-Zheng dataset. The scatter plot demonstrated that scPerb got the average R^2 score of 0.9905 when we used all the genes in this cell type. The performance could go up to 0.9935 when we only consider the top 100 DEGs. In comparison under the same setting, scGen achieved the average R^2 score of 0.9605 over all genes and 0.9963 on the top 100 DEGs. Our scPerb could outperform CVAE (average R^2 score of all genes = 0.9472, average R^2 score of top 100 DEGs = 0.9578) and sc-WGAN (average R^2 score = 0.0924, average R^2 score = 0.9578) on both the evaluation criteria. Specifically, DEGs including *IFIT1*, *IFIT3*, *IFI6*, *ISG20*, and *ISG15*, showed the best performance.

In **Fig. 2d**, the distribution of *IFIT2* in the control dataset varied from its distribution in the perturbed dataset. Based on the predicted gene expression, the mean of scPerb's prediction was closest to the mean of the perturbed dataset, with a relatively large value range between 0.5 to 3.0 after log transformation. The scGen and sc-WGAN provided comparable predictions, but the mean of the predictions was slightly larger than the real perturbed data. In this particular gene, CVAE was more associated with the control data, and stGAN focused on the outliers with high gene expressions. In **Fig. 2e**, the distribution pattern of *FTL* in the control dataset was similar to the distribution in the real perturbed dataset. Under such scenarios, most of the predictions in scPerb were close to the mean of the perturbed data, while the predictions from scGen and CVAE were expanded to a larger range. stGAN responded to the high gene expressions while the predictions in sc-WGAN had lower gene expressions. To further illustrate that our result was better than that of benchmarks, we applied the Wilcoxon [34] test to these results. In this case, only scPerb resulted in an adjusted P value larger than 0.05 for both genes (0.1763, and 0.0742 respectively for the *FTL* gene and the *IFIT2* gene), which showed that the prediction of scPerb did not have a significant difference from the ground truth.

All benchmarking techniques, in contrast, produced P values less than 0.05, indicating a considerable departure from the altered sample. To be more specific, scGen scored 6.3×10^{-15} and 0.0033 for the *FTL* gene and the *IFIT2* gene, while CVAE scored 0.0307 and 1.63×10^{-9}, stGAN scored 4.81×10^{-109} and 3.14×10^{-103}, and sc-WGAN scored 2.01×10^{-31} and 2.41×10^{-10}. Therefore, scPerb demonstrated superior performance than the other benchmarking methods.

5.2 scPerb is an innovative model that predicts single-cell perturbation responses accurately

In this section, we aimed to show that scPerb could accurately predict the single-cell perturbation responses for other cell types. **Fig. 3a** illustrated scPerb's performance across multiple cell types. In CD4-T, CD14 Mono, and FCGR3A Mono cells, scPerb could achieve an average R^2 score = 0.99 in both the top 100 DEGs and all gene expressions. In Dendritic cells, the average R^2 score was 0.98 and 0.98 respectively. In B cells and NK cells, the performance of the top 100 DEGs was slightly better than the performance of all genes, which was 0.99 vs. 0.98 and 0.98 vs. 0.97 respectively. We also observed that in CD8-T cells, the performance of the top 100 DEGs was 0.94, which was slightly lower than the performance on all genes (average R^2 score = 0.96). **Fig. 3b** was the dot plot that demonstrated the correlation of representative genes among different cell types. In half of the selected genes, the dot plot showed a strong difference between the gene expression and the real perturbed gene expression. On the other half of the selected genes, we presented similar gene patterns in both the control and perturbed datasets. This dot plot suggested a strong association between the mean gene expression levels across all cell types in scPerb's predictions and those in the actual perturbed dataset, even when the gene expressions varied in the control dataset. The UMAP in **Fig. 3c** showed that the predicted gene expression from scPerb in CD4-T cells was well-correlated with the real perturbed gene expression in the latent space. In particular, for a specific gene *IFI6*, we also illustrated the consistent observation.

5.3 scPerb can accurately predict the perturbation of cells in multiple PBMC datasets

scPerb had robust results in multiple datasets. In PBMC-Kang dataset [25], scPerb still outperformed other methods, achieving 0.98 in the mean R^2 of all the cell types, followed by scGen with a R^2 of 0.96, CVAE with 0.91, stGAN with 0.42 and sc-WGAN with 0.12 (**Fig. 4a**). Moreover, scPerb precisely predicted the result of FCGR3A Mono cells, reaching R^2 of 0.9948 and 0.9978 respectively for all genes and its top 100 DEGs. The top 100 DEGs as well as the entire gene population showed lower R^2 values for alternative benchmark approaches including scGen, sc-WGAN, and style-transfer GAN. To be more specific, scGen produced R^2 values of 0.9623 and 0.9545 for all genes and the top 100 DEGs, respectively. For the same categories, sc-WGAN revealed R^2 values of 0.3303 and 0.8593, and stGAN produced R^2 values of 0.5223 and 0.7361, in the same categories. This scatter plot further reinforces scPerb's robust predictive abilities. Moreover, in *MT2A* genes, one of the top DEGs in FCGR3A Mono cells, which also had a control condition filled with

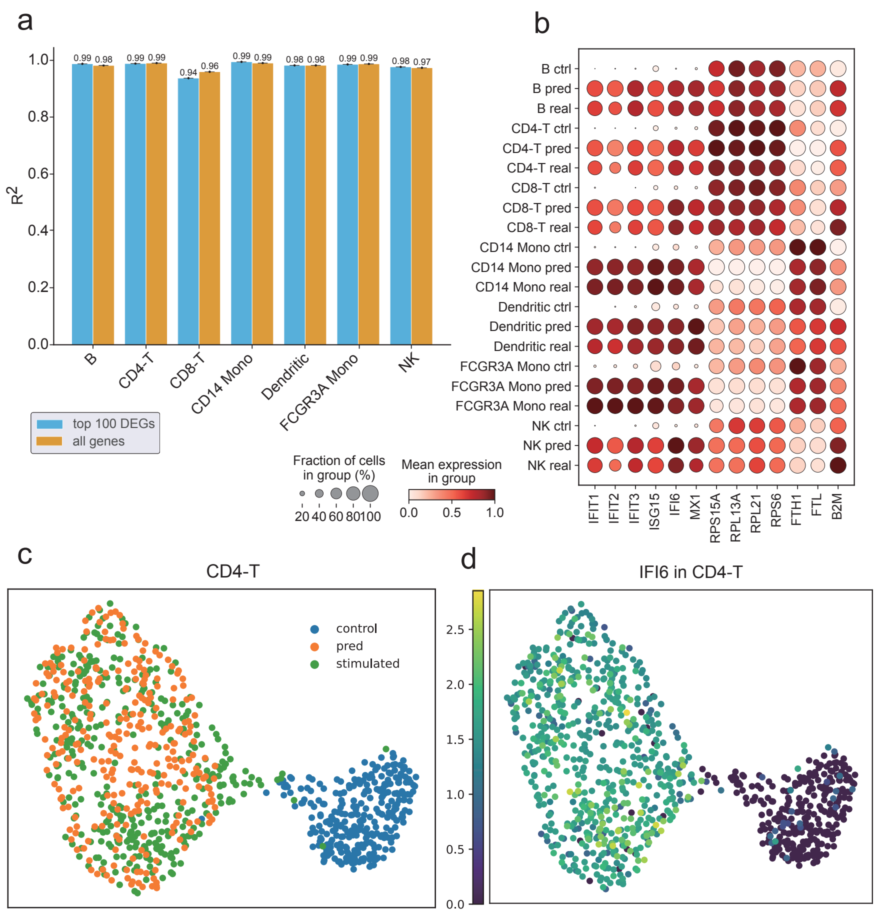

Fig. 3. Result of scPerb in PBMC-Zheng dataset. **a**: Grouped boxplot showed the result of scPerb in R^2 values in all genes and the top 100 DEGs in every cell type in the PBMC-Zheng dataset; **b**: Dot plot illustrating the mean gene expression in each cell type of control, real perturbed, and predicted perturbed condition, including the top DEGs and the least DEGs; **c-d**: UMAP [35] visualizations depicted the condition distribution of the overall CD4-T cell type in the PBMC-Zheng dataset and the expression pattern of *IFI6*, one of the top DEGs in the CD4-T cells.

zero values, scPerb made a better prediction than any other method, capturing the mean of the ground truth. In this case, the prediction of other methods barely captured the mean of the real perturbed data. (**Fig. 4b**) The Wilcoxon test can further explain the difference between the prediction and the real perturbed cells in the *MT2A* genes: only scPerb achieved a P value of 0.8785, meaning that the difference between the prediction of scPerb and the real perturbed data was not statistically different; however, all other methods including scGen, CVAE, and both GAN-based methods resulted in an adjusted P value far less than 0.0001, showing a significant difference between their predictions and the real perturbed data (**Fig. 4c**). Besides, the dot plot (**Fig. 4d**) showed that scPerb could get robust prediction no matter whether the original control gene expression was lower (for example the *IFIT1* gene), approximately the same (for example the *RPL13A* gene), or higher than (for example the *FTH1* gene) the ground truth. Moreover, it is worth noting that the prediction of scPerb correlated better with the real perturbed data, especially the top 5 DEGs (the red dots shown in **Fig. 4e**); and the R^2 values of scPerb (0.9950 and 0.9956 for all genes and the top 100 DEGs) were also higher than all the other benchmarks including scGen, CVAE, and sc-WGAN.

5.4 scPerb has robust results across different datasets

In the H.poly dataset [26], scPerb maintained superior performance with robust predictive capacity. For the cell types in the H.poly dataset, scPerb gained an average of R^2 as 0.96, which was better than the scGen and CVAE

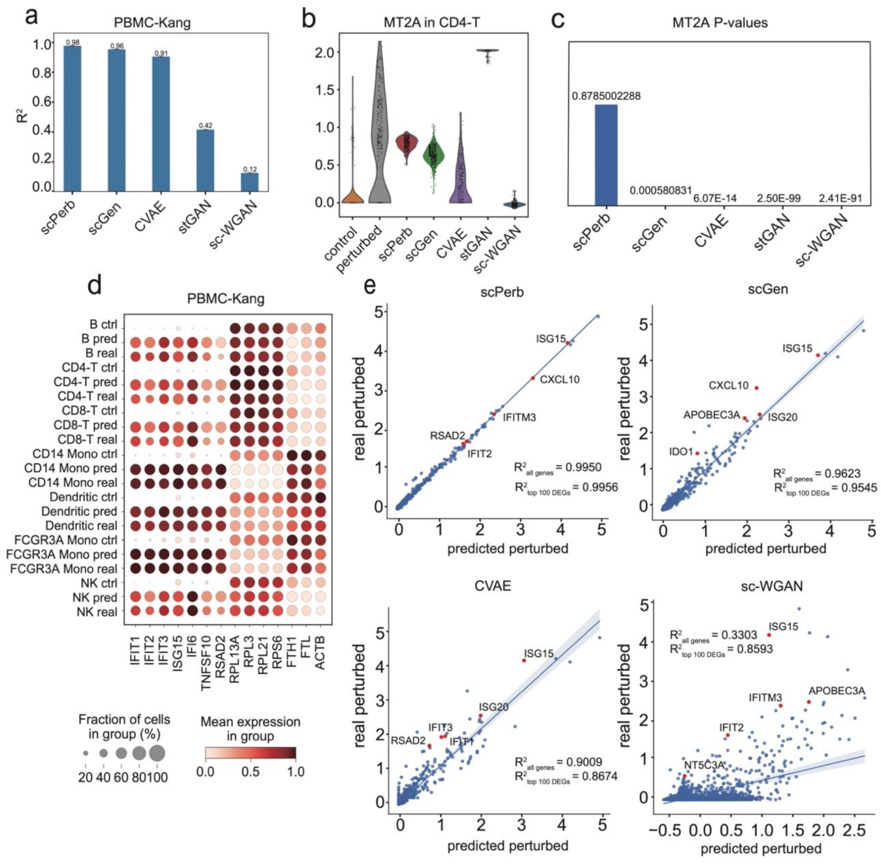

Fig. 4. Result of scPerb in PBMC-Kang dataset. **a**: This bar plot compared the R^2 values of all the methods within the PBMC-Kang dataset, while central values represented the mean R^2 values across all 7 cell types in the dataset; **b-c**: Comparing the distribution of all the methods in the *MT2A* gene in CD4-T cells in the PBMC-Kang dataset. Center values in **Fig. 4c** were the adjusted P values comparing the prediction of each method to the ground truth by using the Wilcoxon test [34]; **d**: A dot plot comparing the mean gene expression of all 7 cell types and all three conditions in the most and least DEGs in the PBMC-Kang dataset; **e**: The correlation of the mean expression of all 6,998 genes in FCGR3A Mono cells. It compared predictions from three of the best benchmark methods and scPerb against the ground truth, with shaded lines representing the 95% confidence interval of the regression estimate.

(scGen = 0.95, CVAE = 0.93), much better than stGAN and sc-WGAN (stGAN = 0.38, sc-WGAN = 0.14). The line plot in **Fig. 5a** also illustrated that scPerb maximized its difference in R^2 compared with other methods in Tuft cells, having R^2 = 0.94. While other VAE-based benchmarks had worse performance (scGen = 0.91, CVAE = 0.84). **Fig. 5a** also showed that all VAE-based methods (scPerb, scGen, CVAE) had a much better result than GAN-based methods (sc-WGAN, stGAN). In 7 out of 8 other cell types, scPerb showed superior performance than the benchmarking methods. Even for the endocrine cell type, in which scPerb presented less outperformance, it still achieved R^2 as 0.87, which was comparable with scGen (R^2 = 0.89).

Moreover, scPerb made better predictions in this dataset, especially in the Enterocyte. Progenitor cells. In **Fig. 5b**, the distance between the prediction (green dot) and real perturbed data (orange dot) was closer than the distance between the perturbed dataset to the control dataset (blue dot). For the other benchmarks, the VAE-base methods, scGen (**Fig. 5c**) and CVAE (**Fig. 5d**) could not easily divide the control data samples from the prediction and perturbed data, so their prediction resulted somewhere in between the control data samples and the perturbed data samples. And for the GAN-based methods, as shown in **Fig. 5e** for stGAN and **Fig. 5f** for sc-WGAN, the predictions were notably distant from both the control and perturbed datasets.

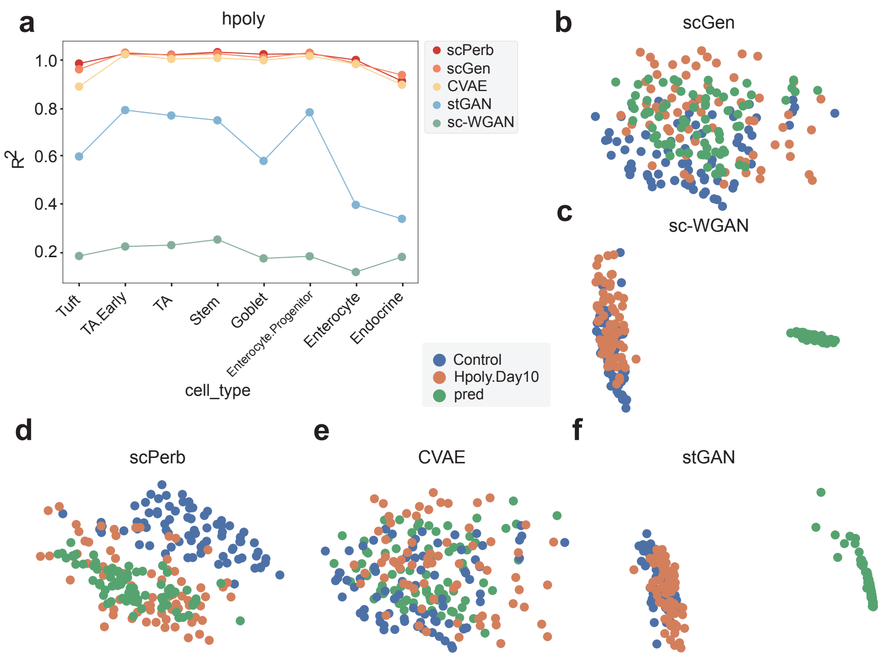

Fig. 5. The result of scPerb in the H.poly dataset a: Line plot using R^2 to compare the outcomes of all the methods; **b-f**: The UMAP visualization of the control (blue dots), perturbed (orange dots), and predicted (green dots) condition of all the methods.

6 DISCUSSION

scPerb is a generative model that dynamically transfers the gene expression in the control dataset into the reliable perturbed dataset. The encoder of scPerb projects the raw control gene data into the high-dimensional latent space. scPerb aggregates it with the dataset-specific styles to generate a high-quality representation for the perturbed dataset. Based on the representation, the decoder from scPerb can reconstruct gene expressions that are correlated with the mean of the perturbed dataset. The experiments demonstrate that scPerb can capture the latent content features and generate stable dataset-specific styles across different cell types and data from multiple studies. Moreover, the quantitative evaluation indicated the performance of scPerb outperforms four representative benchmarks, having state-of-the-art results in three different datasets.

Compared with traditional works [21-24], scPerb is a data-driven algorithm that can fully explore the gene expression in the raw dataset and does not rely on solid domain priors. On the opposite, the traditional works extract the principal components and build up a graph-based model in the low-dimensional manifold. Such methods rely heavily on the experienced domain knowledge, and lack of generalization abilities. Compared with other data-driven algorithms, scPerb incorporates the stableness from the VAE settings and exploits the advantage of the GAN to generate high-quality samples.

However, minor problems still exist. In Endocrine cells in the H.poly dataset, one of the cell types containing the fewest cells in the H.poly dataset (163 in 5,059), scPerb makes predictions slightly worse than scGen [23]. Using R^2 values as a criterion, scGen results in 0.89 while scPerb only results in 0.87. Note that scGen only calculates a fixed liner vector while scPerb uses style transfer, in this case, the problem of "overfitting" exists. However, such cases are very rare and scPerb can still outcompete "simple" methods like scGen in other cases when the data is small. In Tuft cells, also one of the cell types containing the fewest cells in the H.poly dataset (248 in 5,059), scPerb achieves a R^2 value of 0.94 while scGen only gets 0.91.

7 Reference

1. Baron, M., et al., A single-cell transcriptomic map of the human and mouse pancreas reveals inter-and intra-cell population structure. Cell systems, 2016. 3(4): p. 346-360. e4.

2. Puram, S.V., et al., Single-cell transcriptomic analysis of primary and metastatic tumor ecosystems in head and neck cancer. Cell, 2017. 171(7): p. 1611-1624. e24.

3. Athanasiadis, E.I., et al., Single-cell RNA-sequencing uncovers transcriptional states and fate decisions in haematopoiesis. Nature communications, 2017. 8(1): p. 2045.

4. Azizi, E., et al., Single-cell map of diverse immune phenotypes in the breast tumor microenvironment. Cell, 2018. 174(5): p. 1293-1308. e36.

5. Cusanovich, D.A., et al., A single-cell atlas of in vivo mammalian chromatin accessibility. Cell, 2018. 174(5): p. 1309-1324. e18.

6. Muraro, M.J., et al., A single-cell transcriptome atlas of the human pancreas. Cell systems, 2016. 3(4): p. 385-394. e3.

7. Iram, T. and T.M. Consortium, Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris. Nature, 2018. 562(7727): p. 367-372.

8. Buenrostro, J.D., et al., Integrated single-cell analysis maps the continuous regulatory landscape of human hematopoietic differentiation. Cell, 2018. 173(6): p. 1535-1548. e16.

9. Jagadeesh, K.A., et al., Identifying disease-critical cell types and cellular processes by integrating single-cell RNA-sequencing and human genetics. Nature genetics, 2022. 54(10): p. 1479-1492.

10. Shao, X., et al., scCATCH: automatic annotation on cell types of clusters from single-cell RNA sequencing data. Iscience, 2020. 23(3).

11. Crow, M., et al., Characterizing the replicability of cell types defined by single cell RNA-sequencing data using MetaNeighbor. Nature communications, 2018. 9(1): p. 884.

12. Wei, J.-R., et al., Identification of visual cortex cell types and species differences using single-cell RNA sequencing. Nature Communications, 2022. 13(1): p. 6902.

13. Tasaki, S., et al., Inferring protein expression changes from mRNA in Alzheimer's dementia using deep neural networks. Nature Communications, 2022. 13(1): p. 655.

14. Denyer, T., et al., Spatiotemporal developmental trajectories in the Arabidopsis root revealed using high-throughput single-cell RNA sequencing. Developmental cell, 2019. 48(6): p. 840-852. e5.

15. Torre, E., et al., Rare cell detection by single-cell RNA sequencing as guided by single-molecule RNA FISH. Cell systems, 2018. 6(2): p. 171-179. e5.

16. Wu, H., et al., Advantages of single-nucleus over single-cell RNA sequencing of adult kidney: rare cell types and novel cell states revealed in fibrosis. Journal of the American Society of Nephrology: JASN, 2019. 30(1): p. 23.

17. Andrews, T.S., et al., Tutorial: guidelines for the computational analysis of single-cell RNA sequencing data. Nature protocols, 2021. 16(1): p. 1-9.

18. Chen, G., B. Ning, and T. Shi, Single-cell RNA-seq technologies and related computational data analysis. Frontiers in genetics, 2019. 10: p. 317.

19. Goodfellow, I., et al., Generative adversarial nets. Advances in neural information processing systems, 2014. 27.

20. Kingma, D.P. and M. Welling, Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

21. Ghahramani, A., F.M. Watt, and N.M. Luscombe, Generative adversarial networks uncover epidermal regulators and predict single cell perturbations. bioRxiv, 2018: p. 262501.

22. Karras, T., S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.

23. Lotfollahi, M., F.A. Wolf, and F.J. Theis, scGen predicts single-cell perturbation responses. Nature Methods, 2019. 16(8): p. 715-721.

24. Cortes, C., et al. Advances in neural information processing systems 28. in Proceedings of the 29th Annual Conference on Neural Information Processing Systems. 2015.

25. Kang, H.M., et al., Multiplexed droplet single-cell RNA-sequencing using natural genetic variation. Nature biotechnology, 2018. 36(1): p. 89-94.

26. Haber, A.L., et al., A single-cell survey of the small intestinal epithelium. Nature, 2017. 551(7680): p. 333-339.

27. Hagai, T., et al., Gene expression variability across cells and species shapes innate immunity. Nature, 2018. 563(7730): p. 197-202.

28. Dixit, A., et al., Perturb-Seq: dissecting molecular circuits with scalable single-cell RNA profiling of pooled genetic screens. cell, 2016. 167(7): p. 1853-1866. e17.

29. Adamson, B., et al., A multiplexed single-cell CRISPR screening platform enables systematic dissection of the unfolded protein response. Cell, 2016. 167(7): p. 1867-1882. e21.

30. Datlinger, P., et al., Pooled CRISPR screening with single-cell transcriptome readout. Nature methods, 2017. 14(3): p. 297-301.

31. Zheng, G.X., et al., Massively parallel digital transcriptional profiling of single cells. Nature communications, 2017. 8(1): p. 14049.

32. Virtanen, P., et al., SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature methods, 2020. 17(3): p. 261-272.

33. Wolf, F.A., P. Angerer, and F.J. Theis, SCANPY: large-scale single-cell gene expression data analysis. Genome biology, 2018. 19: p. 1-5.

34. Cuzick, J., A Wilcoxon-type test for trend. Statistics in medicine, 1985. 4(1): p. 87-90.

35. McInnes, L., J. Healy, and J. Melville, Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426, 2018.

8 Acknowledgments

It is my privilege to have the opportunity as an intern of Prof. Song this summer. Previously, I had some prior experience in utilizing machine learning for medical image analysis. Therefore, Prof. Song hoped to exploit my previous experience and apply it to single-cell RNA-seq. Therefore, she came up with the idea of single-cell perturbation using the generative model. At the current stage, scRNA-seq can generate sufficient gene expression for the control dataset. However, it was not easy to get the data samples in response to the drugs, doses, and treatments, therefore we need to use computational tools to fill up the gap and generate the perturbed samples.

The perturbation task was very challenging because I didn't have a strong background in bioinformatics, and I spent a month reading tens of previous papers to get familiar with the topic. Thanks to scGen, one of our benchmarking, shared their codes, so Prof. Song recommended I start from there and build up the pipeline step by step. Together, we downloaded the dataset rewrote the TensorFlow codes into the PyTorch-style code, and regenerated the performance as the paper described. scGen was a top journal paper published in Nature Methods, and this paper provides solid results predicting the effect of perturbation which was hard to improve. However, when I rewrote the codes, I found that scGen was using a fixed linear transformation matrix to transfer the control cells into the perturbed samples. From my perspective, scGen did not reach the full potential of the neural network. Therefore, I spent months replacing the linear transformation with the different components, such as using multi-layers, multi-encoders, different loss functions like weighted MSE loss and ZINB loss, and even other models such as GANs or VQVAE. Finally, we found that applying a style encoder on the original VAE can have the best performance.

I would like to thank Prof. Song for her patience and insights during my internship at Wake Forest University. In the first few weeks, I didn't have many ideas and I discussed them with Prof. Song every day. She gave me many useful suggestions, hands-on with my codes, and helped me out in the right direction. When I figured out the methodology and was preparing for the final manuscript, Prof. Song taught me a lot about how to draw the figures, and how to write the manuscripts academically. We went over how to choose the word to clarify a sentence, how to polish the figures and manuscripts, and how to correctly insert the formats and references. It was Prof. Song who showed me a bright new world of computational biology and generative models and provided a platform where I could test all my thoughts. As a high school student, building up a paper from scratch was an unforgettable experience. I believe this experience would greatly benefit me in the future.

Moreover, thanks to my mentor Mr. Zhengyong Hu, who gave me lots of advice when I was stuck and had no idea how to improve the model's performance. He suggested I look up papers via Arxiv.com and the state-of-the-art applications in the field of generative models. Together we dived into VAEs and GANs and found a way to enhance the performance. My experience working with my mentor Zhengyong Hu will benefit my further academic research.