A Robust and Scalable Graph Neural Network for Accurate Single Cell Classification

Yuansong Zeng[1], Xiang Zhou[1], Zixiang Pan[1], Yutong Lu[1*], Yuedong Yang[1,2*]

[1]School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510000, China

[2]Key Laboratory of Machine Intelligence and Advanced Computing (MOE), Guangzhou 510000, China

*yangyd25@mail.sysu.edu.cn; yutong.lu@nscc-gz.cn

ABSTRACT

Motivation: Single-cell RNA sequencing (scRNA-seq) techniques provide high-resolution data on cellular heterogeneity in diverse tissues, and a critical step for the data analysis is cell type identification. Traditional methods usually cluster the cells and manually identify cell clusters through marker genes, which is time-consuming and subjective. With the launch of several large-scale single-cell projects, millions of sequenced cells have been annotated and it is promising to transfer labels from the annotated datasets to newly generated datasets. One powerful way for the transferring is to learn cell relations through the graph neural network (GNN), while vanilla GNN is difficult to process millions of cells due to the expensive costs of the message-passing procedure at each training epoch.

Results: Here, we have developed a robust and scalable GNN-based method for accurate single cell classification (GraphCS), where the graph is constructed to connect similar cells within and between labelled and unlabelled scRNA-seq datasets for propagation of shared information. To overcome the slow information propagation of GNN at each training epoch, the diffused information is pre-calculated via the approximate Generalized PageRank algorithm, enabling sublinear complexity for a high speed and scalability on millions of cells. Compared with existing methods, GraphCS demonstrates better performance on simulated, cross-platform, and cross-species scRNA-seq datasets. More importantly, our model can achieve superior performance on a large dataset with one million cells within 50 minutes.

1. Introduction

Single cell RNA-seq technologies promise to provide high-resolution insights into the complex cellular ecosystem[1-3] by measuring gene expression in millions of single cells from multiple samples [4-8]. Several large-scale single-cell projects, e.g. the Human Cell Atlas, have been established as a result of the decreasing costs in scRNA-seq technologies [9, 10]. In scRNA-seq studies, an essential step is to identify the sequenced cells through the sequenced gene expression [11], which is usually obtained through cell clustering and subsequently manually identifying cell clusters through marker genes[12]. This process is time-consuming and subjective.

With the tremendous increase of well-annotated scRNA-seq datasets, it is feasible to transfer well-defined labels (cell types) of existing single-cell datasets to newly generated single-cell datasets [13, 14]. However, the knowledge transferring is challenging due to various noises among scRNA-seq data (e.g., dropout) [15, 16]. In addition, batch effects exist between single-cell datasets because they are usually collected from different platforms[17, 18], tissues, or species[19, 20]. Early methods were developed to search for similar cells in the reference datasets with well-defined labels. For example, scmap[21] measures the maximum similarity between well-annotated cells of reference data and unknown query data to annotate cell types. SingleR[22] measures the similarity by calculating the correlation between gene expression. CHETAH[23] identifies the unknown cells using the high cumulative density of each cell type correlation distribution. Obviously, these methods consider only pairwise similarity and have ignored the non-linear relations between annotated cells. For this reason, several methods train classifiers using the labeled datasets or reference atlas, and make predictions on query datasets. For example, scPred [24] trains a support vector machine by using the features obtained from singular value decomposition. SingleCellNet [25] applies an ensemble of boosted regression trees and a Random Forest classifier to annotate cells. Seurat[26] is a commonly used toolkit in single-cell studies, which applies a specialized method to transfer labels to unknown cell types. Though valuable in different scenes, these methods still exhibit limited performance, partially due to their ignoration of higher-order relations between cells.

In fact, the high-order representation and topological relations could been naturally learned by the graph neural network (GNN), and GNN have been proven with improved performance in scRNA-seq data analyses such as imputation and clustering [27-29]. ScGCN[30] is currently the only graph neural network method for annotating cells. The method is based on the GNN architecture proposed by Kipf and Welling [31], which relies on an expensive message-passing procedure to propagate information and has to include the full-batch during training. Thus, the huge costs of computations and memory prevent its applications to large datasets, especially with the arrival of datasets containing millions of cells [32, 33].

To solve the scalability of GNN, many studies have been proposed. For example, Chen et al. [34] proposed a scalable GNN model, which could be efficiently trained with mini-batches using GPU. One critical point is its approximation of the diffused information through the bidirectional propagation by the Generalized PageRank algorithm [35], which avoids iterative information diffusion in each training epoch. In addition, the use of mini-batch training reduces the requirement of large GPU memory from full-batch training. Thus, the method could be used on large graphs with billions of edges.

Another issue for GNN is to accurately construct the cell graph among millions of cells. Traditional methods such as Cosine similarity, KNN, UMAP [36], and Annoy [37] (https://github.com/spotify/annoy) are widely used for constructing the cell graph by measuring the cell-to-cell similarity in single-cell RNA-seq data [38-40], but they do not take account of the batch effects between datasets. To consider the batch effects, several methods captures the cell relations through scGCN [30] constructs the cell graph using CCA-MNN, a combination of canonical correlation analysis (CCA) [41] and the mutual nearest neighbour (MNN) [42]. Conos [43] relies on multiple plausible inter-sample mappings to construct a graph connecting all measured cells. BBKNN [44] provides an extremely fast and scalable neighbourhood construction method across all batches. The runtimes of BBKNN scale linearly with the increase in number of cells through integrating the approximate neighbor detection technique in algorithm Annoy.

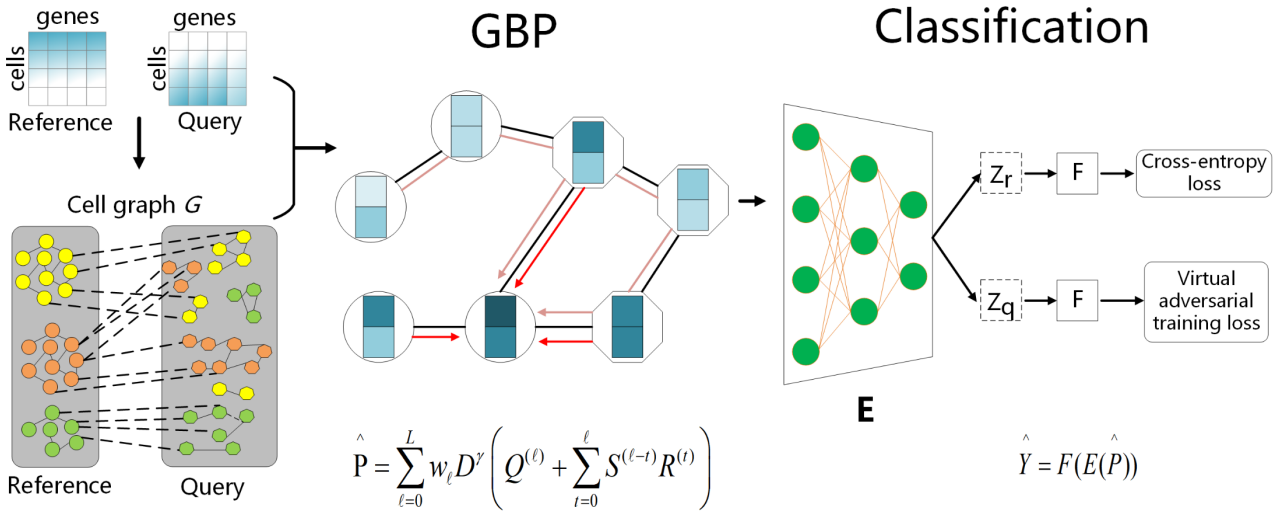

Fig. 1. The Schematic overview of GraphCS for cell type classification. GraphCS consists of graph construction, graph bidirectional propagation (GBP), and classification modules. The graph construction module constructs the cell graph according to gene expression similarity through the BBKNN algorithm. Through the graph, the GBP module diffuses feature information among cells, which is then inputted into the classification module used to classify cells.

Here, we present a scalable graph neural network learning model for cell annotations by constructing the graph via BBKNN, and pre-calculate the diffused features via the graph bidirectional propagation algorithm (GBP). Concretely, GBP propagates information among similar cells within and between labeled and unlabeled datasets, resulting in significant gains of speed and scalability of GNN while efficiently removing the batch effects. The integrated features from the GBP module are then inputted to a classification neural network to annotate cells for the query dataset. To better estimate the decision boundary between different cell types, we also use the virtual adversarial training (VAT) loss [45] to improve model generality. Our method was demonstrated to outperform other methods on both simulated datasets and real datasets across species and platforms. More importantly, the model can be extended to large-scale datasets in a reasonable time scale.

2. Materials and Methods

2.1 Datasets and Pre-processing

Simulated datasets: We simulated different batches of scRNA-seq data through the R package "Splatter" [46]. Specifically, we used Splatter to generate paired batches as the reference and query dataset. Each simulated dataset was comprised of four cell groups with the same cell proportion by setting the parameter group.prob of value 0.25. We fixed the number of cells in the reference and query data respectively as 2000 and 1000 both with the number of genes as 10000. To simulate different magnitudes of batch effects in scRNA-seq data, we tested different batch.facScale values from {0.2, 0.4, 0.6, 0.8,1.0,1.2,1.4, 1.6}, where greater values of batch.facScale correspond to larger batch effects between the reference and query datasets. For other parameters, default values were applied unless otherwise specified. We generated five simulated datasets from different random seeds and reported the average results.

Cross-species datasets: The cross-species datasets consisted of four paired cross-species pancreas datasets downloaded from the SingleCellNet GitHub page. Specifically, we used the mouse pancreas dataset generated by Baron et al[1] as the reference data and the human pancreas datasets generated by Baron et al[1] and Segerstolpe et al[47] as the query data, respectively. We also used the Baron mouse pancreas dataset to annotate a combination dataset containing five human pancreas datasets respectively generated by Baron et al, Wang et al[48], Xin et al[49], Muraro et al[6], and Segerstolpe et al. Finally, we used the human pancreas dataset generated by Baron et al as the reference data and the mouse pancreas dataset also generated by Baron et al as the query data. The compatible genes between species were obtained through homologous genes convert interface provided by SingleCellNet[25].

Multiple reference datasets: To investigate the performance of our model on multiple reference datasets, we integrated the human pancreas datasets by Wang et al, Muraro et al, and Segerstolpe et al as the reference to annotate the Baron human pancreas dataset. Specifically, we removed the acinar and alpha cells from datasets by Segerstolpe et al and Wang et al, only reserving them in the Muraro et al dataset. We also removed the endothelial cells from Muraro et al, only reserving them in the Segerstolpe et al dataset. Thus, each reference dataset has its own unique cell type, while the query dataset Baron et al includes all these unique cell types: cell acinar, alpha, and endothelial.

Cross-platform datasets: We used seven paired cross-platform datasets in this study. The first one is the Peripheral Blood Mononuclear Cells (PBMC) scRNA-seq data from the SeuratData package with dataset name "pbmcsca" [50], which is widely used for evaluating cell annotating methods. PBMC consisted of seven batches from seven different sequencing platforms: 10x Chromium (v2), 10x Chromium (v3), Seq-Well, Smart-seq2, inDrop, Drop-seq, and CEL-Seq2. Two 10x Chromium datasets were combined as the reference data, and the rest five datasets were used as the query data. Thus, PBMC has five paired cross-platform datasets. Then, in order to evaluate the scalability of our method, we downloaded two large cross-platform datasets: the mouse retina and mouse brain datasets from ref [51]. Specifically, the mouse retina dataset contained two batches that were generated based on the Drop-seq technology by two unassociated laboratories [52, 53]. These two mouse retina datasets contained 26,830 and 44,808 cells, and were used as the reference and query data, respectively. The mouse brain data contained two batches that were generated by the Drop-seq and SPLiT-seq protocols, respectively [39]. They were used as the reference and the query data that contained 691,600 and 141,606 cells, respectively.

Preprocessing: All simulated datasets were normalized through the transcripts per million (TPM) method [54]. For real datasets, we followed the standard procedure proposed in Seurat to normalize the gene expression matrix. Specifically, the "NormalizeData" function was run with the default parameter "LogNormalize" and the scaling factor of 10,000. Then, we selected the top 2000 highly variable genes based on the normalized matrix through the "FindVariableFeatures" function.

2.2 The architecture of GraphCS

This study proposed a robust and scalable graph neural network model to annotate cell types in a semi-supervised manner. As shown in Fig. 1, the GraphCS model consists of three modules: graph construction, graph bi-directional propagation (GBP), and classification modules.

2.2.1 Graph construction module

The cell graph G is constructed by linking cells with similar gene expressions within and between the reference and query datasets. Here, we construct the graph G by BBKNN with default parameters, which provides a fast and scalable neighbourhood construction method across all batches. Briefly, for each cell c, three most similar cells are selected with the lowest Euclidean distances from each of N_b batches (including the batch itself). The connected cell graph is then inputted into UMAP for recalculating connectivity scores, through which neighboured cells are trimmed so that each cell contains at most $30N_b$ neighbours (edges).

2.2.2 GBP module

To acquire high scalability, GNN is estimated through the Generalized PageRank algorithm, which is further approximated by the Graph Bidirectional Propagation Algorithm.

Generalized PageRank Algorithm

To acquire high scalability of GNN, the feature propagation is pre-calculated through Generalized PageRank matrix as:

$$P = \sum_{\ell=0}^{L} w_\ell (D^{\gamma-1} A D^{-\gamma})^\ell \cdot X \tag{1}$$

where w_ℓ is the weight of the ℓ-th order convolution matrix, A and D are the adjacency matrix and diagonal degree matrix of graph G, respectively, X is the feature matrix, and γ is the convolution coefficient. This strategy has been proven to well estimate feature propagation [55], and we followed the study to set $w_\ell = \alpha(1-\alpha)^\ell$ for constant decay factor $\alpha \in (0,1)$.

The Graph Bidirectional Propagation Algorithm

To reduce the time complexity, the Generalized PageRank is further approximated with the graph bidirectional propagation that combines the Monte-Carlo Propagation and Reverse Push Propagation. Graph bidirectional propagation has been proven to provide accurate unbiased estimator [34]. Concretely, we use the following formulate as an unbiased estimator for the Generalized PageRank matrix P defined in equation (1).

$$\hat{P}^{n\times d} = \sum_{\ell=0}^{L} w_\ell D^\gamma \cdot \left(Q^{(\ell)} + \sum_{t=0}^{\ell} S^{(\ell-t)} R^{(t)} \right) \tag{2}$$

where n is the total number of cells in the reference and the query data, and d is the size of gene features. Q and R are respectively the reserve matrix and the residue matrix originated from the Reverse Push Propagation algorithm, and S records the fraction of random walks from the Monte-Carlo Propagation. The detailed information and proof of Equation (2) can be found in ref [34].

2.2.3 Classification Module

After obtaining the feature matrix $\hat{P}^{n\times d}$ from the GBP module, we apply a neural network classification with mini-batch training to make the cell type prediction. In addition, the virtual adversarial training is employed to improve the generality of our model.

Neural Network Classification: Our classification module contains a neural network feature extractor E with multiple hidden layers and a label predictor F with a $Softmax$ output layer. The input of classification module includes reference gene expression matrix $\mathrm{X}_r = [x_1^r, \cdots x_{m_r}^r] \in \hat{\mathrm{P}}^{\,m_r\times d}$ with the corresponding labels $\mathrm{Y}_r = \{y_i^r\}_{i=1}^{m_r}$ and query gene expression matrix $\mathrm{X}_q = [x_1^q, \cdots x_{m_q}^q] \in \hat{\mathrm{P}}^{\,m_q\times d}$. We optimize the classification module using the following standard cross-entropy loss:

$$\mathrm{L}_{CE} = -\frac{1}{m_r}\sum_{i=1}^{m_r} y_{i,r}^T F\big(E(x_i^r)\big) \tag{3}$$

where $\mathrm{y}_{i,r} \in \mathrm{R}^{CL\times 1}$ is one-hot encoded vector of y_i^r and CL is the number of class.

Virtual Adversarial Training: Virtual adversarial training (VAT) is applied to improve the generalization of the classification module by incorporating the information of data distribution from query data. VAT is a data augmentation technique without prior label information[56]. VAT tries to make predictions invariant to small perturbation by minimizing the distance between the input and a perturbed version of the input. Then the model is robust to small noises or perturbations in the inputs. We compute VAT's loss function as the following:

$$L_{VAT}(X_q,\theta) = D_{KL}\big[p(Y_q|X_q,\theta), p(Y_q|X_q + r_{vat},\theta)\big] \tag{4}$$

$$r_{\mathrm{vat}} = \underset{\Delta x;\|\Delta x\|_2\le\epsilon}{\arg\max}\, D_{KL}\big[p(Y_q|X_q,\theta), p(Y_q|X_q+\Delta x)\big] \tag{5}$$

where r_{vat} optimizes the difference between the model output of the non-perturbed input and the perturbed input, θ is parameter of the model, Δx is a Gaussian noise and Y_q is predicted by the label predictor F. The hyper-parameter ϵ is the norm constraint for the adversarial direction, and we set ϵ to 0.1 following the previous study [45]. The output distribution is parameterized as $p(\mathrm{Y}_q|\mathrm{X}_q,\theta)$, and D_{KL} $[\cdot,\cdot]$ is KullbackLeibler divergence.

So, the total loss function of classification module as the following:

$$L_{overall} = L_{CE} + \lambda L_{VAT} \tag{6}$$

where λ is the hyper-parameters to balance the contribution of virtual adversarial training to the total loss function.

2.3 Model interpretation by selecting key features

In order to obtain important genes for each predicted cell type, we estimate gene importance based on the gradient of the correct category logit with respect to the input vector used the activation maximization method [57, 58]. Specifically, given the trained neural network φ and a predicted cell type i, activation maximization searches for key input genes x^* by solving the following optimization problem:

$$x^* = \underset{x}{\mathrm{argmax}}(\varphi(x)\cdot e_i) \tag{7}$$

where e_i is the i-th category's natural basis vector. The formulation of (7) can be solved by backpropagation, where the gradient of $\varphi(x)$ with respect to x are computed to update the input x iteratively. Specifically, we initialize the optimization with a zero vector and then run the optimization for 100 iterations with a learning rate of 1 as recommended in ref [59]. Those inputs leading to the largest changes in comparison with the initialization values are selected as the important genes. To evaluate the identified top-important genes, we conduct the GO analysis on the selected top genes with the largest changes using the R package clusterProfiler[60].

2.4 Hyper-parameters setting

The GraphCS was implemented in PyTorch and C++. For GBP module, we set α=0.05 and γ=0.5 for all datasets. For classification module, we set the number of neural network layers and the learning rate as 2 and 0.001, respectively. For the contribution of VAT loss, we set λ=0.1 for all datasets. We set the training batch size of classification module as 1024 and 4096 when the total number of cells exceeds 10000 and 50000, respectively; In other situations, the training batch size is set as 128. For cross-species datasets, the maximum of inter edges in the graph G (edges between reference and query data) for each cell is less than four. All results reported in this paper were conducted on Centos 7.0 with Intel® Core (TM) i7-8700K CPU @ 3.70 GHz and 256 GB memory, with the Nvidia Tesla P100 (16G).

2.5 Benchmarking classification methods

To evaluate the performance, we compared GraphCS with other tools including: Seurat V3, scmap, scPred, CHETAH, SingleR, SingleCellNet and scGCN. For Seurat V3, we applied both the PCA-based and CCA-based version to evaluate whether the aligned data was benefit for classification. We used the default hyper-parameters recommended in the origin paper for the competing methods.

Evaluation metrics: We evaluated the classification performance for all methods using the accuracy, the proportion of correctly annotated cells. For each dataset, we considered the cell type annotations provided by the original dataset as the ground truth.

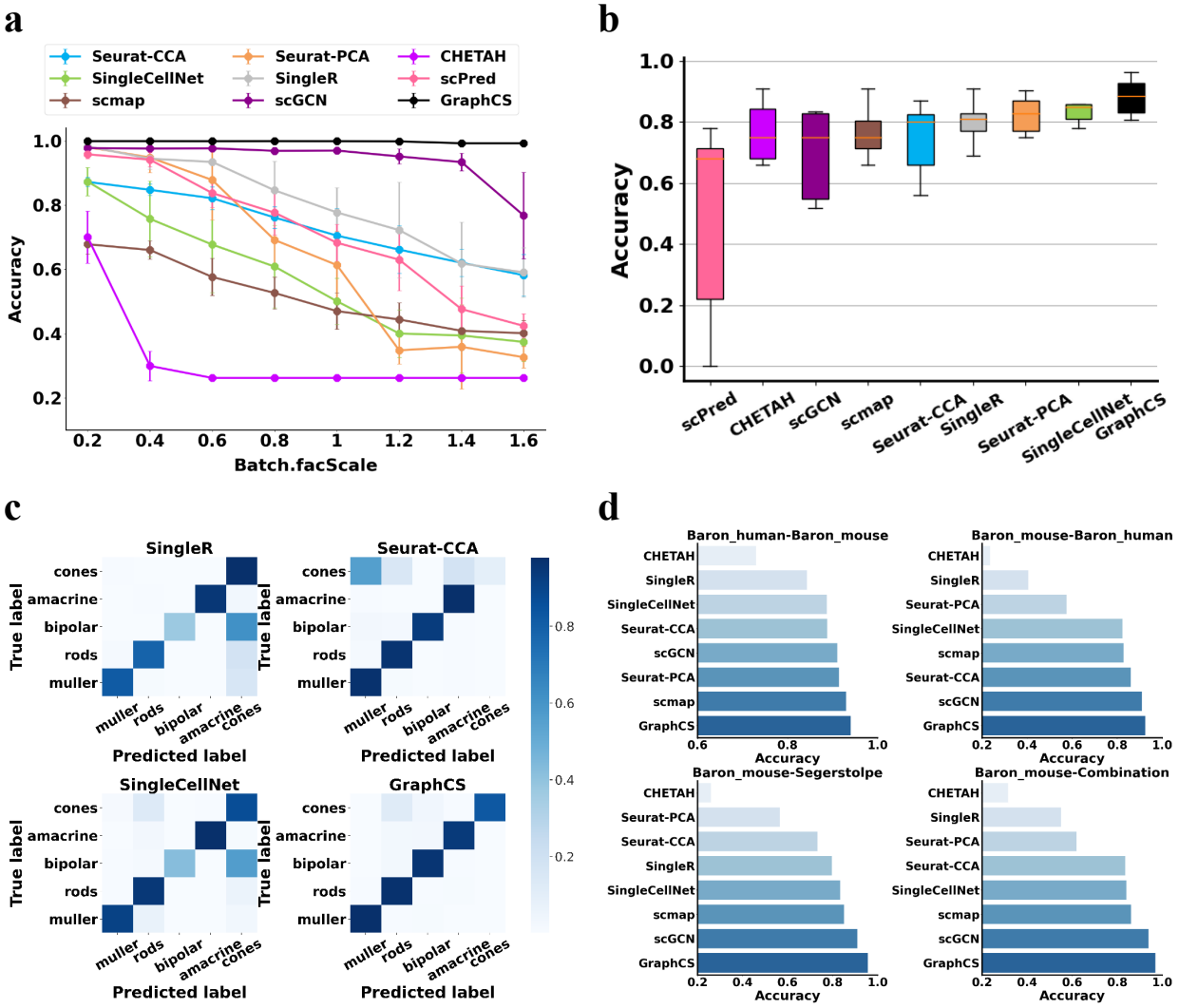

Fig. 2. The performance of GraphCS on simulated, cross-platform, and cross-species datasets: **(a)** the average and mean square error values of cell type prediction accuracy on 5 groups of simulated scRNA-seq data at different batch.facScale values; **(b)** the boxplots of cell type prediction accuracy of all methods based on the cross-platform datasets; **(c)** the accuracy matrix of each cell type identified by different methods on the mouse retina dataset; **(d)** the performance of GraphCS on four paired cross-species datasets. Baron_human-Baron_mouse represents the Baron human pancreas dataset as the reference to annotate the Baron mouse pancreas dataset. The rest results represent that the Baron mouse pancreas dataset as the reference to annotate respectively the Baron, Segerstolpe, and the combination human pancreas datasets (the combination contained five human pancreas datasets, including Baron et al, Wang et al, Xin et al, Muraro et al, and Segerstolpe et al). Each bar represents the accuracy of each method.

3. Results

3.1 Performance on simulated datasets

To investigate the performance of GraphCS under different magnitudes of batch effects, we generated the simulated scRNA-seq data by setting different values of "batch.facScale" through the R package "Splatter". As shown in Fig. 2a, the accuracies of all methods decreased with the increase of batch.facScale since higher batch.facScale represented larger batch effects, i.e., higher annotating difficulty. Overall, our method consistently achieved stable and the best performance with the accuracies only slightly changed from 1.0 to 0.98 when increasing batch.facScale from 0.2 to 1.6. By comparison, scGCN, the second-best method, had significant drop in accuracies when batch.facScale was greater than 1.0, and a sharp drop from 0.93 to 0.76 when increasing batch.facScale from 1.4 to 1.6. The accuracies of SingleR and scPred were larger than 0.9 when the value of batch.facScale was less than 0.4, but their accuracies significantly dropped afterwards and were only 0.6 and 0.4, respectively when batch.facScale=1.6. SingleCellNet, CHETAH, and scmap, could achieve decent results at batch.facScale of 0.2 with accuracies of 0.87, 0.70, and 0.68, respectively, but they performed badly at batch.facScale of 1.2 with accuracies below 0.5. For two Seurat methods, Seurat-PCA was more sensitive from the batch.facScale value. Seurat-PCA had higher accuracies than Seurat-CCA at batch.facScale of <0.6, but lower accuracies at greater batch.facScale values. This is likely because Seurat-CCA overcorrected the batch effects at small batch.facScale values. By comparison, GraphCS always outperformed the competing methods in different magnitudes of batch effects. The superior performance showed that GraphCS could effectively reduce performance degradation brought by batch difference.

3.2 Performance on real datasets

We evaluated GraphCS on real datasets at two different levels: cross-platform and cross-species. For the cross-platform datasets, we tested on seven paired cross-platform datasets. As shown in Fig. 2b, the average accuracy of GraphCS (mean Acc=88%) was 4% higher than the second-ranked method SingleCellNet (mean Acc=84%) and consistently outperformed other competing methods. Seurat-PCA, SingleR, and Seurat-CCA ranked the 3rd, 4-th, and 5-th in terms of the average accuracy, respectively. In comparison to Seurat-PCA, Seurat-CCA did not benefit from aligning and integrating the datasets. CHETAH, scmap, and scGCN achieved similar average accuracy. Though scGCN took a similar technique to ours, the average accuracy of scGCN was lower than GraphCS. It is likely because the scGCN constructed graphs containing fewer edges (averagely one to two times lower than ours) and didn't fully utilized the advantages of graph neural network. ScPred achieved much lower performance than other methods. To highlight the comparison regarding specific cell types, we used the heatmap to show the accuracy of each cell type annotated by different methods on the mouse retina dataset. As shown in Fig. 2c and Figure S1. SingleCellNet, SingleR, and scmap incorrectly assigned most of bipolar cells. Seurat-CCA, Seurat-PCA, and CHETAH incorrectly assigned most of cones cells. In contrast, our method correctly discriminated most cell types. Additionally, we performed an experiment by using multiple reference datasets (Figure S2), and our model was shown to consistently outperform other methods.

For the cross-species datasets. We evaluated all methods on four paired cross-species datasets. We didn't include scPred since it raised exceptions on cross-species datasets. As shown in Fig. 2d, GraphCS achieved an average accuracy of 0.94, respectively 3% and 10% higher than those by the second-ranked method scGCN (0.91) and the third-ranked method scmap (0.84). The left methods are ordered as: SingleCellNet, Seurat-CCA, SingleR, Seruat-PCA, and CHETAH. Specifically, in the combination dataset with only seven T cell and 13 Schwann types, GraphCS could still annotate them accurately (Figure S3a). As shown in the Sankey diagram

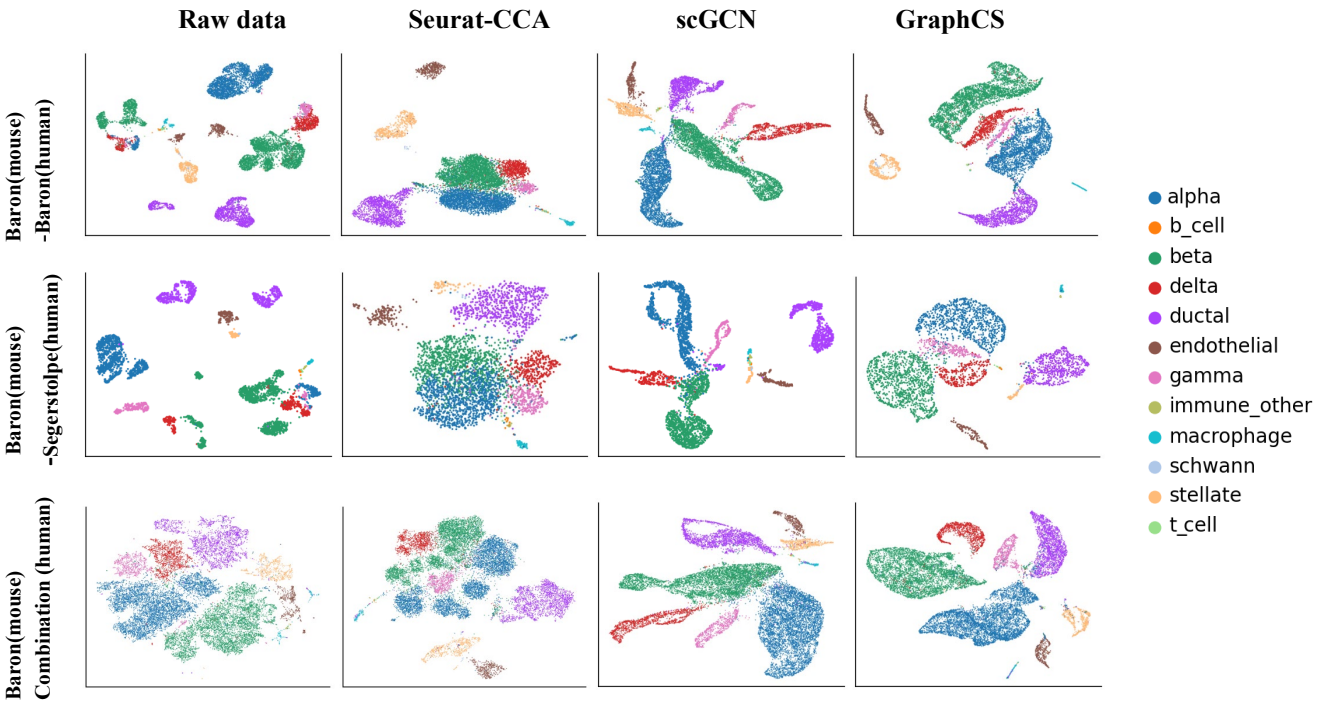

Fig. 3. UMAP visualization of three paired cross-species datasets, based on the aggregated data by different methods. The Baron mouse pancreas dataset is the reference for all query datasets. First row: the Baron human pancreas dataset as the query data. Second row: the Segerstolpe human pancreas dataset as the query data. Third row: we combined datasets Baron et al, Wang et al, Xin et al, Muraro et al, and Segerstolpe et al as the query data.

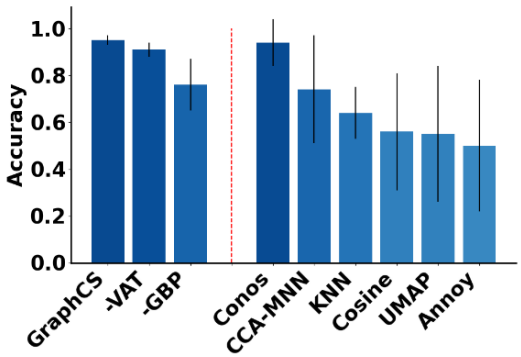

Fig. 4. The average classification accuracies and the standard deviations of GraphCS on the cross-species datasets by excluding VAT and GBP modules, and constructing the cell graph by different methods.

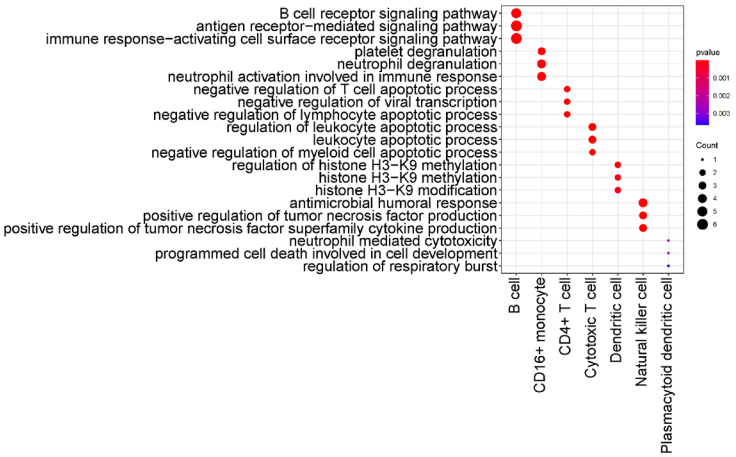

Fig. 5. The Enrichment pathways using the top 10 marker genes for each predicted cell type on the 10x_Drop-seq dataset.

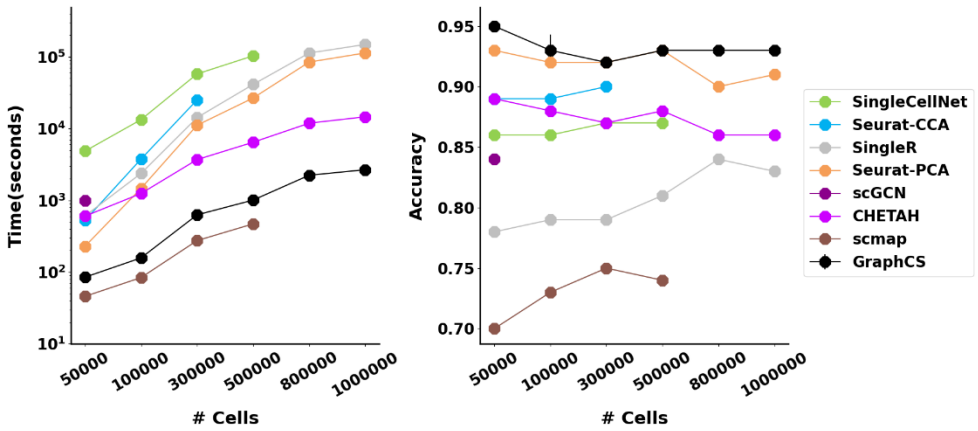

Fig. 6. Comparison of different methods for the running time (left) and cell-type classification accuracy (right) on variably sized datasets.

(Figure S3b), the much smaller number of cells in the reference data than the query data suggests the capability of our model in small reference data.

Finally, we visualized the cells in the aggregated reference-query data of cross-species in Fig. 3. We compared Seurat-CCA, scGCN, and GraphCS since they provided the aggregated data and took account of batch effects between datasets. Cells in the raw data were not separated well due to the substantial noise and batch effects. For example, in the dataset Baron (mouse)-Baron (human), beta cells were separated into two clusters, while alpha and delta cells gathered together. While Seurat and scGCN could discriminate most of the cell populations on all cross-species datasets, they couldn't explicitly distinguish a few cell types, such as beta and delta cells. By comparison, GraphCS could clearly separate most of the cell populations in all scenarios, indicating its ability to deal with strong batch effects between species.

3.3 Ablation Experiments

To investigate the contribution of each component in GraphCS, we performed the ablation experiments on cross-species datasets. As shown in Fig. 4, the removal of the GBP module caused a dramatic drop of 18% in the average accuracy, indicating that GBP module efficiently removed the batch effects of inter-datasets by propagating information among similar cells in the graph. The removal of VAT module caused small but significant drop (3.7%) in the average accuracy. In summary, the cooperation of the modules enabled a better annotating of the scRNA-seq data. The trend was similar on the cross-platform datasets (Figure S4).

We further evaluated the performance of GraphCS using seven different graph construction methods. As shown in Fig. 4, three methods with removal of batch effects (CCA-MNN, Conos, and our method to use BBKNN) achieved higher accuracies than those without removal (Cosine, UMAP, Annoy, and KNN) on the cross-species datasets. BBKNN and Conos achieved similar but better performance than CCA-MNN. It should be noted BBKNN was one to two orders of magnitude faster than Conos and CCA-MNN. The trend was similar on the cross-platform datasets (Figure S5).

3.4 Identifying cell-type important genes.

To interpret our model, we selected 10 most important genes for each predicted cell type according to the activation maximization method, and used these genes to decide the most enriched GO terms of biological processes through clusterProfiler. As seen in Fig. 5, when using the top 10 genes selected on the cross-platform dataset (10x_Drop-seq), most cell types were significantly enriched on their relevant Go terms. For the B

cells, 8 out of top 10 genes (MS4A1, CD79A, BANK1, HLA-DQA1, BLK, CD79B, POU2AF1, and IGHM) were included as maker gene in the PanglaoDB database [61]. These genes were enriched in the Go term "B cell receptor signaling pathway", consistent with the previous report that the B cell receptor signaling is essential for B cell survival and development although varied in different subpopulations and developmental stages [62]. We also listed top 10 genes for other predicted cell types in Table S1. These results suggested that the identified important genes were consistent with prior knowledge, demonstrating the reliability and interpretability of our model.

3.5 Running time evaluation

To evaluate the runtimes of all methods and their scalability with the increase in the number of cells, we sampled the mouse brain dataset in a stratified way (i.e., preserving population frequencies) to 6%, 12%, 36%, 60%, 96%, and 120% of the original number of 833,206 cells and selected the top 2000 highly variable genes as the input features. As shown in Fig. 6, dramatic differences of runtimes could be observed between these methods with increases in the number of cells. GraphCS was faster than all other methods except scmap. GraphCS showed a high scalability with about linear growth of runtimes with the number of cells: 1008s for 500K cells and 2669s for 1000K cells. This was 6 times faster than CHETAH, the next fastest method. Seurate-PCA was close to our method in speed for dataset with 50K cells, but the runtimes dramatically increased when the number of cells with 1000K, and turned 42 times slower than our method. Seurat-CCA was consistently slower than Seurat-PCA, and SingleCellNet was the slowest. While GraphCS was two times slower than scmap, GraphCS consistently achieved average accuracies of 20% higher than scmap. Additionally, under the default parameters, scmap couldn't process the dataset with >800K cells. Since scGCN constructed the graph based on CCA-MNN that only supported small datasets, it can only run on the small dataset with 50K cells and was 10 times slower than GraphCS. The results demonstrated that our model could be extended to large-scale datasets in linear time complexity.

4. Discussion

With the tremendous increase of scRNA-seq datasets, it is feasible to transfer well-defined labels of existing single-cell datasets to newly generated single-cell datasets. In this study, we proposed a robust and scalable graph-based artificial intelligence model, which enables training the well-labeled single-cell data to annotate new data through robust knowledge transferring. We have demonstrated that GraphCS achieves significant improvement compared to existing annotation methods in terms of performance and efficiency using the simulated, cross-platform and cross-species scRNA-seq datasets. Meanwhile, our model can be extended to large data in linear time complexity.

While several commonly used cell annotation algorithms, such as Seurat and SingleR, also possess knowledge transferring functionalities, our model achieved superior results in terms of both performance and efficiency. From a technical perspective, our model provides three major advantages. First, GraphCS removes the batch effects between datasets by propagating shared information among neighboring cells. Second, the generality of our model is improved by virtual adversarial training loss by incorporating data distribution information from unlabeled data. Third, GraphCS precomputes the feature propagation of graph neural network in a localized fashion by the graph bidirectional propagation algorithm to achieve scalability.

Although the superior results, GraphCS can be improved in several aspects. Firstly, our model ignores the relations between genes, which has been shown to improve the imputation of scRNA-seq data [27]. Secondly, the performance of our model is influenced by the constructed cell graph, and a high-quality graph can improve performance. Thus, the model may be useful for spatial transcriptomic data analysis [63, 64], where cells could be naturally connected through the provided spatial coordinates. Thirdly, our model highly relies on a high-quality reference dataset: we can't correctly identify cell types missing in single reference dataset. This problem might be solved by integrating a large number of well-labeled reference datasets since the model was proven able to identify cells simultaneously from two or more reference datasets.

AVAILABILITY

All source code used in our experiments have been deposited at https://github.com/biomed-AI/GraphCS.

Funding

This study has been supported by the National Key R&D Program of China (2020YFB0204803), National Natural Science Foundation of China (61772566), Guangdong Key Field R&D Plan (2019B020228001 and 2018B010109006), Introducing Innovative and Entrepreneurial Teams (2016ZT06D211), Guangzhou S&T Research Plan (202007030010).

References

[1] M. Baron *et al.*, "A single-cell transcriptomic map of the human and mouse pancreas reveals inter-and intra-cell population structure," *Cell systems,* vol. 3, no. 4, pp. 346-360. e4, 2016.
[2] S. V. Puram *et al.*, "Single-cell transcriptomic analysis of primary and metastatic tumor ecosystems in head and neck cancer," *Cell,* vol. 171, no. 7, pp. 1611-1624. e24, 2017.
[3] E. I. Athanasiadis, J. G. Botthof, H. Andres, L. Ferreira, P. Lio, and A. Cvejic, "Single-cell RNA-sequencing uncovers transcriptional states and fate decisions in haematopoiesis," *Nature communications,* vol. 8, no. 1, pp. 1-11, 2017.
[4] E. Azizi *et al.*, "Single-cell map of diverse immune phenotypes in the breast tumor microenvironment," *Cell,* vol. 174, no. 5, pp. 1293-1308. e36, 2018.
[5] D. A. Cusanovich *et al.*, "A single-cell atlas of in vivo mammalian chromatin accessibility," *Cell,* vol. 174, no. 5, pp. 1309-1324. e18, 2018.
[6] M. J. Muraro *et al.*, "A single-cell transcriptome atlas of the human pancreas," *Cell systems,* vol. 3, no. 4, pp. 385-394. e3, 2016.
[7] T. M. Consortium, "Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris," *Nature,* vol. 562, no. 7727, pp. 367-372, 2018.
[8] J. D. Buenrostro *et al.*, "Integrated single-cell analysis maps the continuous regulatory landscape of human hematopoietic differentiation," *Cell,* vol. 173, no. 6, pp. 1535-1548. e16, 2018.
[9] D. A. Jaitin *et al.*, "Massively parallel single-cell RNA-seq for marker-free decomposition of tissues into cell types," *Science,* vol. 343, no. 6172, pp. 776-779, 2014.
[10] T. M. Gierahn *et al.*, "Seq-Well: portable, low-cost RNA sequencing of single cells at high throughput," *Nature methods,* vol. 14, no. 4, pp. 395-398, 2017.
[11] D. Lahnemann *et al.*, "Eleven grand challenges in single-cell data science," *Genome Biol,* vol. 21, no. 1, p. 31, Feb 7 2020, doi: 10.1186/s13059-020-1926-6.
[12] M. D. Luecken and F. J. Theis, "Current best practices in single‐cell RNA‐seq analysis: a tutorial," *Molecular systems biology,* vol. 15, no. 6, p. e8746, 2019.
[13] A. Mezger *et al.*, "High-throughput chromatin accessibility profiling at single-cell resolution," *Nature communications,* vol. 9, no. 1, pp. 1-6, 2018.
[14] J. D. Buenrostro *et al.*, "Single-cell chromatin accessibility reveals principles of regulatory variation," *Nature,* vol. 523, no. 7561, pp. 486-490, 2015.
[15] V. Svensson, "Droplet scRNA-seq is not zero-inflated," *Nature Biotechnology,* vol. 38, no. 2, pp. 147-150, 2020.
[16] B. Vieth, C. Ziegenhain, S. Parekh, W. Enard, and I. Hellmann, "powsimR: power analysis for bulk and single cell RNA-seq experiments," *Bioinformatics,* vol. 33, no. 21, pp. 3486-3488, 2017.
[17] P. Brennecke *et al.*, "Accounting for technical noise in single-cell RNA-seq experiments," *Nature methods,* vol. 10, no. 11, p. 1093, 2013.
[18] A. T. Lun and J. C. Marioni, "Overcoming confounding plate effects in differential expression analyses of single-cell RNA-seq data," *Biostatistics,* vol. 18, no. 3, pp. 451-464, 2017.

[19] A. Regev *et al.*, "Science forum: the human cell atlas," *Elife,* vol. 6, p. e27041, 2017.

[20] N. Schaum *et al.*, "Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris: The Tabula Muris Consortium," *Nature,* vol. 562, no. 7727, p. 367, 2018.

[21] V. Y. Kiselev, A. Yiu, and M. Hemberg, "scmap: projection of single-cell RNA-seq data across data sets," *Nature methods,* vol. 15, no. 5, pp. 359-362, 2018.

[22] D. Aran *et al.*, "Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage," *Nature immunology,* vol. 20, no. 2, pp. 163-172, 2019.

[23] J. K. de Kanter, P. Lijnzaad, T. Candelli, T. Margaritis, and F. C. Holstege, "CHETAH: a selective, hierarchical cell type identification method for single-cell RNA sequencing," *Nucleic acids research,* vol. 47, no. 16, pp. e95-e95, 2019.

[24] J. Alquicira-Hernandez, A. Sathe, H. P. Ji, Q. Nguyen, and J. E. Powell, "scPred: accurate supervised method for cell-type classification from single-cell RNA-seq data," *Genome biology,* vol. 20, no. 1, pp. 1-17, 2019.

[25] Y. Tan and P. Cahan, "SingleCellNet: a computational tool to classify single cell RNA-Seq data across platforms and across species," *Cell systems,* vol. 9, no. 2, pp. 207-213. e2, 2019.

[26] A. Butler, P. Hoffman, P. Smibert, E. Papalexi, and R. Satija, "Integrating single-cell transcriptomic data across different conditions, technologies, and species," *Nat Biotechnol,* vol. 36, no. 5, pp. 411-420, Jun 2018, doi: 10.1038/nbt.4096.

[27] J. Rao, X. Zhou, Y. Lu, H. Zhao, and Y. Yang, "Imputing single-cell RNA-seq data by combining graph convolution and autoencoder neural networks," *Iscience,* vol. 24, no. 5, p. 102393, 2021.

[28] Y. Zeng, X. Zhou, J. Rao, Y. Lu, and Y. Yang, "Accurately Clustering Single-cell RNA-seq data by Capturing Structural Relations between Cells through Graph Convolutional Network," in *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2020: IEEE, pp. 519-522.

[29] J. Wang *et al.*, "scGNN is a novel graph neural network framework for single-cell RNA-Seq analyses," *Nature communications,* vol. 12, no. 1, pp. 1-11, 2021.

[30] Q. Song, J. Su, and W. Zhang, "scGCN: a Graph Convolutional Networks Algorithm for Knowledge Transfer in Single Cell Omics," *bioRxiv,* 2020.

[31] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907,* 2016.

[32] J. Cao *et al.*, "The single-cell transcriptional landscape of mammalian organogenesis," *Nature,* vol. 566, no. 7745, pp. 496-502, 2019.

[33] P. Datlinger, A. F. Rendeiro, T. Boenke, T. Krausgruber, D. Barreca, and C. Bock, "Ultra-high throughput single-cell RNA sequencing by combinatorial fluidic indexing," *BioRxiv,* 2019.

[34] M. Chen *et al.*, "Scalable Graph Neural Networks via Bidirectional Propagation," *arXiv preprint arXiv:2010.15421,* 2020.

[35] P. Li, E. Chien, and O. Milenkovic, "Optimizing generalized pagerank methods for seed-expansion community detection," *arXiv preprint arXiv:1905.10881,* 2019.

[36] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426,* 2018.

[37] A. Sehanobish, N. G. Ravindra, and D. van Dijk, "Self-supervised edge features for improved Graph Neural Network training," *arXiv preprint arXiv:2007.04777,* 2020.

[38] B. Wang, J. Zhu, E. Pierson, D. Ramazzotti, and S. Batzoglou, "Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning," *Nature methods,* vol. 14, no. 4, pp. 414-416, 2017.

[39] B. Hie, B. Bryson, and B. Berger, "Efficient integration of heterogeneous single-cell transcriptomes using Scanorama," *Nature biotechnology,* vol. 37, no. 6, pp. 685-691, 2019.

[40] B. Szubert, J. E. Cole, C. Monaco, and I. Drozdov, "Structure-preserving visualisation of high dimensional single-cell datasets," *Scientific reports,* vol. 9, no. 1, pp. 1-10, 2019.

[41] S. Dudoit, J. Fridlyand, and T. P. Speed, "Comparison of discrimination methods for the classification of tumors using gene expression data," *Journal of the American statistical association,* vol. 97, no. 457, pp. 77-87, 2002.

[42] L. Haghverdi, A. T. L. Lun, M. D. Morgan, and J. C. Marioni, "Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors," *Nat Biotechnol,* vol. 36, no. 5, pp. 421-427, Jun 2018, doi: 10.1038/nbt.4091.

[43] N. Barkas *et al.*, "Joint analysis of heterogeneous single-cell RNA-seq dataset collections," *Nature methods,* vol. 16, no. 8, pp. 695-698, 2019.

[44] K. Polański, M. D. Young, Z. Miao, K. B. Meyer, S. A. Teichmann, and J.-E. Park, "BBKNN: fast batch alignment of single cell transcriptomes," *Bioinformatics,* vol. 36, no. 3, pp. 964-965, 2020.

[45] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE transactions on pattern analysis and machine intelligence,* vol. 41, no. 8, pp. 1979-1993, 2018.

[46] L. Zappia, B. Phipson, and A. Oshlack, "Splatter: simulation of single-cell RNA sequencing data," *Genome Biol,* vol. 18, no. 1, p. 174, Sep 12 2017, doi: 10.1186/s13059-017-1305-0.

[47] Å. Segerstolpe *et al.*, "Single-cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes," *Cell metabolism,* vol. 24, no. 4, pp. 593-607, 2016.

[48] Y. J. Wang *et al.*, "Single-cell transcriptomics of the human endocrine pancreas," *Diabetes,* vol. 65, no. 10, pp. 3028-3038, 2016.

[49] Y. Xin *et al.*, "RNA sequencing of single human islet cells reveals type 2 diabetes genes," *Cell metabolism,* vol. 24, no. 4, pp. 608-615, 2016.

[50] J. Ding *et al.*, "Systematic comparative analysis of single cell RNA-sequencing methods," *BioRxiv,* p. 632216, 2019.

[51] H. T. N. Tran *et al.*, "A benchmark of batch-effect correction methods for single-cell RNA sequencing data," *Genome biology,* vol. 21, no. 1, pp. 1-32, 2020.

[52] E. Z. Macosko *et al.*, "Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets," *Cell,* vol. 161, no. 5, pp. 1202-1214, 2015.

[53] K. Shekhar *et al.*, "Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics," *Cell,* vol. 166, no. 5, pp. 1308-1323. e30, 2016.

[54] B. Li, V. Ruotti, R. M. Stewart, J. A. Thomson, and C. N. Dewey, "RNA-Seq gene expression estimation with read mapping uncertainty," *Bioinformatics,* vol. 26, no. 4, pp. 493-500, 2010.

[55] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," *arXiv preprint arXiv:1810.05997,* 2018.

[56] Y. Ouali, C. Hudelot, and M. Tami, "An Overview of Deep Semi-Supervised Learning," *arXiv preprint arXiv:2006.05278,* 2020.

[57] M. T. Ribeiro, S. Singh, and C. Guestrin, "" Why should i trust you?" Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135-1144.

[58] S. Ge, H. Wang, A. Alavi, E. Xing, and Z. Bar-Joseph, "Supervised adversarial alignment of single-cell RNA-seq data," *Journal of Computational Biology,* 2021.

[59] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034,* 2013.

[60] G. Yu, L.-G. Wang, Y. Han, and Q.-Y. He, "clusterProfiler: an R package for comparing biological themes among gene clusters," *Omics: a journal of integrative biology,* vol. 16, no. 5, pp. 284-287, 2012.

[61] O. Franzén, L.-M. Gan, and J. L. Björkegren, "PanglaoDB: a web server for exploration of mouse and human single-cell RNA sequencing data," *Database,* vol. 2019, 2019.

[62] W. Liu, P. Tolar, W. Song, and T. J. Kim, "BCR signaling and B cell activation," *Frontiers in immunology,* vol. 11, p. 45, 2020.

[63] N. Crosetto, M. Bienko, and A. Van Oudenaarden, "Spatially resolved transcriptomics and beyond," *Nature Reviews Genetics,* vol. 16, no. 1, pp. 57-66, 2015.

[64] X. Qian *et al.*, "Probabilistic cell typing enables fine mapping of closely related cell types in situ," *Nature methods,* vol. 17, no. 1, pp. 101-106, 2020.