bioRxiv, TBD, pp. 1–8

doi: TBD
Paper

PAPER

Consensus Label Propagation with Graph Convolutional Networks for Single-Cell RNA Sequencing Cell Type Annotation

Daniel P Lewinsohn,[1,2,*] Katinka A Vigh-Conrad,[1] Donald F Conrad[1] and Cory B Scott[2]

[1]Division of Genetics, Oregon National Primate Center, Oregon Health and Science University, Oregon, USA and [2]Department of Mathematics and Computer Science, Colorado College, Colorado, USA
[*]Corresponding author. d_lewinsohn@coloradocollege.edu

Abstract

Motivation: Single-cell RNA sequencing (scRNA-seq) data, annotated by cell type, are useful in a variety of downstream biological applications, such as profiling gene expression at the single-cell level. However, manually assigning these annotations with known marker genes is both time-consuming and subjective.
Results: We present a Graph Convolutional Network (GCN) based approach to automate the annotation process. Our process builds upon existing labeling approaches, using state-of-the-art tools to find cells with highly confident assignments through consensus and spreading these confident labels with a semi-supervised GCN. Using simulated data and two scRNA-seq data sets from different tissues, we show that our method improves accuracy over a simple consensus algorithm and the average of the underlying tools. We then compare our method to a non-parametric neighbor majority approach, showing comparable results. We also demonstrate that our GCN method allows for feature interpretation, identifying important genes for cell type classification. We present our completed pipeline, written in Pytorch, as an end-to-end tool for automating and interpreting the classification of scRNA-seq data.
Availability: Our code for conducting the experiments in this paper and using our model is available at `https://github.com/lewinsohndp/scSHARP`
Contact: d_lewinsohn@coloradocollege.edu

Key words: GCN, scRNA-seq, consensus, cell type labeling

Introduction

Single-cell RNA sequencing (scRNA-seq) measures the RNA from each gene present in an individual cell, serving as a proxy for gene expression. High-quality labels of cell type (based on the transcriptional profile produced by scRNA-seq) have proven valuable for characterizing gene expression of cells, and for discovering cell types and genetic drivers of disease. Traditionally, these labels are produced by unsupervised clustering followed by labeling clusters with known marker genes. However, unsupervised clustering is limited by issues such as the size of scRNA-seq data sets as well as subjectivity in reclustering and biological interpretation of clusters [14].

The limitations of traditional cell type annotation methods have necessitated the development of automated methods for cell labeling. Three main categories of tools have emerged: marker gene based, correlation based, and supervised classification based [16]. Marker based approaches employ known marker genes for labeling, while correlation and supervised learning based approaches require manually labeled scRNA-seq data sets with the cell types of interest. Within these broad categories, the performance of individual tools varies widely across data sets [3]. As a result, using the consensus of multiple classification tools could yield higher accuracy. However, there currently exist no tools for researchers to easily apply multiple classification algorithms to their scRNA-seq data. Even once different predictions have been collected, deciding how to handle cells with ambiguous predictions between tools remains an open question.

We address these issues in two ways. First, we provide a pipeline for annotation of scRNA-seq data with multiple state-of-the-art annotation algorithms. Second, we implement and test a semi-supervised Graph Convolutional Network (GCN) as a mechanism to propagate labels from confidently labeled cells to unconfidently labeled cells. We compare our method to several state-of-the-art cell classification tools, a simple max consensus, and a non-parametric neighbor majority approach. We show our method improves overall classification accuracy (and specifically, classification accuracy on unconfidently

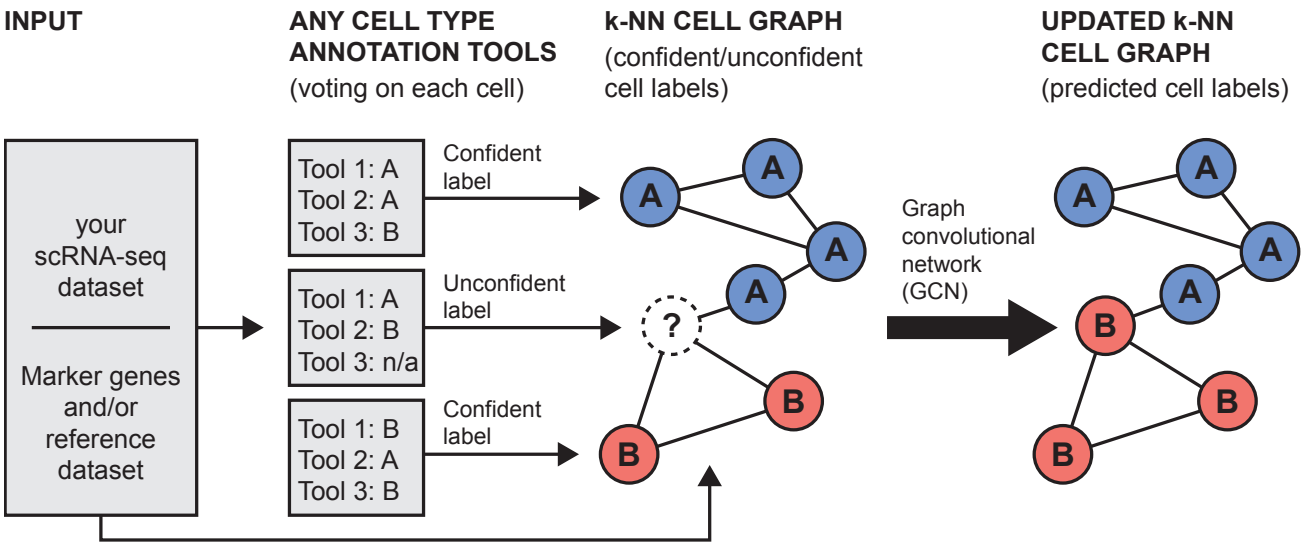

Fig. 1. Starting from an scRNA-seq data set, a user can apply any number of tools which classify cell types (for example, classification from a pre-labeled reference data set or from reference genetic markers). If a majority of these underlying tools assign the same label to a cell, we say the ensemble is confident in this label. Our GCN learns to propagate labels from confidently labeled cells to the rest of the cells ("unconfident cells") via message passing in a K-nearest neighbor graph.

labeled cells) compared to component tools and the simple consensus of the tools. We also demonstrate our approach shows comparable performance with the non-parametric neighbor majority approach.

Following accurate classification, an important aspect of model usage within the biological sciences is interpretability. There has been significant development around neural network interpretation, resulting in a variety of approaches for assigning attribution scores to input features. Gradient-based attribution methods for neural networks have proven valuable given their efficiency and ease of implementation [6]. Importantly, trained neural network models can be queried for important features for classification that are not otherwise obvious. One effective algorithm for gradient-based interpretation is DeepLIFT. This method decomposes the output prediction of a neural network on a specific feature by backpropagating through the network. DeepLIFT tries to explain to the difference between the output and a neutral reference output, generated by a neutral reference input [17].

We demonstrate the use of DeepLIFT [17] as an effective interpretation tool for our GCN model, allowing researchers insight into classification decisions and important cell type gene markers. The ability to apply DeepLIFT to our model is a valuable feature compared to other baseline methods used in this study. Specifically, we show interpretation of our trained model identifies differentially expressed genes with biological significance not explicitly used as markers. Furthermore, we show adding these highly-attributed genes as markers for marker-based tools can improve accuracy.

Methods

Our Model

Selected Component Methods

Our method relies on picking confident labels for a subset of cells in a data set. To do this, our pipeline currently includes five different state-of-the-art annotation methods: SCINA [21], ScType [12], scSorter [10], SingleR [7], and scPred [5]. We briefly describe each of these tools and refer the reader to these tools' original papers for more details. ScType employs a clustering-based approach that inputs the scRNA-seq cells by genes matrix along with cell type gene markers, and outputs predictions [12]. scSorter is another clustering-based approach that uses the scRNA-seq cells by genes matrix along with cell type gene markers. This method recognizes that over-expression of certain marker genes is not present in populations of many cell types and attempts to address this problem [10]. SCINA is another state-of-the-art approach that inputs the same information as ScType and scSorter, with the added benefit of being much faster. SCINA uses an expectation-maximization algorithm to assign labels [21]. SingleR requires both the input scRNA-seq cells by genes matrix and a labeled reference cells by genes matrix. This method uses correlation between the reference and query sets to extend labels [7]. scPred requires the same information as SingleR. This method uses feature space reduction to pull out important cell type features and then a machine learning probability-based prediction algorithm [5]. We note that our pipeline could be applied to a broader set of tools.

Picking Confident Labels via Consensus

Gene counts are initially filtered to first remove cells expressed in fewer than 200 genes and then filtered to remove genes expressed in fewer than 3 cells. Tools make predictions on further preprocessed gene counts as specified by the given tool, returning an n by t matrix R where t is the number of annotation tools and $R_{i,j}$ is the prediction from tool j for cell i. We then one-hot encode the predictions from each column of R resulting in t matrices of size n by the number of cell types. We then add these matrices to construct a new matrix E where the rows correspond to cells and columns correspond to the amount of votes for each cell type. Finally, we create confident labels $c \in \mathbb{R}^n$ where c_n corresponds to the predicted cell type for cell n if a majority of tools agreed and -1 if not. Selecting confident labels entails setting a threshold of

confidence which we set at .51 for a simple majority. For cell x_i we find $m_i = max(E_i/sum(E_i))$ and assign label -1 if $m_i < .51$ and the label of the corresponding column of m_i if $m_i >= .51$. This results in $c \in \mathbb{R}^n$ containing the confidently labeled and unconfidently labeled cells.

Semi-supervised GCN

Our data is presented to our model as a preprocessed scRNA-seq matrix for n cells, denoted by $X = \{x_1, ..., x_n\} \subseteq \mathbb{R}^F$ where F is the number of features for each cell. We use Principal Components Analysis (PCA) to project the cell-by-gene matrix X into a 500D vector space. To train our GCN, we draw random minibatches of size b at a time. We construct G as the k-nearest neighbor (k-NN) graph of the cells included in each minibatch with k neighbors, including self-loops. We experimented with different values of k, finding lower neighbors and batches performed better. We input G into the following GCN architecture.

We construct a GCN with l EdgeConv [19] layers with SiLU activation function and summation aggregation. A final linear layer projects node embeddings into label space (whose dimension is the number of cell types in our data set). For architecture details see Appendix A. We train our GCN for 150 epochs, with the Adam optimizer [4] at a learning rate of 0.0001. Our training loss is Cross-Entropy loss on the set of confidently labeled cells. Nearest neighbors are generated separately for each batch of each epoch.

Interpretation with DeepLIFT

We employ DeepLIFT [17] with the Rescale rule as implemented by Captum [15]. We use the same hyperparameters batch size b, neighbors k, number of message passing steps l, and final embedding layer size e as used during training. Please refer to the Experiment Settings section for how these hyperparameters are chosen. DeepLIFT requires a reference input to determine the reference activation of a neuron. We use a reference count matrix of all zeros and the same size as our scRNA-seq count matrix X. We then use our tool's predictions for each cell type as its target neuron for calculating the gradient. This results in attribution matrix A with the same size as X. We use A to calculate the mean feature attribution scores for each gene, for all cells of each type. We then take the absolute value of these attributions to compute a general importance score per feature/gene. It is important to note we are not interpreting negative or positive importance for each feature, just overall importance. Finally, we scale each cell type to have unit variance to allow comparison of gene importances across cell type. Calculating these attribution scores is only possible for a differentiable model, like our GCN – not possible for any of the underlying tools. We note here that the pipeline we analyzed with DeepLIFT included the PCA step, resulting in attribution scores per gene for each cell.

Non-parametric Neighbor Majority Approach

As an additional baseline to test our GCN model we implemented a non-parametric neighbor majority approach. This is similar to the message passing step in our GCN model, with the difference that this method does not use a neural network to encode/decode messages. This method operates on the 500D vectors produced as the principal components of the preprocessed gene expression matrices for each data set. We first construct a k nearest neighbors graph in euclidean distance, using a batch size of 1000. During each round of message passing, the unconfidently labeled nodes' labels are updated as the majority label of their k nearest neighbors in euclidean distance (only considering those neighbors who have been labeled thus far). We test three strategies in hyperparameter optimization for convergence of this method:

- one round of label propagation;
- iterating until less than 5 percent of labels change between epochs; and
- iterating until all cells are labeled or 50 epochs have gone by.

We show accuracy percents across a variety of k values and convergence strategies in Appendix D.

Data Sets

Preprocessing Data

The initial input to our pipeline is a scRNA-seq count matrix X where X_{ng} corresponds to observed gene counts for gene g in cell n. Cells expressing < 200 genes and genes expressed in < 3 cells are removed, and X is row-normalized according to $x_i = log(1+((x_i * 10000)/\Sigma x_i)$. Both of these steps are common scRNA-seq preprocessing steps [11][8]. Finally, we use Principal Components Analysis (PCA) to project X down to 500 features per cell.

Prototyping Data

Simulated data with a different seed than the test simulated data was used for developing our model. An added subset of the PBMC data not included in the final test was also used. We approached this task as a transductive semi-supervised graph learning task where each scRNA-seq data set is its own graph. However, no ground truth labels are provided for the learning task. We did not run our model on our four test data sets until architecture details were finalized. Additionally, only confidently labeled nodes are used for hyperparameter optimization in these test sets, meaning no ground truth labels are involved. It is also important to note no trained model weights carried over from the prototyping data sets to the testing data sets.

Simulated Data

We generated our simulated data sets with Splatter [20] and parameters estimated from 4000 Pan T Cells from a healthy donor [1]. Each simulated data set contains 1000 cells, evenly split between four cell types with different transcriptomic profiles. To demonstrate that our pipeline can also incorporate reference-based tools (like SingleR and scPred), we also generated 1000 reference cells for each data set with the same gene profiles (separated from the original data with a batch.facScale of 0.5 to simulate batch effects). Simulated data sets vary by the de.facScale parameter which determines the magnitude of differential expression in differentially expressed genes between cell type groups. Five markers were selected randomly from the top ten differentially expressed genes from each cell type. The 0.7 de.facScale and 0.8 de.facScale simulated data sets had 156 and 85 unconfidently labeled cells respectively.

Real Data

It is not usually feasible to acquire ground truth labels for scRNA-seq data. An alternative gold standard is Fluorescence-activated Cell Sorting (FACS), which pre-sorts cells by markers

Table 1. Accuracy percent scores for all data sets for both all cells and "unconfident cells": cells for which the underlying methods did not have consensus. GCN accuracies are the mean ± standard deviation of accuracies from five randomly initialized trials.

	Simulation 0.7		Simulation 0.8		Testis		PBMC	
Method	All	Unconf.	All	Unconf.	All	Unconf.	All	Unconf.
Ours (GCN)	**90.0 ± .61**	66.0 ± 3.9	**96.1 ± .22**	**83.3 ± 2.6**	86.2 ± .12	80.9 ± 1.8	**93.1 ± .05**	70.6 ± 1.5
Max Consensus	86.4	42.9	91.2	25.9	80.8	0.0	91.2	6.8
Tool Avg.	69.3 ± 14	33.1 ± 32	76.7 ± 12	34.6 ± 28	72.1 ± 16	21.7 ± 36	75.0 ± 6.1	36.9 ± 34
ScType	64.8	13.5	79.9	29.4	84.8	63.0	85.5	**81.8**
scSorter	85.8	51.3	88.7	38.8	77.5	2.2	71.5	65.9
SCINA	53.7	7.7	58.7	2.4	53.9	0.0	73.3	6.2
SingleR	83.1	**80.1**	85.2	77.6	*NA*	*NA*	70.3	13.4
scPred	59.2	12.8	71.0	24.7	*NA*	*NA*	74.4	17.1
Non-Parametric	82.3	16.7	90.3	15.3	**86.3**	**82.6**	**93.1**	70.8

prior to conduction of scRNA-seq [18]. We test our model on two FACS-labeled data sets.

First, we use a scRNA-seq data set generated from mouse testis cells [13]. This data contains three cell types: 292 Spermatogonia, 244 Spermatocytes, and 156 Spermatids after filtering. Cell type markers were selected from relevant literature [9] and no reference data set was used for this data. There were 46 unconfidently labeled cells after prediction tool voting on the data set.

Second, we use an scRNA-seq data set generated from Human peripheral blood mononuclear cells (PBMCs) [22]. This data contains ten cell types, however, we removed cell types not purely sorted by FACS, combined CD4+ T cells, and combined CD8+ T Cells. This resulted in five cell types: 9,106 B Cells, 2,341 Monocytes, 7,572 Natural killer (NK) Cells, 38,006 CD4+ T Cells, and 19,856 CD8+ T Cells. We used the same markers as scSorter [10] and used the 10X PBMC 3k data set as a reference as in [2]. This data set contained 2,310 unconfidently labeled cells.

Results

Accuracy on Test Sets

Experiment Settings

For the simulated and testis data sets, 20 percent of confidently labeled cells were masked and held out as a validation set. We performed a hyperparameter optimization search (see Appendix A for details) for options of batch size b, neighbors k, layers l, and embedding layer size e, selecting the GCN architecture with the highest validation accuracy. For the PBMC data set, five percent of the data set was used for this same hyperparameter optimization process as using the entire data set was computationally intractable. Each GCN used EdgeConv feature propagation between each node and its k closest neighbors, with distance determined dynamically between node features (including the PCA features at the first layer). Five random initializations of the optimal model were then trained for 150 epochs as described above, and mean accuracy was recorded. We use max consensus, the non-parametric neighbor majority approach, and other tool accuracies as baselines for our model. Max consensus simply chooses the cell type with the most votes. In the event of a tie, this method returns "unknown". The k hyperparameter and convergence method for the non-parametric neighbor majority approach was chosen with the same validation set used for GCN hyperparameter optimization. We report total accuracy and unconfident cell accuracy for each data set. "Unconfident cell accuracy" refers to the accuracy on only the cells where the underlying tools did not find consensus.

Simulated Data Sets

For both simulated data sets, the optimal model has batch size 20, 2 nearest neighbors, and 2 EdgeConv layers. For the simulation with 0.7 de.facScale, 25-dimensional embedding space was optimal, whereas for the simulation with 0.8 de.facScale the optimal value was 40. Additionally, 15.6% and 8.5% of cells were unconfidently labeled in the 0.7 de.facScale and 0.8 de.facScale data sets respectively. Table 1 shows our model outperforms all other methods for total accuracy and slightly under performs SingleR for unconfident cell accuracy on the 0.7 de.facScale data.

The non-parametric neighbor majority approach proves extremely ineffective in propagating confident labels to unconfidently labeled cells in both simulated data sets. This could be due to the transcriptional similarity of all four cell types in the simulated data sets. All four cell types in the simulated data sets are equally different. This suggests that in data sets with many similar cell types, this approach would fail.

Testis Data Set

Only marker based prediction tools were used for this data set as no labeled reference was easily available, and we wanted to test our method without reference based tools. The optimal model for this data set used batch size 20, 2 nearest neighbors, 2 EdgeConv layers, and embedding layer size of 25. Importantly, 6.6% of cells were unconfidently labeled by the component tools. Table 1 shows accuracy results, demonstrating our GCN model outperforms all other methods for both total and unconfident cell accuracy, except for the non-parametric approach. The accuracy of the non-parametric approach is within the standard deviation of our model for both the total and unconfident accuracies.

These results suggest both our GCN model and the non-parametric neighbor majority effectively propagate consensus labels to unconfidently labeled cells in this testis scRNA-seq data. This is clear given the improved accuracy over all component methods and significant improvement of our model over the tool average accuracy by 14.1%. Our model therefore demonstrates improved accuracy over component methods and comparable accuracy to the non-parametric neighbor majority approach, with the major advantage of being interpretable.

PBMC Data Set

The optimal model for this data set used batch size 50, 2 nearest neighbors, 2 EdgeConv layers, and embedding layer size of 25. 3.0% of cells were unconfidently labeled by the component tools. Table 1 shows accuracy results. For accuracy on unconfident cells, the GCN model places third behind ScType and the non-parametric approach. For overall accuracy, our model ties with the non-parametric approach to the third significant digit. Additionally, the accuracy of the non-parametric approach is well within the standard deviation of our model accuracy on the unconfident set.

These accuracy results highlight the ability of our GCN model and the non-parametric neighbor majority approach to propagate confident labels in the PBMC data. We again show a clear improvement over all component methods, demonstrating the value of this approach in labeling scRNA-seq cells.

Feature Interpretation

Figure 2A shows the five most highly attributed genes by DeepLIFT for each cell type in the testis data set. Interestingly, all of these top genes have uniquely high attribution in their important cell type. The highly attributed genes for a cell type also have relatively high gene expression in that cell type as indicated by Figure 2B. We also observe high expression of Spermatocyte genes in Spermatid cells. It is important to note DeepLIFT interpretation indicates importance of both input marker genes and genes not explicitly given to marker based tools. For example, DeepLIFT indicates genes Ncl, Ldhc, and Tnp1 that are differentially expressed in those cell types, but not explicitly included as marker genes. Additionally, Tsga8 which was used as a marker gene, appears highly attributed to DeepLIFT in Spermatids.

Figure 2C shows the DeepLIFT attribution scores for the top five highly-attributed genes by cell type for the PBMC data. Figure 2D shows actual gene expression of these highly attributed genes in the data set. For B Cells, Monocytes, and NK Cells we see a clear connection between the genes picked out as important by DeepLIFT and the genes expressed by those cell types. However, for CD4 and CD8 T Cells, the expression is not clearly higher for all genes. Importantly, we do observe CD8B as the most important gene for CD8 T Cell classification, a key marker for the cell type. We also observe CD3E (another important marker for all T Cells) as an important gene for both sub types of T Cells. One potential reason for less informative DeepLIFT scores for CD4 and CD8 T Cells is that the GCN often misclassifies CD8 T Cells as CD4 T Cells. This is likely due to the large heterogeneity and general transcriptional similarity of T Cell subtypes. Finally, DeepLIFT again reveals both inputted marker genes and novel genes. Specifically, DeepLIFT highly attributes CD74, FTH1, GNLY, and more genes that are differentially expressed, but not originally included as marker genes. DeepLIFT also highly attributes original marker genes such as CD79A, CD8B, and CD3E.

Overall, DeepLIFT interpretation proves effective in both the testis and PBMC data sets, revealing novel genes not originally included as markers. Importantly, the GCN is the only one of our tested methods that can be interpreted using DeepLIFT. This adds a valuable feature for researchers to both further understand classification decisions and get insight into novel cell type markers.

Table 2. Change between accuracy percent before and after adding highly attributed genes as markers in testis data set. Accuracy difference is recorded for each cell type: Spermatogonia (SPG), Spermatocytes (SPC), and Spermatids (SPD).

Method	SPG	SPC	SPD
ScType	0.00	0.00	0.00
scSorter	4.11	12.71	2.56
SCINA	7.53	2.87	33.97

Table 3. Change between accuracy percent before and after adding highly attributed genes as markers in PBMC data set. Accuracy difference is recorded for the following cell types: B Cells, Monocytes, Natural Killer Cells (NK Cells), CD4 T Cells (CD4), and CD8 T Cells (CD8)

Method	B Cells	Monocytes	NK Cells	CD4	CD8
ScType	0.00	-1.02	0.00	0.00	0.00
scSorter	0.43	4.44	15.73	33.47	-8.28
SCINA	5.15	41.44	39.54	-68.21	24.00

Highly Attributed Genes Improve Marker Based Methods

We added the top three highly attributed genes from DeepLIFT for each cell type not already included as a marker gene to the list of marker genes. We then re-ran the marker based tools ScType, scSorter, and SCINA with these markers on the testis and PBMC data sets to compare tool accuracy. This experiment shows the value of highly attributed DeepLIFT genes in assisting future classification.

Table 2 shows the change in accuracy before and after adding highly attributed genes in the testis data set. Both scSorter and SCINA show significant improvement in accuracy with the addition of highly attributed DeepLIFT genes. scSorter shows large in improvement in Spermatocytes while SCINA shows extreme improvement in Spermatids, demonstrating the effectiveness of using DeepLIFT attributed genes across different cell types and tools. ScType accuracy remains the same across all cell types. Although ScType does not improve, it is important to note accuracy does not go down with the addition of these new marker genes.

Table 3 shows the change in accuracy before and after adding highly attributed genes in the PBMC data set. ScType accuracy remains the same for B Cells, NK Cells, CD4 T Cells, and CD8 T Cells. For Monocytes, ScType accuracy slightly decreases. scSorter accuracy increases in B Cells, Monocytes, NK Cells, and CD4 T Cells. In CD8 T Cells, accuracy decreases. SCINA accuracy increases in all cell types except CD4 T Cells, showing large increases in Monocytes, NK Cells, and CD8 T Cells. In general, we observe increased or maintained improvement in B Cells, Monocytes, and NK Cells, indicating the effective use of DeepLIFT attributed genes as markers in these cell types. However, both T Cell subtypes show varying performance, with decreases in accuracy in both scSorter and SCINA. We believe this is due to the inaccuracy of our model in distinguishing these T Cell subtypes as previously discussed. This suggests using DeepLIFT attributed genes falls short in cell types with poor model accuracy. We suggest further validation of all genes with direct expression of genes in the scRNA-seq data to confirm new markers.

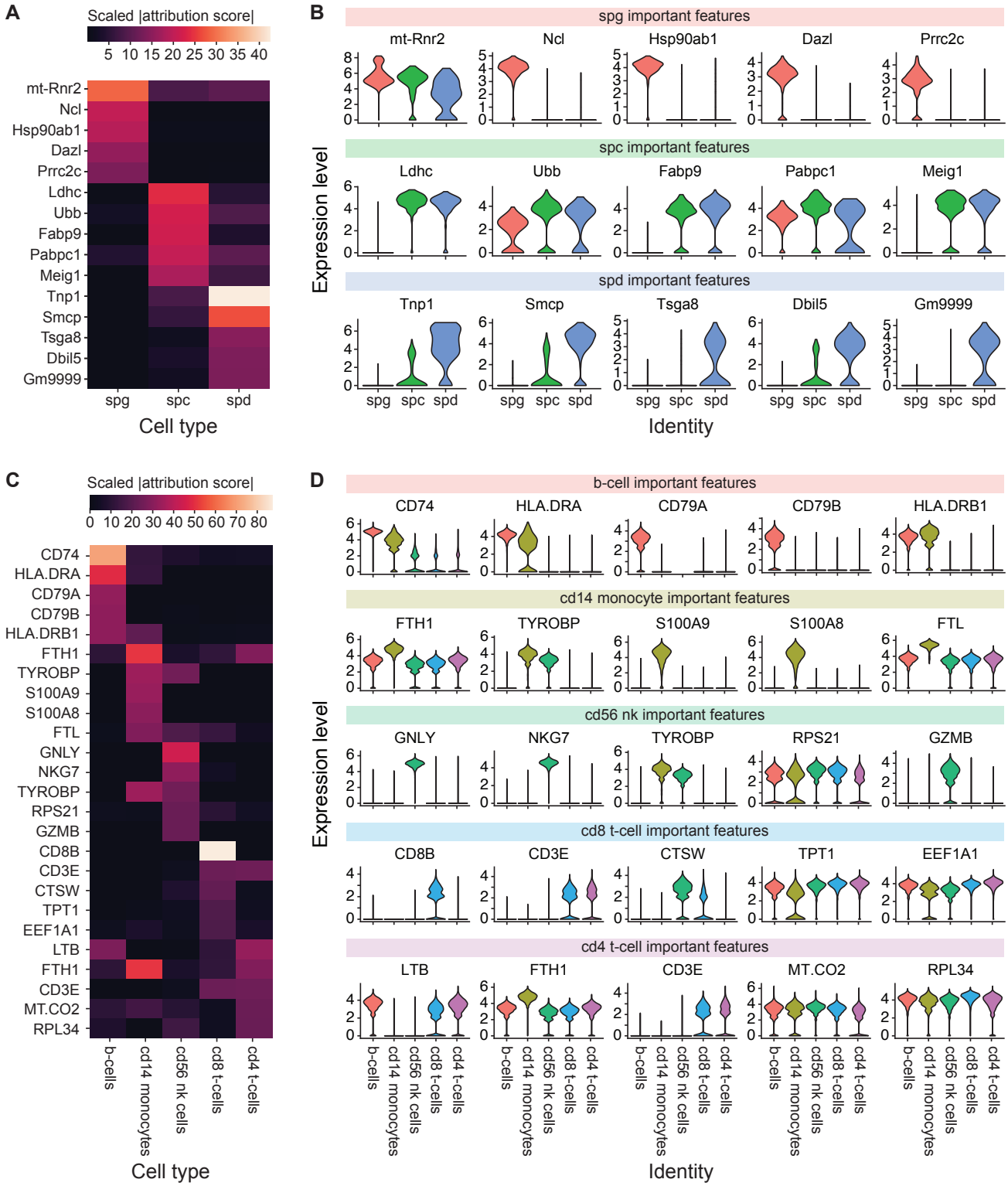

Fig. 2. Heatmap of DeepLIFT attribution scores after absolute value and scaling by cell type for top five most important features by predicted cell type and violin plot of log normalized expression for each gene by predicted cell type. a. Attribution heatmap for testis data set. b. Expression plots for testis data set by predicted cell type. c. Attribution heatmap for PBMC data set. d. Expression plots for PBMC data by predicted cell type.

Discussion

scRNA-seq cell type annotation remains an open and essential problem in bioinformatics. Accurate cell type labels are critical for downstream analysis and robust hypothesis testing with single-cell data. In this work we propose a novel framework for scRNA-seq cell type annotation that improves upon existing

approaches. Building upon existing annotation tools, we implement an EdgeConv based GCN model to propagate consensus based confident labels to the remaining unlabeled cells. We show an improvement in accuracy over a baseline max consensus algorithm and the average tool accuracy. We also show similar, if not improved, performance over a non-parametric neighbor majority propagation approach. We then demonstrate the ability to identify important genes for classification via model interpretation with DeepLIFT. The model interpretation is especially valuable for researchers as it has the potential to uncover novel gene markers and provide insight into the model's decisions.

We provide evidence for the accuracy of our GCN approach through comparison to state-of-the-art methods ScType, scSorter, SCINA, SingleR, and scPred on a variety of data sets, both simulated and real, including both Mouse and Human tissues. Our method improves accuracy in all data sets over all of these methods, providing strong evidence for the value of our method. A key takeaway from our research is the value of using a consensus of multiple annotation tools to improve general classification accuracy in scRNA-seq data.

We further provide evidence for the interpretability of our trained model with DeepLIFT. Highly attributed DeepLIFT genes include both given markers and novel genes that are differentially expressed in the actual data. This interpretation method is only possible for our method out of all compared methods, and provides a valuable tool for researchers when analyzing their scRNA-seq data.

One interesting aspect of our research is the similarity in classification accuracy of the non-parametric neighbor majority approach and our GCN approach for both real data sets. This suggests for these two data sets, the majority of valuable information for classification is simply in the existing labels of each cell's neighbors. However, the GCN approach does improve performance when multiple cell types are transcriptionally similar (as in the simulated data sets). This suggests the value of message passing the full transcriptional profile of each cell emerges when transcriptional profiles become similar across multiple cell types. These findings are relevant to both this project and the general function of message passing in graphs. As a whole, we find the most value in our GCN approach given its consistency in classification accuracy across all tested data sets and interpretability.

Overall, our method will serve as a valuable tool for researchers. They will be able to easily apply five state-of-the-art labeling tools and obtain accurate labels with our GCN propagation. Importantly, our pipeline accommodates any set of predictions, allowing our tool to be applied to any cell type with or without a reference data set.

Competing interests

No competing interest is declared.

Acknowledgments

This work was supported by the National Institutes of Health [P50 HD096723-01]. Thanks to the Colorado College Department of Mathematics and Computer Science for supporting this research via the faculty summer research stipend. The authors thank Eisa Mahyari for valuable conversations.

Data Availability

All used real scRNA-seq data sets are publicly available and cited above. Our code for conducting the experiments in this paper and using our model is available at `https://github.com/lewinsohndp/scSHARP`.

References

1. 4k pan t cells from a healthy donor.
2. Seurat - guided clustering tutorial, Jan 2022.
3. Tamim Abdelaal, Lieke Michielsen, Davy Cats, Dylan Hoogduin, Hailiang Mei, Marcel JT Reinders, and Ahmed Mahfouz. A comparison of automatic cell identification methods for single-cell rna sequencing data. *Genome biology*, 20:1–19, 2019.
4. Diederik P. AKingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv*, 2014.
5. J. Alquicira-Hernandez, A. Sathe, and H.P. et al Ji. scpred: accurate supervised method for cell-type classification from single-cell rna-seq data. *Genome Biol*, 2019.
6. Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Gradient-based attribution methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 169–191. Springer, 2019.
7. D. Aran, A.P. Looney, and L. et al Liu. Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. *Nat Immunol*, 2019.
8. Joseph Collin, Rachel Queen, Darin Zerti, Sanja Bojic, Birthe Dorgau, Nicky Moyse, Marina Moya Molina, Chunbo Yang, Sunanda Dey, Gary Reynolds, et al. A single cell atlas of human cornea that defines its development, limbal progenitor cells and their interactions with the immune cells. *The ocular surface*, 21:279–298, 2021.
9. Christopher Daniel Green, Qianyi Ma, Gabriel L Manske, Adrienne Niederriter Shami, Xianing Zheng, Simone Marini, Lindsay Moritz, Caleb Sultan, Stephen J Gurczynski, Bethany B Moore, et al. A comprehensive roadmap of murine spermatogenesis defined by single-cell rna-seq. *Developmental cell*, 46(5):651–667, 2018.
10. H. Guo and J Li. scsorter: assigning cells to known cell types according to marker genes. *Genome Biol*, 2021.
11. Shuai He, Lin-He Wang, Yang Liu, Yi-Qi Li, Hai-Tian Chen, Jing-Hong Xu, Wan Peng, Guo-Wang Lin, Pan-Pan Wei, Bo Li, et al. Single-cell transcriptome profiling of an adult human cell atlas of 15 major organs. *Genome biology*, 21(1):1–34, 2020.
12. Aleksandr Ianevski, Anil K Giri, and Tero Aittokallio. Fully-automated and ultra-fast cell-type identification using specific marker combinations from single-cell transcriptomic data. *Nature communications*, 13(1):1–10, 2022.
13. Min Jung, Daniel Wells, Jannette Rusch, Suhaira Ahmad, Jonathan Marchini, Simon R Myers, and Donald F Conrad. Unified single-cell analysis of testis gene regulation and pathology in five mouse strains. *Elife*, 8:e43966, 2019.
14. Vladimir Yu Kiselev, Tallulah S Andrews, and Martin Hemberg. Challenges in unsupervised clustering of single-cell rna-seq data. *Nature Reviews Genetics*, 20(5):273–282, 2019.
15. Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020.

16. Giovanni Pasquini, Jesus Eduardo Rojo Arias, Patrick Schäfer, and Volker Busskamp. Automated methods for cell type annotation on scrna-seq data. *Computational and Structural Biotechnology Journal*, 19:961–969, 2021.
17. Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017.
18. James W Tung, Kartoosh Heydari, Rabin Tirouvanziam, Bita Sahaf, David R Parks, Leonard A Herzenberg, and Leonore A Herzenberg. Modern flow cytometry: a practical approach. *Clinics in laboratory medicine*, 27(3):453–468, 2007.
19. Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.
20. Luke Zappia, Belinda Phipson, and Alicia Oshlack. Splatter: simulation of single-cell rna sequencing data. *Genome biology*, 18(1):1–15, 2017.
21. Ze Zhang, Danni Luo, Xue Zhong, Jin Huk Choi, Yuanqing Ma, Stacy Wang, Elena Mahrt, Wei Guo, Eric W Stawiski, Zora Modrusan, et al. Scina: A semi-supervised subtyping algorithm of single cells and bulk samples. *Genes*, page 531, 2019.
22. Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8(1):1–12, 2017.

A Hyperparameter Search Details

See Figure 3 for details of hyperparameter search on validation set of each data set.

Our model architecture consists of l EdgeConv layers. Each EdgeConv layer consists of one round of message passing along edges of the graph, followed by a dense neural network model that maps from one layer's embedding space to the next layer's. Each node aggregates information using the sum of its received messages (from neighbors and itself). In all of our model architectures, the first layer takes input embedding size 500 and outputs embedding size 1000. The middle layers accept embedding size 1000 and output embeddings of the same size. The final layer accepts embedding size 1000 and outputs final embedding size e. Both hyperparameters number of layers l and final embedding size e are included in the hyperparameter search.

B de.facScale Simulation Parameter

See Figure 4 for details on how the de.facScale parameter affects classification difficulty. With de.facScale ≤ 0.5, the generated data is too difficult for any of the component tools to analyze – all of the component tools do poorly. On the other hand, de.facScale ≥ 0.9 is too easy – the cell types are well-separated enough in gene space that all component methods are able to classify them correctly. We generated simulated data with de.facScale values of 0.7 and 0.8, as these values produce a challenging, but still attainable, benchmark for classification.

C GCN Confusion Matrices

See Figure 5 for confusion matrices from GCN predictions on all data sets. We note that nearly all of these confusion matrices are characterized by a single cell type being the majority of unconfident cells. It is unclear why this is the case for the PBMC and testis data. In the synthetic data, one possible explanation is the way we choose marker genes. We randomly select five of the top ten differentially expressed genes in the simulated data of each cell type as markers (this is standard practice for using Splatter). It is possible this method results in some cell types with better markers than others. This was intentional, as in real-world data not all cell types will always have the same strength of cell type marker. For the actual data sets, this is likely because certain cell types (such as CD4 and CD8 T Cells) are more transcriptionally similar and likely to be misclassified. A common theme in both of these cases is that within each data set, some cell types are inherently easier to classify than others.

D Non-parametric Neighbor Majority Label Propagation

The results for the neighbor-majority label propagation approach across a variety of k values are shown in Figure 6.

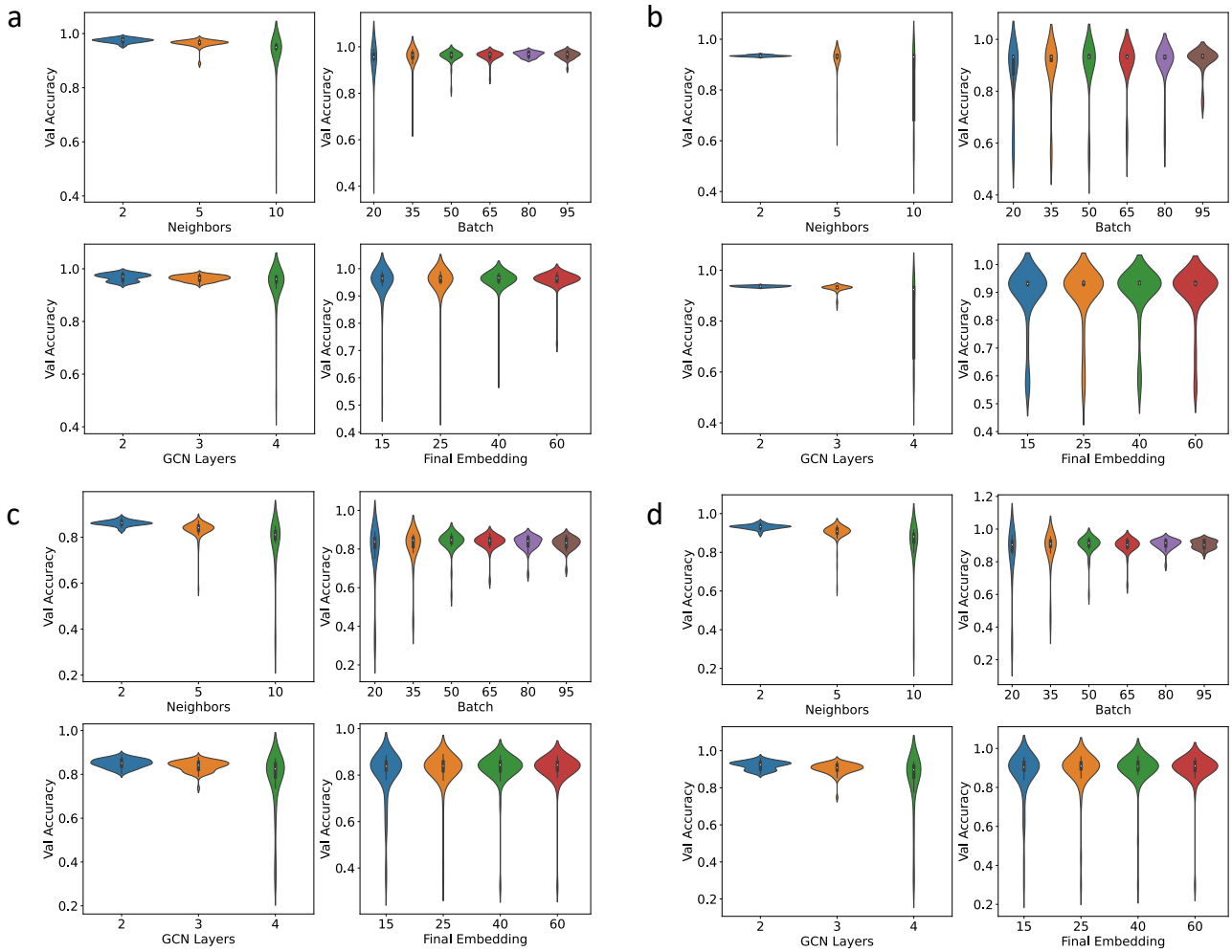

Fig. 3. Spread of validation accuracy scores as a function of various hyperparameters. The hyperparameters included are number of neighbors, batch size, GCN layers, and final embedding layer size. a. Testis data set. b. PBMC data set. c. Simulation 0.7 data set. d. Simulation 0.8 data set

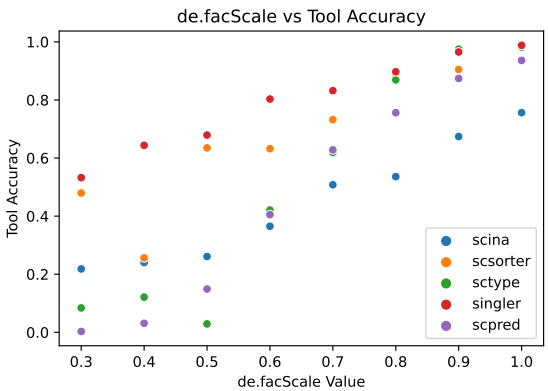

Fig. 4. Plot showing relationship between de.facScale simulation parameter and component tool accuracy. ScType, SCINA, ScSorter, SingleR, and ScPred were included as component tools.

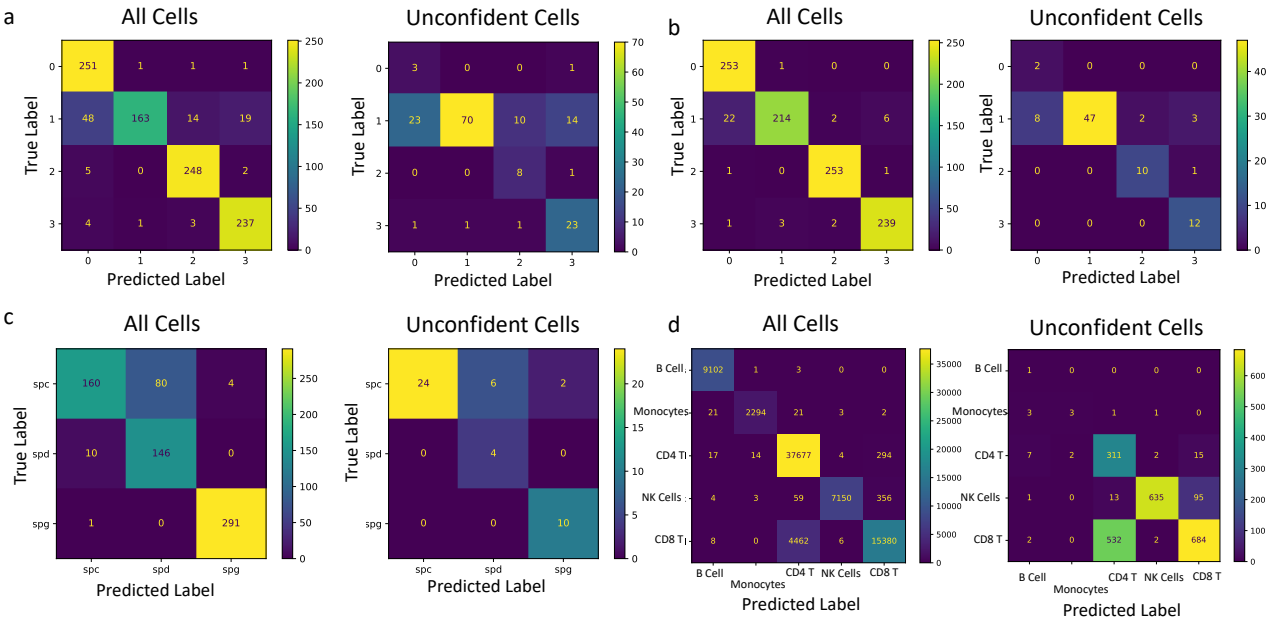

Fig. 5. Confusion matrices for GCN predictions in all and unconfidently labeled cells. a. Simulation 0.7 b. Simluation 0.8 c. Testis d. PBMC

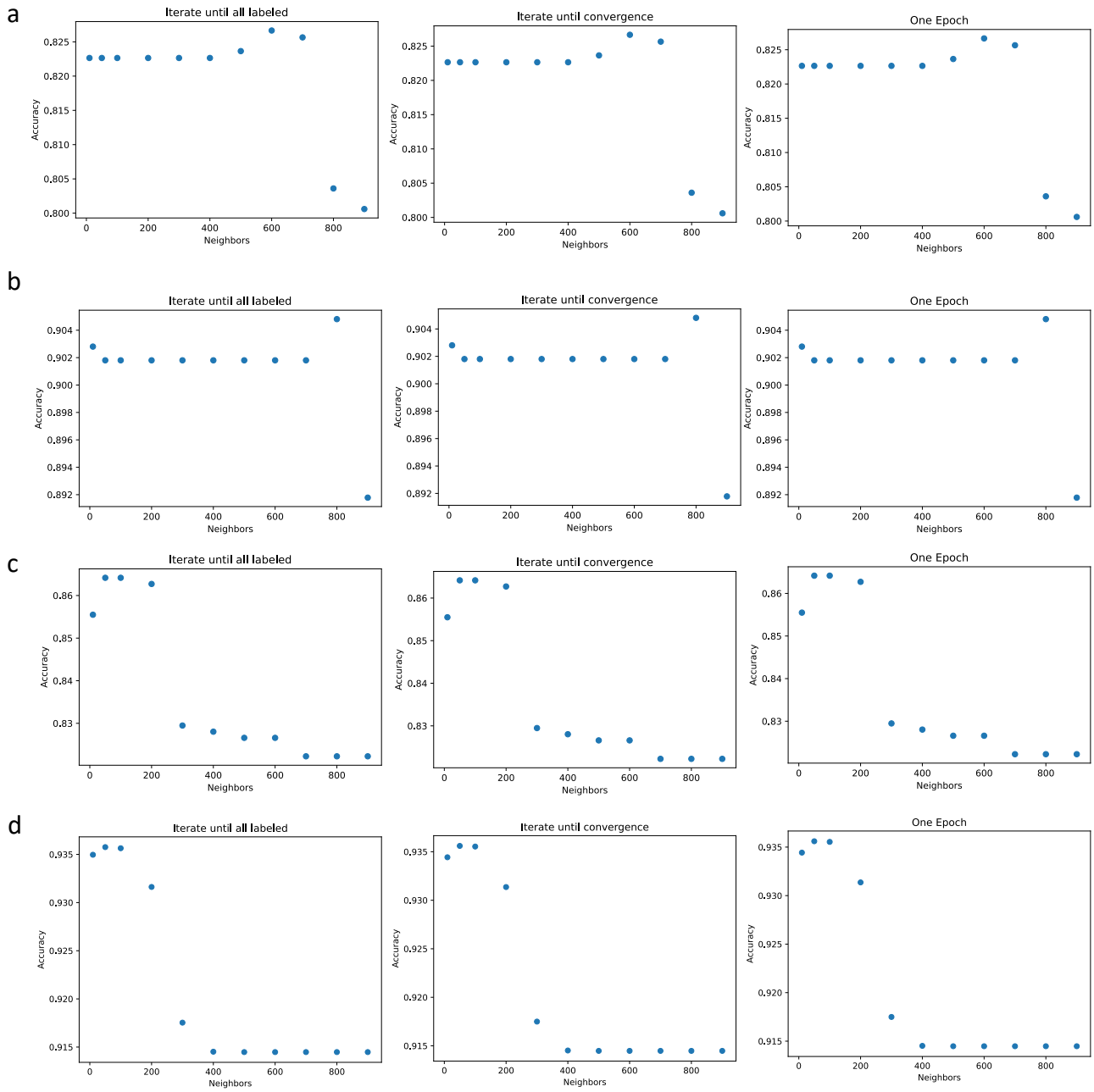

Fig. 6. Number for neighbors vs accuracy percent for different convergence methods and data sets. a. Simulation 0.7 b. Simulation 0.8 c. Testis d. PBMC