Bioinformatics, 38(23), 2022, 5322–5325
https://doi.org/10.1093/bioinformatics/btac684
Advance Access Publication Date: 17 October 2022
Applications Note

Gene expression

scGNN 2.0: a graph neural network tool for imputation and clustering of single-cell RNA-Seq data

Haocheng Gu[1,†], Hao Cheng[1,†], Anjun Ma [1,2], Yang Li[1], Juexin Wang [3], Dong Xu [3,*] and Qin Ma [1,2,*]

[1]Department of Biomedical Informatics, College of Medicine, The Ohio State University, Columbus, OH 43210, USA, [2]Pelotonia Institute for Immuno-Oncology, The James Comprehensive Cancer Center, The Ohio State University, Columbus, OH 43210, USA and [3]Department of Electrical Engineering and Computer Science, and Christopher S. Bond Life Sciences Center, University of Missouri, Columbia, MO 65211, USA

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.
Associate Editor: Inanc Birol

Received on June 13, 2022; revised on August 30, 2022; editorial decision on October 10, 2022; accepted on October 14, 2022

Abstract

Motivation: Gene expression imputation has been an essential step of the single-cell RNA-Seq data analysis workflow. Among several deep-learning methods, the debut of scGNN gained substantial recognition in 2021 for its superior performance and the ability to produce a cell–cell graph. However, the implementation of scGNN was relatively time-consuming and its performance could still be optimized.
Results: The implementation of scGNN 2.0 is significantly faster than scGNN thanks to a simplified close-loop architecture. For all eight datasets, cell clustering performance was increased by 85.02% on average in terms of adjusted rand index, and the imputation Median *L*1 Error was reduced by 67.94% on average. With the built-in visualizations, users can quickly assess the imputation and cell clustering results, compare against benchmarks and interpret the cell–cell interaction. The expanded input and output formats also pave the way for custom workflows that integrate scGNN 2.0 with other scRNA-Seq toolkits on both Python and R platforms.
Availability and implementation: scGNN 2.0 is implemented in Python (as of version 3.8) with the source code available at https://github.com/OSU-BMBL/scGNN2.0.
Contact: xudong@missouri.edu or qin.ma@osumc.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Single-cell RNA-Seq (scRNA-Seq) data provides an opportunity to reveal complex gene regulation mechanisms, build cell–cell relationships and perform analysis in conjunction with other transcriptome data (Luecken and Theis, 2019). One of the primary challenges of the current scRNA-Seq methods is the dropout events (Lähnemann *et al.*, 2020), characterized by the widespread zeros in the gene expression matrix. Our in-house tool, scGNN, aimed to perform gene expression imputation as well as cell clustering using a graph neural network model (Wang *et al.*, 2021). It achieved outstanding performance over existing methods, such as Seurat and MAGIC, on selected datasets. This method provides an effective representation of gene expression and cell–cell relationships. However, the original implementation (referred to as scGNN 1.0) still had some room to improve: (i) cell clustering and gene imputation were not optimized together in the iteration framework; (ii) it is time-consuming, which took a long time for a large scRNA-seq dataset (e.g. about 47 min for analyzing a scRNA-seq data with 6800 cells); (iii) it could not be easily adapted by other existing tools, such as Seurat; and (iv) it did not provide a detailed and clear tutorial. To this end, we developed scGNN 2.0 to overcome these challenges. Empowered by a graph attention mechanism, its efficient implementation imputes gene expressions and identifies cell clusters simultaneously with outstanding performance. We reformed the main framework to optimize gene imputation and cell clustering together in the iteration learning process, making the framework robust and efficient. scGNN 2.0 is geared with new visualizations and expanded ways to integrate with Seurat for result interpretations and downstream analyses. Moreover, as bulk RNA-Seq data can capture lowly expressed genes missed in the scRNA-seq data, integrating bulk RNA-seq and scRNA-seq data may reduce dropout issues in the scRNA-seq data (Peng *et al.*, 2019). To this end, as a new option, scGNN 2.0 allows users to integrate bulk RNA-seq data with scRNA-seq data by bulk deconvolution and formulating a constrained optimization problem to enhance the performance of cell clustering and gene imputation.

Lastly, the overall framework of scGNN 2.0 has been optimized to increase the computational efficiency significantly compared to scGNN 1.0.

2 Implementation

The scGNN 2.0 algorithm consists of three stacked autoencoders following a program initialization. During the initialization, we preprocess the data and employ a left-truncated mixture Gaussian (LTMG) model (Wan *et al.*, 2019). This robust statistical model effectively detects regulatory signals for each gene used as regularization in the feature autoencoders. Unlike scGNN 1.0, where imputation relies on a separate imputation autoencoder that runs after the iterative process, we eliminated this additional autoencoder, keeping only the three autoencoders in an interactive loop (Fig. 1A). This straightforward architecture achieved better performance than the previous version. During the graph autoencoder, we employ a multi-head graph attention mechanism to learn a graph embedding (Veličković *et al.*, 2017). Details of these methods can be found in Supplementary Methods.

With the above modifications, on eight benchmarking datasets (Chu *et al.*, 2016; Goolam *et al.*, 2016; Klein *et al.*, 2015; Leng *et al.*, 2015; Semrau *et al.*, 2017; The Tabula Sapiens Consortium *et al.*, 2022; Trapnell *et al.*, 2014; Usoskin *et al.*, 2015), scGNN 2.0 outperforms Seurat (Hao *et al.*, 2021) on cell clustering and surpasses MAGIC (van Dijk *et al.*, 2018) on imputation (Supplementary Table S1). Compared to scGNN 1.0, scGNN 2.0 demonstrates superior performance in cell clustering (Fig. 1B) and gene imputation (Fig. 1C) (Supplementary Table S2). The per epoch time saved in scGNN 2.0 becomes increasingly pronounced as the expression matrix gets larger (Fig. 1D). We attribute this improvement in run time to efficiency-conscious code designs and more vectorized operations to optimize matrix operations. The recommended minimum computer/workstation configuration for running scGNN 2.0 is provided in Supplementary Method S1.

The input formats of scGNN 2.0 include csv, 10×, SeuratObject and RData, and it can output csv or RData format, opening opportunities to embed scGNN 2.0 into a Seurat workflow for downstream analysis. Furthermore, scGNN 2.0 introduces three visualizations that can be used to evaluate and understand the results: (i) a cell–cell relationship graph where each node represents a cell, colors reflect predicted cell clusters, node positions are based on UMAP projection and the thickness of each edge reflects the similarities between the connecting cells. (ii) When given the ground-truth expression matrix, we can also generate an imputation error heatmap. In this cell-by-gene matrix, cells are grouped by predicted clusters, and the darker color represents a larger difference in the imputed matrix (Fig. 1E). (iii) A Sankey graph can be generated to compare the clustering performance. This may be useful for evaluating the effectiveness of using bulk RNA-Seq data (Fig. 1F).

3 Functions and examples

scGNN 2.0 is packaged into different modules. On a high level, we can interchange each module and use it individually in a custom workflow. Here, we present the main components that make up the scGNN 2.0 program. A complete list of program arguments can be found in Supplementary Table S3. More details of the coding demonstration and corresponding results can be found in Supplementary Examples S1–S3.

3.1 Preprocessing and LTMG modeling

Users can specify the file locations and format using arguments that start with '*–load*'. The optional preprocessing in scGNN 2.0 will perform cell and gene filtering, keeping the top 2000 highly variable genes. Users can customize preprocessing-related options using arguments that start with '*–preprocess*'. Then, a regulatory signal matrix is generated from the preprocessed expression matrix using the function RunLTMG. To filter and only keep the top highly variable genes based on their standard deviation across cells, users can set '*–preprocess_top_gene_select*' to 2000. Users can save the preprocessed data with the '*–output_preprocessed*' indicator. As a result, a directory named 'preprocessed_data' will be created in the output folder, and preprocessed expression data will be saved in the csv format, with rows being genes and columns being cells. If provided, ground-truth cell-type labels will also be saved to reflect the remaining cells with cell names in the first column and cell-type labels in the second column. Details of data preprocessing and LTMG modeling can be found in Supplementary Methods S2 and S3.

3.2 Simultaneous gene expression imputation and cell clustering

The preprocessed expression matrix and the generated regulatory signal matrix are used to initiate the iterative process. To fine-tune the three autoencoders and control the regularizations, users can use arguments prefixed with one of the three autoencoders: '*–feature_AE*', '*–graph_AE*' or '*–cluster_AE*'. For example, to specify the number of epochs and learning rate for cluster autoencoder, the arguments '*–cluster_AE_epoch*' and '*–cluster_AE_learning_rate*' can be applied. To adjust the number of graph embedding features produced by the graph autoencoder, the argument '*–graph_AE_embedding_size*' can be used.

After the final iteration, scGNN 2.0 will produce: (i) an imputed gene expression matrix in the csv format with rows being cells and columns being genes; (ii) a graph embedding matrix in the csv format with rows being cells and columns being the graph embedding dimensions; (iii) a cell graph in the csv format in the form of an edge list. There are three columns, representing the starting node, the ending node and the edge weight; and (iv) a list containing cell cluster labels in the csv format with cell names in the first column and cell labels in the second column. These csv outputs are always provided. In addition, users can set the '*–output_rdata*' indicator argument to get all the outputs above in the RData format. Details of cell clustering and gene imputation can be found in Supplementary Methods S4 and S5.

3.3 Bulk data deconvolution

To integrate bulk RNA-Seq data, scGNN 2.0 implements a deconvolution algorithm in deconvolution.py and imputation.py. Users can fine-tune the algorithm using arguments prefixed with '*–deconv*'. The deconvolution is achieved by solving four optimizations—each optimization has its own set of the learning rate, max epoch and epsilon for convergence. To set these for the first optimization, arguments are: '*–deconv_opt1_learning_rate*', '*–deconv_opt1_epoch*' and '*–deconv_opt1_epsilon*'. Details about the bulk data deconvolution and integration can be found in Supplementary Method S6. We used Chu's data to showcase the results of scGNN 2.0 with and without adding bulk RNA-seq data. Human embryonic stem cell bulk data were downloaded online with 19 samples. Specifically, the cell clustering Adjusted Rand Index (ARI) is 0.772 with bulk data deconvolution (0.717 for not using bulk data), and the median $L1$ error for gene imputation is 0.599 (0.698 for not using bulk data) (Supplementary Table S4). We also tested how bulk data can enhance scGNN 2.0 results in terms of the number of bulk samples. The result indicates that more bulk samples can potentially increase cell clustering and gene imputation results (Supplementary Table S5).

3.4 Visualization

Three built-in visualizations can be generated: (i) a cell–cell graph, using an edge list representing the graph and a set of cell labels. Users can also tune the relative thicknesses of edges and sizes of the nodes by specifying '*edgescale*' and '*node_size*'; (ii) an imputation heatmap, using the ground-truth and imputed expression matrix, as well as the cell labels; and (iii) a Sankey diagram, whether to compare the clustering results between different hyperparameters, or before and after applying bulk data, with a list of cell label files in '*file_list*'. To further visualize the results or perform downstream analysis using other toolkits like Seurat, users can apply the RData

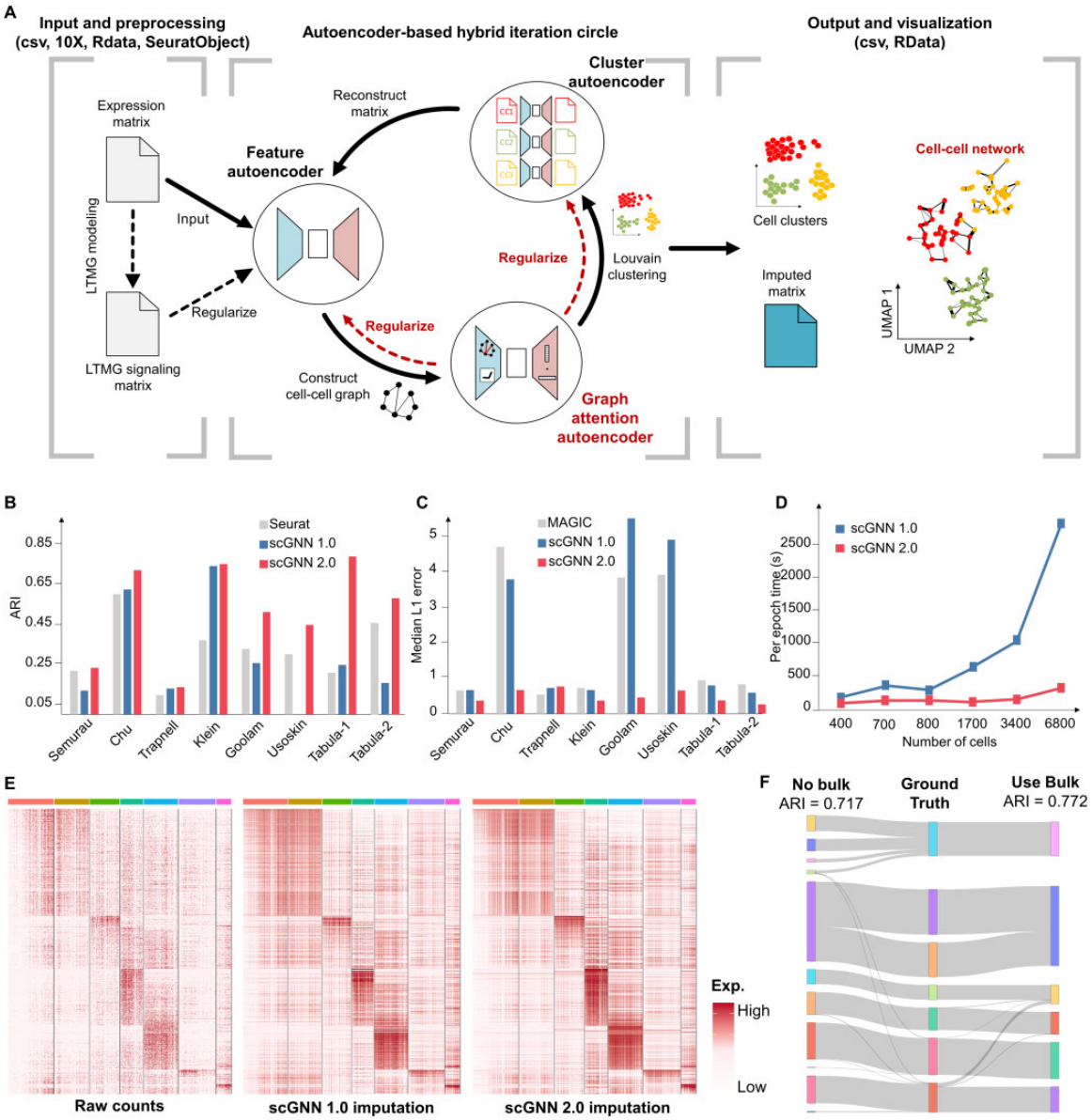

Fig. 1. (**A**) scGNN 2.0 workflow. Comparing to the previous version, we modified the framework by adding new regularizations, the use of graph multi-head attention mechanism, and UMAP based cell-cell network visualization. (**B**) Comparison of cell clustering performance among Seurat, scGNN 1.0 and scGNN 2.0 on six benchmarking datasets. The metric ARI can range from 1 to −1, with 1 meaning having the exact same cell-type prediction as the ground-truth. (**C**) Comparison of gene expression imputation performance, using median $L1$ error as the metric. (**D**) Comparison of per epoch running time between scGNN 1.0 and scGNN 2.0. Six subsets were taken from the same expression matrix to get different cell counts. (**E**) Comparison of differentially expressed genes (DEGs) between original expression (left), after imputation by scGNN 1.0 (middle) and after scGNN 2.0 imputation (right). DEGs were identified using Seurat. (**F**) A Sankey plot comparing the cell cluster predictions without (left) and with (right) integrating bulk RNA-Seq data against ground truth (middle) using Chu's dataset

output format and follow the sample workflow, we provided on GitHub.

4 Conclusion and discussion

scGNN 2.0 has the advantages of better imputation and cell clustering performance, faster runtime and the ability to visualize cell graph results and connect with other toolkits in R and Python. scGNN 2.0 can receive Rdata and SeuratObject files generated by other R toolkits as input (Supplementary Method S7). Using the attention mechanism in the graph autoencoders, we can capture the heterogeneous relations among cells and genes within local and global contexts, making scGNN 2.0 robust to data noise and scale (Ma *et al.*, 2021). Moreover, the option of bulk RNA-Seq data integration can effectively constrain and improve imputation and clustering results. This also opens opportunities for leveraging bulk data and their phenotype information to guide the identification of key cell subpopulations from single-cell data (Sun *et al.*, 2022). We foresee scGNN 2.0 becoming more popular with the expanded connectivity with other analytical platforms like Seurat. Continuous improvements will be made to scGNN 2.0 to drive a deeper understanding of cell–cell interactions and their relationship with bulk data.

Acknowledgements

The authors would like to thank Sheen Brower (Ohio State University, OSU) for his assistance and advice in writing Supplementary Methods. This work was supported by the Pelotonia Institute of Immuno-Oncology (PIIO). The content is solely the responsibility of the authors and does not necessarily represent the official views of the PIIO.

Funding

This work was supported by the National Institutes of Health [R01-131399, U54-AG075931 and R35-GM126985] and the National Science Foundation [NSF1945971].

Conflict of Interest: none declared.

Data availability

scGNN 2.0 is implemented in Python (as of version 3.8) with the source code available at https://github.com/OSU-BMBL/scGNN2.0. The data underlying this article are available in Supplementary Table S1.

References

Chu,L.F. *et al.* (2016) Single-cell RNA-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm. *Genome Biol.*, **17**, 173.

Goolam,M. *et al.* (2016) Heterogeneity in Oct4 and Sox2 targets biases cell fate in 4-cell mouse embryos. *Cell*, **165**, 61–74.

Hao,Y. *et al.* (2021) Integrated analysis of multimodal single-cell data. *Cell*, **184**, 3573–3587.e3529.

Klein,A.M. *et al.* (2015) Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, **161**, 1187–1201.

Lähnemann,D. *et al.* (2020) Eleven grand challenges in single-cell data science. *Genome Biol.*, **21**, 31.

Leng,N. *et al.* (2015) Oscope identifies oscillatory genes in unsynchronized single-cell RNA-seq experiments. *Nat. Methods*, **12**, 947–950.

Luecken,M.D. and Theis,F.J. (2019) Current best practices in single-cell RNA-seq analysis: a tutorial. *Mol. Syst. Biol.*, **15**, e8746.

Ma,A. *et al.* (2021) DeepMAPS: single-cell biological network inference using heterogeneous graph transformer. *bioRxiv preprint*, https://doi.org/10.1101/2021.10.31.466658.

Peng,T. *et al.* (2019) SCRABBLE: single-cell RNA-seq imputation constrained by bulk RNA-seq data. *Genome Biol.*, **20**, 88.

Semrau,S. *et al.* (2017) Dynamics of lineage commitment revealed by single-cell transcriptomics of differentiating embryonic stem cells. *Nat. Commun.*, **8**, 1096.

Sun,D. *et al.* (2022) Identifying phenotype-associated subpopulations by integrating bulk and single-cell sequencing data. *Nat. Biotechnol.*, **40**, 527–538.

The Tabula Sapiens Consortium *et al.* (2022) The tabula sapiens: a multiple-organ, single-cell transcriptomic atlas of humans. *Science*, **376**, eabl4896.

Trapnell,C. *et al.* (2014) The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nat. Biotechnol.*, **32**, 381–386.

Usoskin,D. *et al.* (2015) Unbiased classification of sensory neuron types by large-scale single-cell RNA sequencing. *Nat. Neurosci.*, **18**, 145–153.

van Dijk,D. *et al.* (2018) Recovering gene interactions from Single-Cell data using data diffusion. *Cell*, **174**, 716–729.e727.

Veličković,P. *et al.* (2017) Graph attention networks. *arXiv preprint, arXiv*: 1710.10903.

Wan,C. *et al.* (2019) LTMG: a novel statistical modeling of transcriptional expression states in single-cell RNA-Seq data. *Nucleic Acids Res.*, **47**, e111.

Wang,J. *et al.* (2021) scGNN is a novel graph neural network framework for single-cell RNA-Seq analyses. *Nat. Commun.*, **12**, 1882.