Structural Inductive Bias in Supervised Learning for Single-cell Data

Elyas Heidari[*1]

[1]Department of Biological Systems Sciences and Engineering, ETH Zürich, Switzerland

[*]eheidari@student.ethz.ch

January 12, 2021

Abstract

Recent advancements in single-cell RNA sequencing (scRNAseq) have unraveled the transcriptional diversities underlying heterogeneous batches of cells. Current single-cell classification methods take genes as independent features to assign cell types disregarding their interactome which has been shown to be of great importance in cellular processes. As a result, existing methods are prone to dataset specific classification artifacts which prevents generalization of the model for cell type classification in new datasets. Here, we introduce single-cell Graph Convolutional Network (scGCN) which takes gene-gene interactions into account by constructing a co-expression network of a subset of genes and using a graph deep learning approach to classify single-cells. Using two different peripheral blood mononuclear cell (PBMC) datasets as train and test sets, we demonstrate the potentials of scGCN for transfer learning.

Contents

gi

1 Introduction and motivation

Single cell transcriptomics have enabled resolution of transcriptional diversities of individual cells, hence the possibility to classify cells' identity and function via their transcriptional state. Noting that such identity classifications can elucidate organization and function of tissues in health and disease states, recently more investments have been devoted to this task, for example, in the Human Cell Atlas project (**???**).

Current single-cell RNA sequencing (scRNA-seq) classification methods use genes expression as independent features to assign cell types, disregarding their interactome which is of great importance in consistency of different cell states. Cell type classification based on a few marker genes, for example, yields better results in intra data set settings where both train and test datasets are sampled from the same dataset (Abdelaal et al. 2019) and may fail to correctly classify cell types under perturbations when expression of the predefined marker genes varies with respect to the control reference data set. Moreover, using genes expression as independent features can delude cell lineage relationships for cell states and cell types with subtle difference in their gene expression profiles. In (Fig. **??**) we illustrate the two fold benefit of a graph representation of cell states in removing unwanted and technical variations (batch effects) as well as providing a better resolution of transitory cell states and cell lineages.

Structures are ubiquitous in biology. In reality, components of a biological system, interact with each other, forming functional modules (e.g., pathways), and in turn, modules interact to form a complex network creating the biological system. Gene regulatory networks, namely, perfectly embody such a phenomenon. While in many applications, as mentioned above, biological states (e.g., cell type), can be described by a few components (e.g., marker genes), incorporating the underlying structure can provide many useful inputs both for discriminative tasks (e.g., cell type classification) and interpretation tasks (e.g., finding marker genes/pathways). This is the goal of this project. That is to leverage structures on which input features function upon as an inductive bias for supervised learning on single-cell data. More precisely, here we examine how we can exploit such underlying structures to perform cell type classification based on single-cell RNA seq data.

In this project, we focus on two underlying structures concerning the transcriptome. One is the gene-gene interaction network, on which genes interact with each other. The other is genomic locations, which can be precieved as a linear ordering of genes on the chromosomes. We show how including such structures in the neural network classifiers facilitates better accuracy, robustness, and interpretablity. We also provide an end-to-end pipeline to reconstruct and feed in such structures to the neural network classifiers. We decompose the problem of supervised learning into two parts. The first part is to train a model to perform annotation, which hereafter we call the "forward" problem. Once the model is trained, we interpret the model, that is, we aim to capture features which are the classes' characteristics In other words, we seek for a few number of features which can discriminate classes. We refer to the later as the "backward" problem.

2 Materials and Methods

2.1 PBMC dataset

A pivotal application of cell annotation is transferring knowledge gained from one dataset to new unseen datasets. Therefor, we sought fro two datasets with the same cell type profile, but acquired in separate experiments to inspect how well the method is able to transfer between various environments and overcome unwanted variations (such as intrinsic noise and batch effects). We chose two human peripheral blood mononuclear cell (PBMC) scRNA-seq experiments (Ding et al. 2019). The dataset contains two separate experiments, hereafter called PBMC 1 and PBMC 2, generated in a single center, but employing seven different scRNA-seq methods. This suits our goal to verify our framework on a simulated task of real transfer learning. As in potential applications we expect our methods to learn annotation priorly on a complete dataset and transfer the learned model to annotate an unseen one.

2.2 Models to incorporate genomic locations

Genomic locations and gene proximity on chromosomes can be used as an additional information for cell annotation. The approach is inspired by the fact that genes in proximity on the chromosomes, are more likely to undergo the same epigenetic events, such as chromatin openness and genomic interactions (e.g., in topologically associating domains, TADs). One can sort genes based on genomic locations. By doing so, each cell can be represented as either a 1-dimensional signal in which each (time) point represents the expression/activation level of the corresponding gene, or a linear graph in which each node is assigned by the expression level of the corresponding gene. By such a modeling, we use both 1-dimensional Convolutional Neural Networks (ref) and Graph Neural Networks (ref). To perform the forward task of supervised learning.

2.3 Models to incorporate gene-gene interactions

It is known that in each cell proteins interact with each other through several regulatory mechanisms, forming an interaction network. In the single-cell field, however, we lack such an information at the protein level, yet, we can use rich RNA-seq datasets. One can use prior information, such as protein-protein interaction databases (ref), and replace genes with their protein products. Also, one can reconstruct the gene-gene interaction network from scratch, by inferring it from the input dataset, e.g., using graphical models. The idea is to provide the model with a structure, on which features (here genes) interact with each other. There is a multitude of approaches to do so, but we do not take all of them into account here. Instead, we introduce a class of models, which are applicable to all graphical structures.

Recent advancements of deep learning models to capture modular structures in other applications such as image, audio and speech analysis as well as emergence of high-throughput scRNA-seq technologies which provide the large amounts of data required for such models motivated us to employ deep learning models applicable to non-regular graph data. Amongst such models, which are generally referred to as “geometric deep learning” (Bronstein et al. (2017), Monti et al. (2017)), Graph Convolutional Networks (GCNs) have significant power in extracting informative feature maps from signals defined over nodes of a graph while considering meaningful local structures and connectivities among them. This makes GCNs suitable for our purpose of analyzing non-regular gene networks (Zhou et al. 2018).

2.4 Baseline model

We investigate to which extent inferring the graph structure from the data and using it as an inductive bias to a GCN improves robustness, generalization, and interpretation in comparison to the standard fully connected neural networks (FCNN) without any prior structural information.

2.5 Interpreting Neural Network models

While Neural Networks are widely known as black box models for they are difficult to interpret, several methods have been recently put forward to interpret and explain Neural Networks (ref). In particular, captum library (ref) has been recently introduced in python to facilitate interoperability of neural networks. We use captum to find the characteristic features or marker genes of cell types.

All Neural Networks are implement in python, using pytroch (ref) and pytorch-geometric (ref). For preprocessing, visualization, and exploratory data analysis we used R (ref).

3 Experiments and results

3.1 Preprocessing

We take PBMC 1 as our training dataset and PBMC 2 as our test dataset. The task is to learn the model on PBMC 1, and validate it on PBMC 2, by predicting cell type annotations, and comparing them to true labels. Preprocessing includes three steps, first, cell type selection and cell-wise data preprocessing, second, gene subset selection and lastly, structure reconstruction:

- **Cell type selection & preprocessing:** We select a subset of cell types in the intersection of cell types present in both training and test datasets. Afterwards, data for each cell should be preprocessed. Here, we used scran package (ref) in R to normalize counts. We ended up with 400 cells per class for the training dataset. We do not subsample the test dataset. Cell counts for training and test datasets is shown in (ref).
- **Gene subset selection (for gene interaction networks):** Following the widely-used procedure in the field, we select highly variable genes, using the scran package. We ended up with 260 genes in total.
- **Gene subset selection (for genomic locations):** We pick one chromosome to map genes to their genomic location. We use the gene mapping from the (ref).
- **Gene network reconstruction:** We use Gaussian graphical models implemented by R package glasso (ref) to reconstruct a sparse gene interaction network for the training dataset. The outcome is a single gene interaction network which we will feed in along the count matrices to the classifiers.

A 2-dimensional embedding of the training and test datasets after preprocessing is shown in figure (ref). As one can see, the classes are not clearly separated and such a heterogeneity affects the classification accuracy.

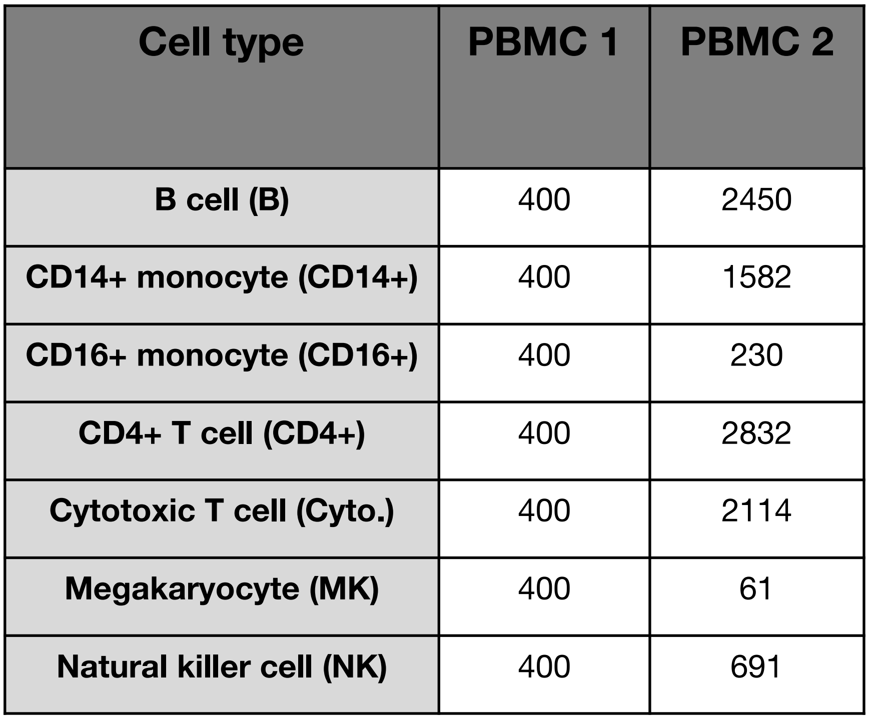
3.2 Classification based on gene interaction networks

We use a well-known graph neural network called GraphSAGE. In GraphSAGE, each node generates a message based on its features, and sends it to its neighbors. Then each node aggregates the messages of its neighbors and uses this aggregate to updates its features. The aggregation is done with a permutation-invariant function. In GraphSAGE (ref), the updated feature vector of node i after one round of message passing is $x'_i = W_1 x_i + W_2 \mathrm{mean}_{j \in \mathcal{N}(i)}(x_j)$ where W_1 and W_2 are learnt weight matrices, shared for all of the nodes, and $\mathcal{N}(i)$ is the set of neighbors of i. In our setting, each node starts with only one feature (which is the expression level of one gene), but we can have a message-passing function that creates hidden feature vectors of higher dimensions.

As mentioned before, after training the graph neural network, we use captum to find the most important features, based on Integrated Gradients method (ref). The results are shown in fig (ref). We also compared the results to a FCNN which does not impose any structure on the input features. By comparison to the Human Protein Atlas, our results suggest how using graphical structure as an inductive bias leads to improvement in interpretation. While, FCNN can not fully capture marker genes of the PBMC cell types, GCN perfectly recovers the characteristic genes (ref). While, FCNN simply recovers the genes with the higher mean in each class, GCN recovers the true genes or gene modules which are known to contribute in cell identities. Both FCNN and GCN are optimized naively, by selecting a proper learning rate, dropout, and number of hidden layers and hidden nodes.

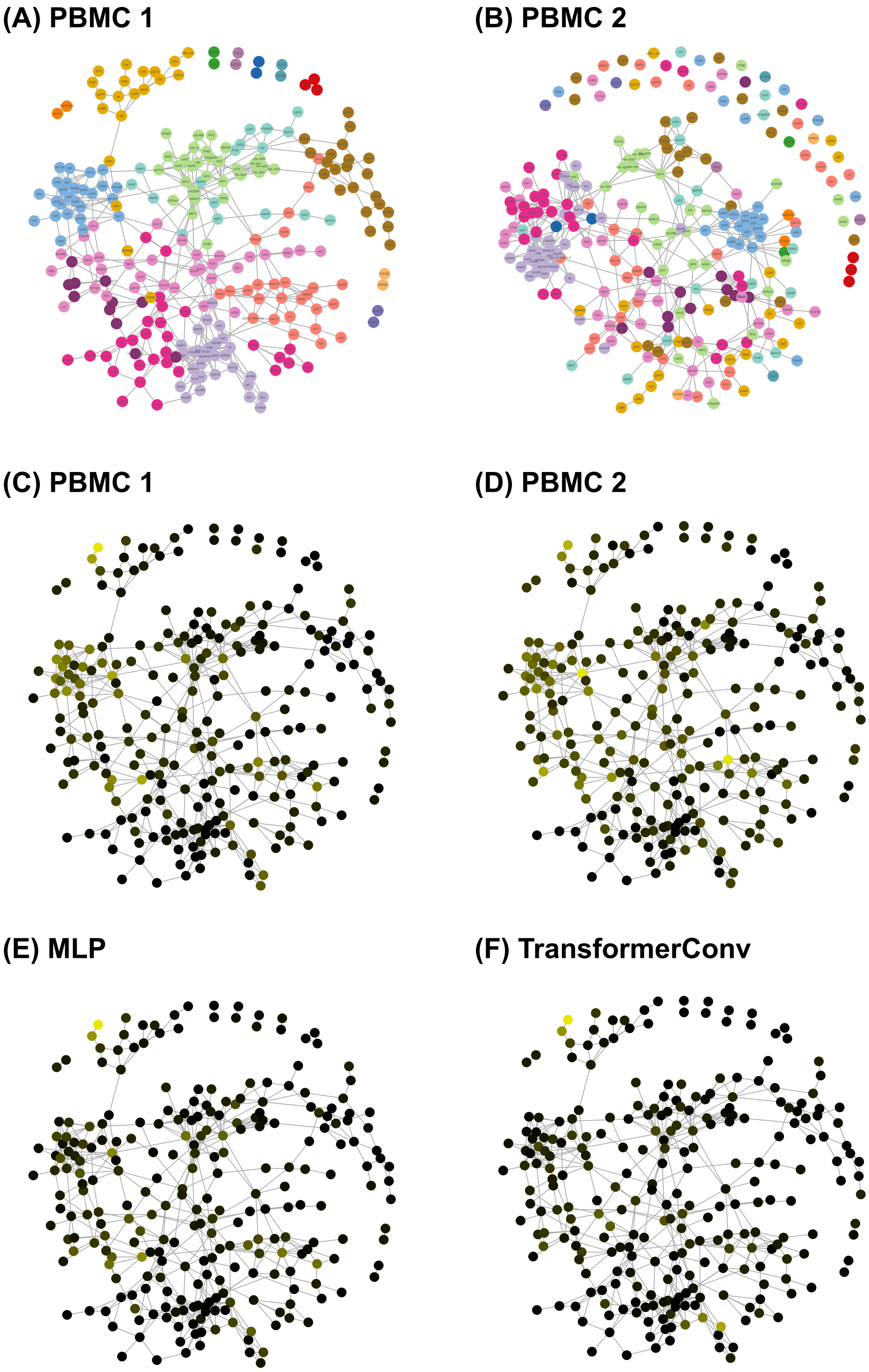
3.3 Classification based on genomic locations

TODO



Cell type	PBMC 1	PBMC 2
B cell (B)	400	2450
CD14+ monocyte (CD14+)	400	1582
CD16+ monocyte (CD16+)	400	230
CD4+ T cell (CD4+)	400	2832
Cytotoxic T cell (Cyto.)	400	2114
Megakaryocyte (MK)	400	61
Natural killer cell (NK)	400	691

Figure 1: The graphical abstract of scGCN



(A) PBMC 1

(B) PBMC 2

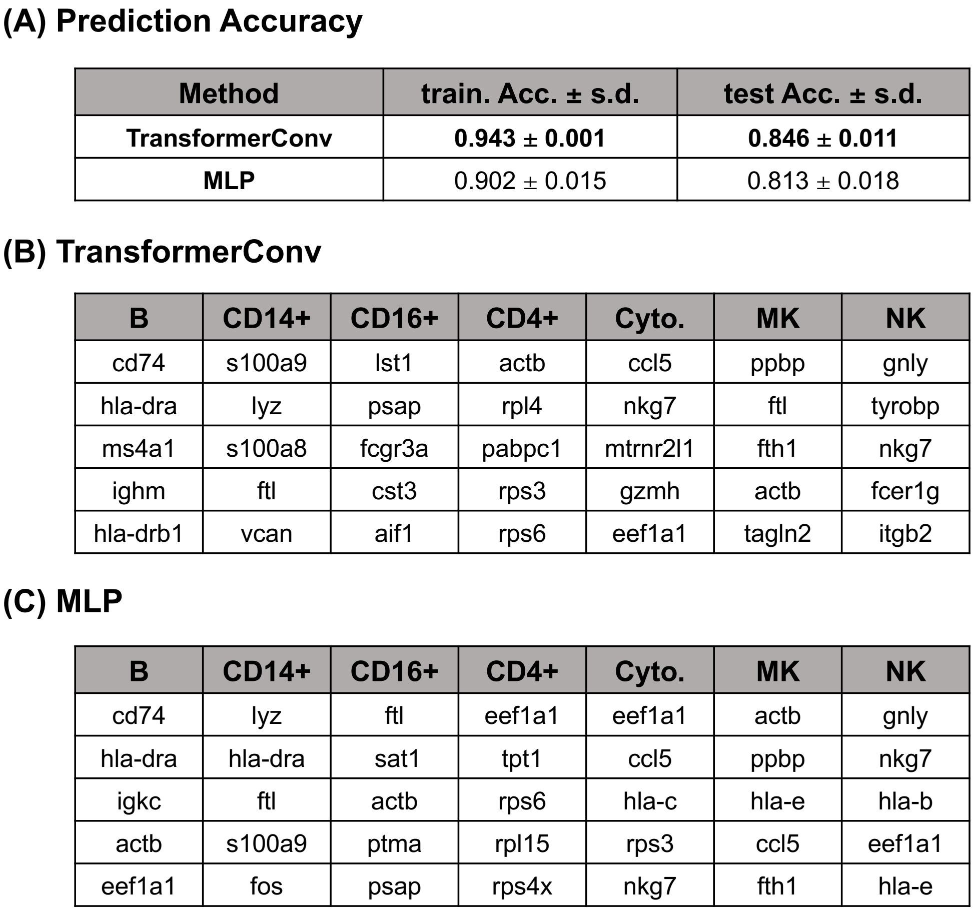
(C) PBMC 1

(D) PBMC 2

(E) MLP

(F) TransformerConv

Figure 2: The graphical abstract of scGCN



(A) Prediction Accuracy

Method	train. Acc. ± s.d.	test Acc. ± s.d.
TransformerConv	**0.943 ± 0.001**	**0.846 ± 0.011**
MLP	0.902 ± 0.015	0.813 ± 0.018

(B) TransformerConv

B	CD14+	CD16+	CD4+	Cyto.	MK	NK
cd74	s100a9	lst1	actb	ccl5	ppbp	gnly
hla-dra	lyz	psap	rpl4	nkg7	ftl	tyrobp
ms4a1	s100a8	fcgr3a	pabpc1	mtrnr2l1	fth1	nkg7
ighm	ftl	cst3	rps3	gzmh	actb	fcer1g
hla-drb1	vcan	aif1	rps6	eef1a1	tagln2	itgb2

(C) MLP

B	CD14+	CD16+	CD4+	Cyto.	MK	NK
cd74	lyz	ftl	eef1a1	eef1a1	actb	gnly
hla-dra	hla-dra	sat1	tpt1	ccl5	ppbp	nkg7
igkc	ftl	actb	rps6	hla-c	hla-e	hla-b
actb	s100a9	ptma	rpl15	rps3	ccl5	eef1a1
eef1a1	fos	psap	rps4x	nkg7	fth1	hla-e

Figure 3: The graphical abstract of scGCN

4 Discussion

4.1 Transferability of the gene networks

4.2 Applications

4.3 Software developed

4.4 Future steps

4.4.1 Input

We take PBMC 1 as our training dataset and PBMC 2 as our test dataset. The task is to learn the model on PBMC 1, and validate it on PBMC 2, by predicting cell type annotations, and comparing them to true labels. We select a subset of size 6 of cell types, to carry out our computational experiment. Cells with less than 100 counts and cells with high percentage of mithocondrial gene counts (>0.01) are filtered out due to low sequencing/experimental quality. We subsample CD4+ and cytotoxic T cells, B cells, Natural killer cells, Megakaryocytes, Denderic cells, CD14+ and CD16+ monocytes, each of which with 150 cells in the training dataset. We do not subsample the test dataset for the aforementioned cell types. Afterwards, a subset of 300 most highly variable genes is selected from PBMC 1 and PBMC 2 is subset to the same set of genes. For GRN reconstruction, we use the glasso algorithm with tuned parameters, which gives us a modular as well as sparse GRN. The GRN is presented in Fig. **??**.

Fig. **??**(A) and Fig. **??**(B) indicate that although our GRN reconstruction method is based on data and no prior knowledge, it is robust across datasets. As one finds in the figures, the network communities from PBMC 1, perfectly transfer to PBMC 2. Fig. **??**(C) illustrates the same observation, as the density of expressions, i.e., activated modules are the same for both datasets, for each cell type. In fact, such an observation motivated us for the whole idea of exploiting a network topology to classify cell types.

4.4.2 Classification with InceptionGCN

We implement an InceptionGCN with three hidden layers with 36, 18, and 9 hidden nodes, respectively. At each layer, a complex, non-linear transformation (Chebnet filters) are applied to the inputs (see Fig. **??**). One can describe such operations as diffusing information along the input GRN which further facilitates an accurate classification. Our InceptionGCN gives an accuracy of 87% on test dataset, which is rather a very high accuracy in a transfer learning task, specifically, when the classes are quite similar to each other (including two classes of T cells, and two classes of monocytes). Fig. **??** indicates the results of the classification on training and test datasets. In order to compare our results with a standard model, we also implement a fully connected (FC) model with two hidden layers (with 200 and 100 neurons respecetively). We optimized hyperparametrs (learning rates and locality sizes for the InceptionGCN) of the models by running multiple experiments examining perfrmance using different sets of hyperparameters. For both models, we use early stopping[1] to prevent overfitting[2].

[1] **Early stopping:** To stop training at the point which validation accuracy start to drop.

[2] **Overfitting:** when a learning model corresponds too tightly to a specefic dataset and fails to generalize to new data. Complex models such as deep neural networks are very prone to such a phenomenon.

4.4.3 Analysis of the results

We achieve 89.6% and 87.5% label prediction accuracy on the training and test data sets respectively. This is a significant result demonstrating the transferable learning capabilities of scGCN on a test data set which has been collected in a different experiment than that used for training the model. For comparison, we include classification results from the FC neural network which yields an accuracy of 92.8% on training data, but drops to 86.3 on the second PBMC dataset as the test set. Although the difference between prediction accuracies is not significant, we observe a bigger difference between accuracy on training data and test data for the FC model. This shows a better generalization potential for our InceptionGCN model.

As mentioned before, one of our objectives in the design of the single-cell geometrical deep learning framework has been interpretability. Our framework enables us to explore the learning procedure in our deep neural network. Our design facilitates representation learning,[3] in that one can visualize the data at the hidden layers and observe the trajectory of data passing through the complex transformations, done by the InceptionGCN. This gives an overview of the learning procedure, inspecting batch-effect removal, and heterogeneous cell populations being separated. One can also use the same information to study the trajectory of expression density along the GRN, for each class. This gives one an idea of how information diffuses on the network and utilizes the classification task. Fig. **??** represents an embedding trajectory analysis, visualized as UMAPs. As indicated, we observe gradual batch-effect removal and cluster separation along the layers. Again for comparison, embeddings of the hidden layers for the FC network is also shown in Fig. **??**.

5 Discussion and future steps

We have demonstrated the potentials of learning on gene networks for transferable learning, i.e., learning models which are least affected by data set specific variations and hence are more general and can be used for predictions on new data as well.

On a test data of PBMC cells our scGCN outperforms a standard neural network which does not use the GRN information by about 1 percent (87 percent versus 86 percent, on test data). Regarding the GRN construction step, among all method that we examined so far, Gaussian Graphical Models (GGM) provide the most robust gene network which are compatible with the scGCN design as they are relatively sparse and constitute modular networks architecture. Sensibility of the used GRN is crucial for inferring correct cell type relationships, as it is through this network that we impose the inductive bias on our learning models. Thus, reliable knowledge about the GRN could boost the methods generalizability and transferable learning. Therefore, inclusion of available biological knowledge in our GRN stays of great interest for our future work and advancement of the framework. Upon availability of gene regulatory network architecture, we anticipate application of scGCN framework for transferable learning among different data modalities, e.g. learning on RNA-seq data and making reliable predictions on ATAC-seq data set of the same system.

Lastly, one of our motivations for creating the scGCN framework has been to obtain a more comprehensive understanding of cell type relationships and developmental trajectories resolution as shown in Fig. **??**. In fact, cell-cell Euclidean distances found on the scGCN hidden layers (i.e. after application of graph convolutional filters) is such a desired distances as defined by kernel F in Fig. **??** as one can write:

$$(FX - FY)^2 = X^T F^2 X + Y^T F^2 Y - X^T F^2 Y - Y^T F^2 X$$

[3] **Representation learning** is learning representations of the data by transforming or projections into repetitively simpler spaces (Bengio, Courville, and Vincent 2013).

We can check improvement of cell lineage relationships resolution by comparing embedding (e.g. by UMAP or Diffusion map) of raw data with that of later (or the final) hidden layer of the scGCN, as in Fig. **??**.

Code and data availability

- Code: GitHub repository
- PBMC dataset: link

References

Abdelaal, Tamim, Lieke Michielsen, Davy Cats, Dylan Hoogduin, Hailiang Mei, Marcel JT Reinders, and Ahmed Mahfouz. 2019. "A Comparison of Automatic Cell Identification Methods for Single-Cell Rna Sequencing Data." *Genome Biology* 20 (1). Springer: 194.

Bengio, Yoshua, Aaron Courville, and Pascal Vincent. 2013. "Representation Learning: A Review and New Perspectives." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (8). IEEE: 1798–1828.

Bronstein, Michael M, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. "Geometric Deep Learning: Going Beyond Euclidean Data." *IEEE Signal Processing Magazine* 34 (4). IEEE: 18–42.

Ding, Jiarui, Xian Adiconis, Sean K Simmons, Monika S Kowalczyk, Cynthia C Hession, Nemanja D Marjanovic, Travis K Hughes, et al. 2019. "Systematic Comparative Analysis of Single Cell Rna-Sequencing Methods." *BioRxiv*. Cold Spring Harbor Laboratory, 632216.

Monti, Federico, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. 2017. "Geometric Deep Learning on Graphs and Manifolds Using Mixture Model Cnns." In *Proceedings of the Ieee Conference on Computer Vision and Pattern Recognition*, 5115–24.

Zhou, Jie, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2018. "Graph Neural Networks: A Review of Methods and Applications." *arXiv Preprint arXiv:1812.08434*.