Gene Regulatory Network Inference using 3D Convolutional Neural Network

Yue Fan, Xiuli Ma*

Department of Machine Intelligence, School of EECS, Key Laboratory of Machine Perception (MOE), Peking University, Beijing, China

fanyue@pku.edu.cn, xlma@pku.edu.cn

Abstract

Gene regulatory networks (GRNs) consist of gene regulations between transcription factors (TFs) and their target genes. Single-cell RNA sequencing (scRNA-seq) brings both opportunities and challenges to the inference of GRNs. On the one hand, scRNA-seq data reveals statistic information of gene expressions at the single-cell resolution, which is conducive to the construction of GRNs; on the other hand, noises and dropouts pose great difficulties on the analysis of scRNA-seq data, causing low prediction accuracy by traditional methods. In this paper, we propose *3D Co-Expression Matrix Analysis* (3DCEMA), which predicts regulatory relationships by classifying 3D co-expression matrices of gene triplets using a 3D convolutional neural network. We found that by introducing a third gene as a comparison factor, our method can avoid the disturbance of noises and dropouts, and significantly increase the prediction accuracy of regulations between gene pairs. Compared with other existing GRN inference algorithms on both in-silico datasets and scRNA-Seq datasets, our algorithm based on deep learning shows higher stability and accuracy in the task of GRN inference.

Introduction

Single-cell RNA sequencing (scRNA-seq) is a technique of high-resolution transcriptomic analysis of individual cells (Kolodziejczyk et al. 2015). The first scRNA-seq method was proposed by Fuchou Tang et al (Tang et al. 2009). In contrast with bulk transcriptomic data which provide the average gene expression values of all cells within each sample, scRNA-seq data provide a two-dimensional matrix of cells and genes, with each value representing a gene's expression level in a cell. The variation of the gene expression values across cells provides new research perspectives. By observing gene expression level at the single-cell resolution, scRNA-seq brings new computational biology topics such as cell-clustering (Tasic et al. 2016) and cell-differentiation (Xie et al. 2020), helping people understanding cell heterogeneity, dynamic biological processes (Deng et al. 2014) and diseases such as cancer (Nam, Chaligne, and Landau 2021). However, scRNA-seq data suffer from features such as low signal-to-noise ratio and high sparsity caused by measurement dropouts, which bring great challenges to the analysis of the scRNA-seq data.

Genes are the key to all biological processes, therefore figuring out how genes interact with each other has become an important and attracting scRNA-seq study. Gene regulatory networks (GRNs) are used to depict regulatory relationships among genes, aiming to simplify the analysis of complex biological systems (Chasman, Fotuhi Siahpirani, and Roy 2016). A GRN is composed of transcription factors (TFs), genes and regulations. Each node of the GRN represents a TF or a gene. The edges connecting TFs and target genes indicate regulations. The edges are either labeled as activation or inhibition, showing the regulatory function of the TFs on their target genes.

ScRNA-seq based GRN inference algorithms are computational methods of reconstructing GRNs using scRNA-seq datasets. Given the datasets, the task is to find out all possible regulations among genes. Methods have been proposed from different computational perspectives, such as regressions, mutual information, correlations, etc. But most existing methods suffer from the drawback of low accuracy. The following factors may account for it.

Zero-Inflation and Noise Zero-inflation and low signal-to-noise ratio are two main characteristics of scRNA-seq data. Large percentage of dropouts, up to 80% for instance, poses difficulties on traditional computational methods such as regression and correlation. Besides, noises may overwhelm the exact expression values of genes, making it hard for algorithms to recognize true regulations.

Cascade Effect Tools such as SLINGSHOT (Kelly et al. 2018) take raw scRNA-seq data as input, aiming at giving cells the timestamps to indicate the differential order. Models have been proposed to infer GRNs using temporal scRNA-seq data with pseudotime order. However, cascade effect on accuracy occurs when the tool fails to generate the precise pseudotime order, causing poor performance of prediction tasks.

Assumption Dependence Most algorithms work on their own assumptions to infer GRNs. For instance, some methods construct GRNs by estimating an ordinary differential equation given the pseudotime information. However, whether these assumptions depict the very nature of gene

*Corresponding author: Xiuli Ma

regulation remains open to question. Methods based on wrong assumptions may suffer from inaccuracy.

To address these challenges, we propose our method called *3D co-expression matrix analysis* (3DCEMA), which applies a 3D convolutional neural network (CNN) to infer gene regulations. To avoid the cascade effect on accuracy, 3DCEMA takes raw scRNA-seq data rather than temporal data as input. No assumptions are needed for our deep learning model. Given a gene pair of x and y, the model compares y with every other gene z to test if there is a regulation of x on y with reference to z. 3D CNN do this comparison process by classifying the 3D co-expression matrix of x, y and z, which implies the joint distribution of the three genes in all cells. A GRN is inferred by synthesizing the predicted labels of all 3D matrices. 3DCEMA is less affected by zero-inflation and low signal-to-noise ratio by focusing on the asymmetry of y and z, which are under equal influence of noises and measurement dropouts.

The model is trained and tested on the datasets of BEELINE (Pratapa et al. 2020), which is an evaluation framework for benchmarking different GRN inference algorithms based on single-cell transcriptomic data. Experiments show that 3DCEMA has strong stability over the in-silico datasets and has significant higher accuracy on large scRNA-seq datasets compared to state-of-the-art algorithms.

Our work makes the following contributions:

- We propose a deep learning method called 3DCEMA, which is the first to apply the 3D convolutional neural network to the task of GRN inference based on single-cell transcriptomic data.
- A unique labeling method is employed for training and inferring. Given three genes x, y and z, the label of them reveals the relative regulation of x on y to z by reflecting the asymmetry between y and z. The GRN can be inferred by synthesizing the predicted labels of all matrices.
- 3DCEMA significantly outperforms other GRN inference methods in both stability and accuracy on scRNA-seq datasets, and may serve as a reliable tool for other co-expression analysis tasks.

Related Work

Gene regulatory network inference has attracted much research attentions with the prevailing of sc-RNA sequencing technology in the past decade. Methods have been proposed from different computational perspectives.

Regression methods determine a gene's possible regulators by focusing on large coefficients of its regression on other genes. GENIE3 (Huynh-Thu et al. 2010) first employed random forests and extra-trees to solve the regression problems. Later, a method based on gradient boosting trees called GRNBoost2 was proposed and was aggregated with GENIE3 into a framework called Arboreto (Moerman et al. 2018). Besides tree based methods, there are also attempts to apply regressions to discover the dynamics of regulations. For instance, GRNVBEM (Sanchez-Castillo et al. 2017) analyzes pseudotime ordered data by using a regressive model within a variational Bayesian framework; regularized linear regression is employed by SINCERITIES (Papili Gao et al. 2017) to identify regulations after cell stimulation; SCINGE (Deshpande et al. 2019) uses Granger causality regression to overcome irregular pseudotime and dropouts. Regressive models explain the mechanics of regulations to some extent, while they are not well capable of discovering possible nonlinear patterns of regulations, which however, can be captured by the deep learning model.

There are methods using ordinary differential equation (ODEs) to infer regulations. An ODE is a kinetic equation which depicts the dynamic regulation functions between genes. Based on the ODE assumption, SCODE (Matsumoto et al. 2017) rebuilds gene regulations, while GRISLI (Aubin-Frankowski and Vert 2020) models the dynamics of cell trajectories by inferring the velocity of each cell. Cascade effects on accuracy become a major obstacle for these methods when using pseudotime created by trajectory inference tools.

With the use of mutual information, SCRIBE (Qiu et al. 2020) predicts regulations by estimating the strength of information transferred from a TF to its target gene. PIDC (Chan, Stumpf, and Babtie 2017) uses partial information decomposition (PID) to find out possible regulatory relationships between genes; specifically, it determines the regulation between a pair by introducing a third gene and compare PIDs among them. This comparison process shares similarity with 3DCEMA.

Few deep learning methods have been proposed for the task of GRN inference. CNNC (Yuan and Bar-Joseph 2019) uses a 2D convolutional neural network to predict regulations by classifying co-expression histograms of gene pairs. Good performance though it has compared to other non-deep methods, CNNC suffers from the distortion of the histograms caused by dropouts and extreme values. By contrast, 3DCEMA deals with extreme values properly and overcomes zero-inflation by introducing a third gene.

A comprehensive evaluation framework named BEELINE (Pratapa et al. 2020) has been developed in an attempt to assess both temporal and none-temporal GRN inference algorithms on accuracy, robustness and efficiency.

Problem Statement

ScRNA-seq provides a sparse gene expression matrix called A of g rows and c columns,

$$A^{g\times c} = \begin{bmatrix} a_{11} & \dots & a_{1c} \\ \vdots & \ddots & \vdots \\ a_{g1} & \dots & a_{gc} \end{bmatrix}$$

where g is the number of genes, c is the number of cells. a_{ij} is the expression value of the i-th gene in the j-th cell. Let G be the set of the genes, and R be the set of unknown regulatory edges. G and R make up the gene regulatory network $N = \langle G, R\rangle$ hidden behind scRNA-seq data. The task is to infer the regulations among the genes in G, with the supervision on a training graph N' and its gene expression matrix, whose gene set can be totally different from G.

Method

GRN inference is to find out all possible regulations between genes. To determine whether there is a regulation of gene x on gene y, one thought is to focus exactly on x and y by learning the regulatory patterns between them. However, a large number of false positives may occur when patterns of irrelevant gene pairs are all similar to them. Instead of predicting regulations directly, 3DCEMA focuses on the relative regulation of x on y to z given the 3D co-expression matrix of them. The GRN is finally inferred by synthesizing all predicted labels of matrices. In the following part, we will first present the process of building 3D matrices. We then demonstrate our unique labeling method. Finally the training and inference phases are introduced.

Building 3D Co-Expression Matrices

A 3D co-expression matrix E_{xyz} reflects the joint distribution of x, y, z in all cells. Here, we set the matrix size to $16 \times 16 \times 16$, which is a tradeoff between accurately preserving information and consuming appropriate amount of disk space. If the expression values of x, y and z are discretized into integers of $1, 2, ..., 16$, then the element e_{ijk} of E_{xyz} can be interpreted as the number of cells where the discrete expression levels of x, y and z are i, j and k correspondingly.

For every gene, the first task is to divide its range of expression values into 16 bins ranging from 1 to 16, with the bin index reflecting the levels of expression. However, extreme value points will stretch the whole range, causing normal value points squeezing together instead of uniformly distributed in all bins. The following logarithm step is to reduce the extreme points by mitigating the variation of gene expression values.

Given the gene expression matrix A with g genes and c cells, we replace every non-zero element of A by its logarithm, and for zero elements mainly caused by dropouts, we marked them as 'NaN'. Therefore we obtain matrix A':

$$A'^{g\times c} = \begin{bmatrix} a'_{11} & \dots & a'_{1c} \\ \vdots & \ddots & \vdots \\ a'_{g1} & \dots & a'_{gc} \end{bmatrix}$$

where

$$a'_{ij} = \begin{cases} ln(a_{ij}) & a_{ij} > 0 \\ \text{NaN} & a_{ij} = 0 \end{cases}.$$

Considering that extreme points still exist after logarithm, we calculate a limited range for each gene for binning. For the i-th row in A', which represents the expression values of the i-th gene, we calculate the mean m_i and standard deviation s_i of the non-'NaN' values, and use them to get the range of the i-th gene as follows.

$$R_i = [m_i - 2s_i, m_i + 2s_i)\,.$$

We then divide R_i into 16 bins with equal range:

$$R_{ik} = \left[m_i + \frac{(k-9)s_i}{4}, m_i + \frac{(k-8)s_i}{4}\right), k = 1, 2, ...16.$$

Algorithm 1 The co-expression matrix of the three genes.

Require: gene x, y and z, matrix B, constant p.
Ensure: 3D co-expression matrix $E_{xyz}^{16\times16\times16}$.

```
initialize E_xyz = 0;
for each cell n do
    get the integer expression values of the three genes in the n-th cell i = b_xn, j = b_yn, k = b_zn;
    record co-expression e_ijk += 1;
end for
for each element e_ijk of E_xyz do
    e_ijk = e_ijk * p/c;
end for
output E_xyz.
```

Given the discretization of all genes, we can rewrite the expression values as their bin index numbers, and obtain a bin-normalized matrix B:

$$B^{g\times c} = \begin{bmatrix} b_{11} & \dots & b_{1c} \\ \vdots & \ddots & \vdots \\ b_{g1} & \dots & b_{gc} \end{bmatrix}$$

where

$$b_{ij} = \begin{cases} k & a'_{ij} \in R_{ik} \\ 1 & a'_{ij} < m_i - 2s_i \text{ or } a'_{ij} = \text{NaN} \\ 16 & a'_{ij} \geq m_i + 2s_i \end{cases}.$$

Here b_{ij} is the integer expression level of the i-th gene in the j-th cell. The total time complexity of bin-normalization is $O(gc)$.

Based on the bin-normalized matrix B, we can build co-expression matrices of all gene triplets. Specifically, we scan every column of B to get the three integers of gene x, y and z in the cell, and use the three integers as index to find the element in E_{xyz}, and plus it by 1 to record one co-expression of the three genes. We then multiply every element by a factor so that the sum of all elements in a 3D matrix is a constant p, which is set to 4096 across our experiments. Algorithm 1 shows the building process in details. The time complexity of building a 3D co-expression matrix is $O(c)$.

Labeling

Once co-expression matrices are prepared, the next task is to design labels for the matrices. For a gene triplet of x, y and z, we view x as a potential regulator, with y and z being its targets. Here, we use mark $+$, $-$, and 0 to represent activation, inhibition and no regulation. We use nine basic labels:

$$00, +0, 0+, -0, 0-, ++, --, +-, -+$$

to distinguish different regulatory types of x on y and z. For instance, label $+0$ means that x has an activation regulation on y but no regulation on z.

Though exactly reflecting the regulation types, the nine basic classes are not suitable for the training and inference task, mainly because matrices of $++, --, +-, -+$ are too rare when the GRN is sparse. Therefore, more simplified and balanced labeling is needed.

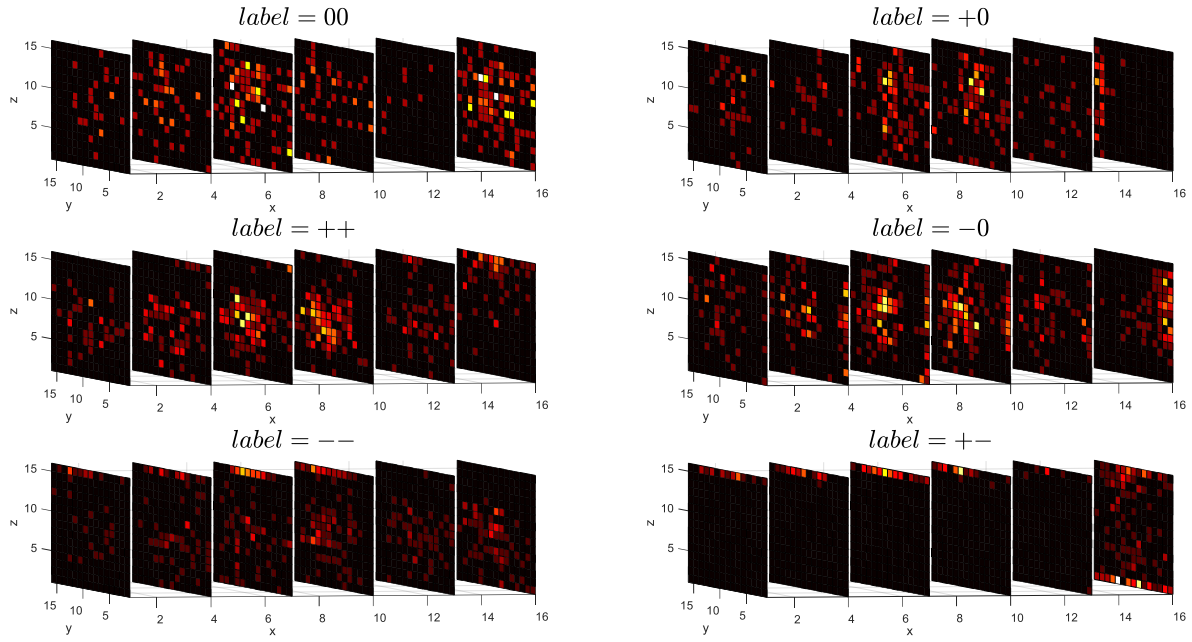

Figure 1: The heatmaps of 3D co-expression matrices of six different labels. The co-expression matrices are sliced along x-axis. The matrices displayed are built on the GSD datasets of BEELINE (Pratapa et al. 2020).

The Characteristics of the Nine Basic Classes In order to design such labels, we first examine the characteristics of the nine basic classes. Figure 1 shows the heat maps of 3D co-expression matrices of six different basic classes. Classes of $0+$, $0-$ and $-+$ are not included for they are symmetric to $+0$, -0, $+-$. Each 3D co-expression matrix are displayed by 6 squares along x axis, where the expression level of gene x increases from left to right. Each square reveals the joint distribution of y and z at a certain expression level of x.

We find out that in each class, the cluster of bright spots in the squares follows a unique pattern with the increase of x. For instance, in the matrix of class 00, there is a uniform distribution of spots in all squares, mainly because the change in x have no influence on y and z. While as to class $+0$ and -0, with the increase of x, the spots are moving towards left and right correspondingly, respectively illustrating the activation and inhibition regulation of x on y. Deduced by this pattern, the regulation of x on z will results in the spots moving towards top or bottom with the increase of x, which is illustrated in the matrices of class $++$, $--$ and $+-$ with the moving directions being top-left, bottom-right, bottom-left correspondingly. We conclude the moving directions of the nine classes in a clock shown in figure 2.

New Labels Reflecting Relative Regulations Here we introduce three new labels. We aggregate $0+, -+, -0$ as new label 'Y<Z', $00, ++, --$ as new label 'Y=Z', and $0-, +-, +0$ as new label 'Y>Z'. The decision boundary is the diagonal line of the YZ plane in the clock. Here Y and Z in the labels are not specific gene names, but represent the second and third genes of the matrix.

Given the regulation order $- < 0 < +$, the new labels have the semantic explanation: the relative regulation of the first gene on the second to the third. In contrast to the nine basic labels, The new labeling method is more simple and balanced.

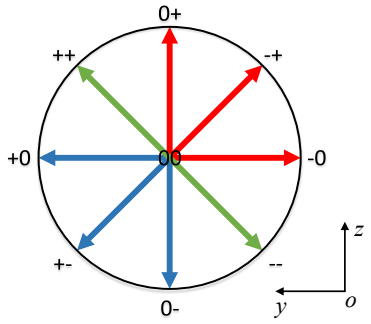

Figure 2: The clock of the nine basic labels. The arrow of each label shows the moving direction of the spots with the increase of x. Red, green and blue represent 3 new labels 'Y<Z', 'Y=Z', and 'Y>Z' correspondingly.

The final step is to examine whether the three labels contain enough information for the GRN inference task. Given a prediction 'Y>Z' for a 3D matrix by 3D CNN, we may be curious about what exact kind of regulations the first gene has on the second and the third, for there are three possible regulation types $0-, +-, +0$. However, if the predicted labels are mostly 'Y>Z' for set $\{E_{xyi} | i = 1, 2, ..., g\}$, meaning x has a relative activation regulation on y compared to every other gene i, we can confirm that there exists an acti-

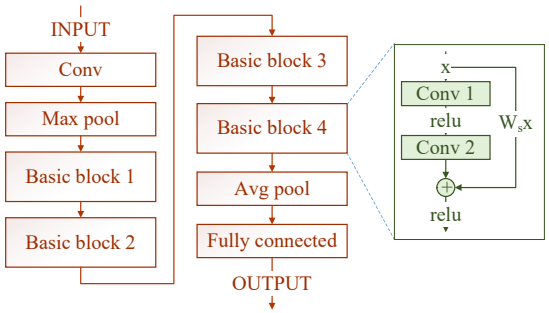

Figure 3: The 3D CNN based on ResNet-10 structure. Each basic block contains two convolutional layers and a shortcut.

vation regulation of x on y. Noticing that Y and Z play equal roles in labeling, we can also infer the regulation of the first gene on the third in a similar way.

Training and Inferring

3D CNN 3D CNN are mainly used in computer vision tasks such as video classification and motion detection. Here we use the 3D CNN to detect the moving directions of the spots along x axis. We follow ResNet-10 structure (He et al. 2016) to build the 3D CNN as shown in Figure 3. Each basic block contains two convolutional layers and a short cut, and is defined by He et al. as:

$$y = W_2\sigma\left(W_1 x\right) + W_s x.$$

Here x and y are the input and output vector, σ is the relu function, W_1 and W_2 are the weights of the first and second convolutional layer in the block, and W_s is a linear projection when the output planes of the two layers are different.

Training During the training process, it is desirable that we build all g^3 matrices, while they will devour disk memory rapidly when g is getting larger. Therefore, a limitation on the number of training matrices are required. Given a predetermined limit k for the number of matrices, the time and disk space complexities for building and storing training matrices are $O(ck)$ and $O(k)$, which show good scalability. Also, training time will not increase with g when the size limit k and the epoch number are fixed.

For sparse GRN, it is likely that the basic class of 00 dominates the whole set of matrices, causing 'Y=Z' being a major class. Therefore, before feeding matrices to the 3D CNN, adjusting the proportions of the classes is essential. The specific proportions are data-dependent and will be mentioned in the experiments. After adjusting the proportion of each class, we then feed matrices and their labels to the 3D CNN to complete training.

Inferring the GRN As for the inference part, we want to obtain a ranked edge list showing the confidence of an edge being a regulation. We use positive edge weights to indicate activation while the negative ones indicating inhibition. The absolute values of edge weights can be viewed as the confidence.

Algorithm 2 Synthesizing Labels to Infer the GRN.

Require: set of matrices S, trained CNN W.
Ensure: directed graph N.
1: initialize directed graph N with nodes being the genes and edge weights being all 0;
2: **for** every matrix T_{ijk} in S **do**
3: get $label$ predicted by W;
4: **if** $label$ ==’Y>Z’ **then**
5: $N(i,j)+=1, N(i,k)-=1$;
6: **end if**
7: **if** $label$ ==’Y<Z’ **then**
8: $N(i,j)-=1, N(i,k)+=1$;
9: **end if**
10: **end for**
11: output N.

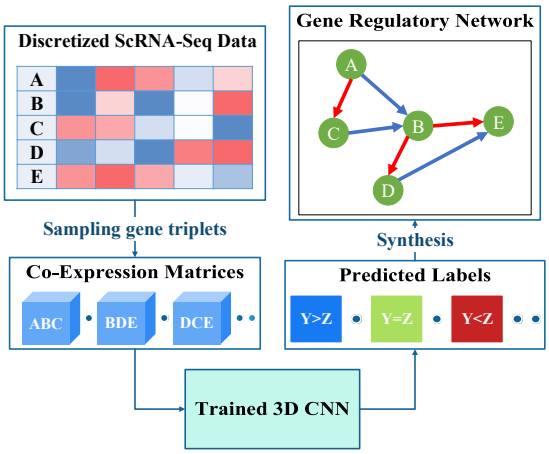

Figure 4: The whole GRN inference process.

First, We use the trained model to predict the labels of the matrices, and then we synthesize all the predicted labels to a GRN. The synthesis is simple: for every co-expression matrix E_{xyz}, if its label is 'Y>Z', we add n_{xy} by 1 and minus n_{xz} by 1, and reversely for label 'Y<Z', where n_{ij} represents the weight of the directed edge from i to j. Algorithm 2 shows the synthesis process. In this way, a final edge weight n_{xy} are determined by the predicted labels of $2g$ matrices from set $\{E_{xyi}|i=1,2,...,g\}$ and set $\{E_{xiy}|i=1,2,...,g\}$, therefore more stable and accurate prediction will be made. Figure 4 shows the whole GRN inference procedure.

When inferring a GRN, usually g^3 matrices are needed for determining all g^2 directed edges, which is unacceptable when g scales up. By randomly building k matrices from set $\{E_{xyi}|i=1,2...g\}$ to predict the regulation of x on y, we can reduce the total time complexity to $O(g^2ck)$. Another way is to compute a new 'gene' called 'AVG' to represent the average expressions of all genes. The regulation of each pair can be determined by only one comparison with 'AVG', and the total time complexity is reduced to $O(g^2c)$. In the inference process, a matrix can be stored in run-time memory

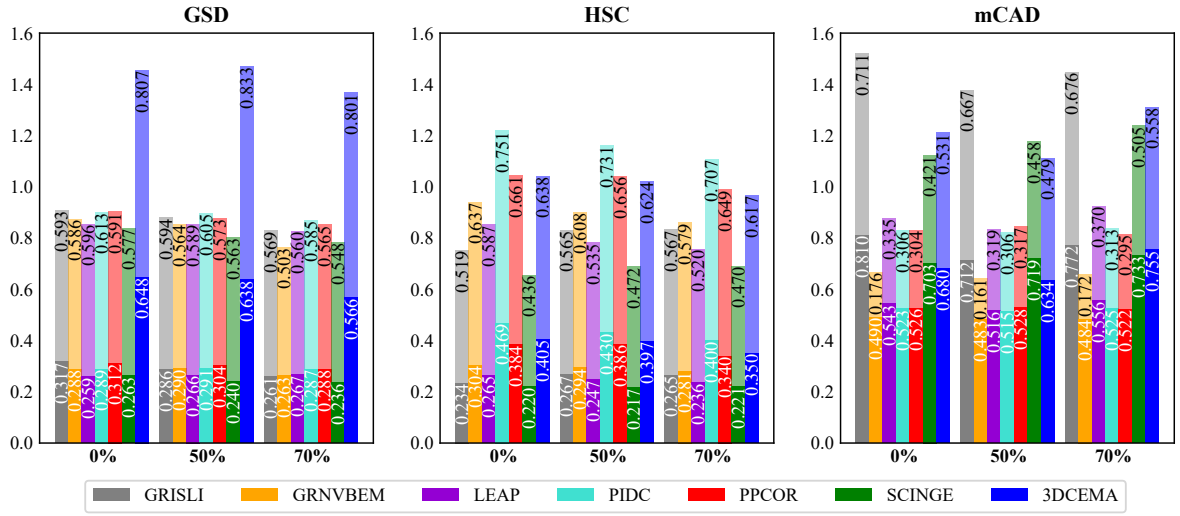

Figure 5: Results on three in-silico datasets under different dropout rates. The dark shade of the same color is AUPRC while the light shade is AUROC. The total height of each bar is the sum of the two metrics.

and released after the prediction, therefore no disk memory is needed for storing matrices.

Experiments

We conduct the experiments under the BEELINE framework (Pratapa et al. 2020), which benchmarks the algorithms of GRN inference using single-cell transcriptomic data. Three in-silico datasets and two scRNA-seq datasets are used to test the performance of 3DCEMA[1] and other seven algorithms.

Datasets Gonadal sex determination (GSD), hematopoietic stem cell (HSC) differentiation, and mammalian cortical area development (mCAD) are published boolean models, each representing a small GRN. BEELINE created in-silico datasets by randomly simulating each GRN for 10 times using BoolODE (Pratapa et al. 2020). The gene numbers of GSD, HSC and mCAD are 19, 11 and 5, and the cell numbers are 2000 for all simulations. 50 and 70 percent of dropout rates of matrix elements are applied, and the pseudotime information is provided.

Mouse embryonic stem cells (mESC) (Hayashi et al. 2018) and erythroid-lineage mouse hematopoietic stem cells (mHSC-E) (Nestorowa et al. 2016) are two scRNA-seq datasets. mHSC-E contains 4764 genes and 1071 cells, and its GRN is constructed from ChIP-Atlas database. mESC contains 18387 genes and 421 cells, with ESCAPE and ChIP-Atlas being the GRN source. Slingshot was used by BEELINE to create pseudotime for scRNA-seq datasets.

Algorithms for Comparison We examine 7 comparative algorithms, among which six are from BEELINE, namely GRISLI (Aubin-Frankowski and Vert 2020) applying ODEs and regressions, GRNVBEM (Sanchez-Castillo et al. 2017) basing on regressions, LEAP (Specht and Li 2016) and PPCOR (Kim 2015) utilizing correlations, PIDC (Chan, Stumpf, and Babtie 2017) employing mutual information, SCINGE (Deshpande et al. 2019) using Granger causality regressions. Besides, we include CNNC (Yuan and Bar-Joseph 2019), a deep learning method using 2D CNN to predict regulations by classifying 2D histograms of gene pairs. When testing, all 6 algorithms under the BEELINE framework output ranked edge lists where weights are confidence of regulations. For CNNC, we use the value of the last sigmoid layer of the CNN as the edge weight for a given gene pair.

Experiments on In-Silico Datasets

Settings Here, 3DCEMA uses the labels of 'Y<Z','Y=Z' and 'Y>Z' for training, The matrices numbers of the three classes are set to the same. We separately trained our model on GSD-1, GSD-1-50 and GSD-1-70, meaning the 1st simulated data of GSD with 0, 50% and 70% dropout rates. We obtained the trained model weights named as w_1, w_2 and w_3. The model weights w_1, w_2 and w_3 were then used individually to reconstruct the GRNs on simulated datasets with dropout rate of 0, 50% and 70% correspondingly. Area under PRC curve (AUPRC), Area under ROC curve (AUROC) are computed for all edge lists with the knowledge of the ground-truth network.

Given that the gene numbers of in-silico datasets are relatively small, the total g^2 matrices are too few to train CNNC, therefore we exclude CNNC here.

3DCEMA Shows Strong Stability across GRNs Figure 5 shows the results on in-silico datasets. 3DCEMA performs the best both on AUPRC and AUROC over the simulated datasets of GSD, which is the GRN used for training. On HSC datasets, PIDC, PPCOR and 3DCEMA are the three algorithms that perform best, while GRISLI, SCINGE and

[1]Code is avaliable at https://github.com/YueFan1014/3DCEMA.

	mESC 200			mESC 400			mESC 1000		
	AUPRC	**AUROC**	**k-Precision**	**AUPRC**	**AUROC**	**k-Precision**	**AUPRC**	**AUROC**	**k-Precision**
GRISLI	0.0951	0.5138	0.0977	0.1299	0.5062	0.1141	-	-	0.0614
GRNVBEM	0.0992	0.4998	0.0989	0.1337	0.5011	0.1327	-	-	0.0654
LEAP	0.0556	0.2056	0.0176	0.0829	0.2528	0.0347	-	-	0.0168
PIDC	0.0885	0.4331	0.0944	0.1230	0.4228	0.1155	-	-	0.0690
PPCOR	0.0960	0.5068	0.0955	0.1311	0.4995	0.1317	-	-	0.0782
SCINGE	0.0967	0.5089	0.1164	0.1340	0.5031	0.2027	-	-	0.1238
CNNC	0.1964	0.6945	0.2580	0.2670	0.7162	0.3166	-	-	0.1483
3DCEMA	**0.3427**	**0.8281**	**0.3787**	**0.3059**	**0.7720**	**0.3392**	-	-	**0.3681**
	mHSC-E 200			**mHSC-E 400**			**mHSC-E 1000**		
	AUPRC	**AUROC**	**k-Precision**	**AUPRC**	**AUROC**	**k-Precision**	**AUPRC**	**AUROC**	**k-Precision**
GRISLI	0.0439	0.5014	0.0448	0.0441	0.5126	0.0463	-	-	0.0252
GRNVBEM	0.0441	0.5000	0.0330	0.0435	0.5000	0.0452	-	-	0.0248
LEAP	0.0231	0.0793	0.0122	0.0231	0.1046	0.0090	-	-	0.0248
PIDC	0.0329	0.3815	0.0142	0.0321	0.3813	0.0104	-	-	0.0179
PPCOR	0.0420	0.4921	0.0330	0.0419	0.4944	0.0405	-	-	0.0265
SCINGE	0.0472	0.5119	0.0590	0.0870	0.5374	0.1523	-	-	0.0729
CNNC	0.0771	0.5357	0.1958	0.3732	**0.8983**	0.3920	-	-	0.1121
3DCEMA	**0.2682**	**0.8776**	**0.3491**	**0.4010**	0.8302	**0.4505**	-	-	**0.2395**

Table 1: Results on scRNA-seq datasets.

3DCEMA are the top three algorithms on mCAD datasets. Although mCAD and HSC differ from GSD significantly in both regulation types and network topology, by learning regulation patterns from GSD, 3DCEMA is still capable of the inference task on HSC and mCAD, showing itself being transferable across datasets with different GRN properties. By contrast, PIDC and PPCOR have poor performances on mCAD though they are among the top three algorithms on HSC; GRISLI and SCINGE performs badly on HSC despite that they predict well on mCAD. Therefore, 3DCEMA shows strong prediction stability across all datasets, though its performances are not the best on HSC and mCAD due to limited training materials caused by small g.

Experiments on ScRNA-seq Datasets

Settings For the large scRNA-seq datasets, the ground-truth regulation edges are listed without specifying activation or inhibition. Therefore we change 3DCEMA into a binary classifier with label 1, 0 representing regulations and no regulations. We compute a new 'gene' called 'AVG' to represent the average expressions of all genes. For a given pair x, y, 3D matrices are built with three dimensions of x, y and 'AVG'. We choose 200, 400, 1000 genes from mESC and mHSC-E to build 6 datasets. In each dataset, genes are divided into two sets without intersection, with one for training and the other for testing. In the training phase, the proportions of class 0 and 1 are set to 0.8 and 0.2. In the testing process, the last layer outputs are used as the predicted edge weights. Apart from AUPRC and AUROC, we adopt the precision of the top-k confidence edges, with k equal to the number of regulations in the ground-truth GRNs. AUPRC and AUROC are not calculated on datasets with 1000 genes due to high computational cost.

3DCEMA Recalls More Regulations Significantly Table 1 shows the results on scRNA-seq datasets. Our algorithms outperforms all the competitive methods including the deep learning method CNNC. We observed that the 6 algorithms from BEELINE performs poorly on all datasets, mainly because of the sparsity of the two GRNs. Deep learning methods of CNNC and 3DCEMA, however, learn the regulation patterns of co-expression matrices from the training sets which are sub-graphs of the sparse GRNs, and achieve higher top-k precision on the testing sets.

Between the two deep learning methods, 3DCEMA shows better performance, especially on datasets with 1000 genes. That is because in 3DCEMA, 'AVG' serves as a reference of average gene expressions and noises, which is especially useful when g is large. While as to CNNC, noises will disturb the co-expression histogram of x and y and affect the final prediction results. By using a third dimension for comparison, 3DCEMA can avoid this disturbance.

Among the top-k edges inferred by 3DCEMA on scRNA-seq data, we find regulations absent from the ground-truth database, but appear in other sources such as RegNetwork database (Liu et al. 2015). This indicates that 3DCEMA would serve as a capable tool for inferring potential regulations other than recalling existing ones, and would significantly increase the chances for researchers to find real regulations with high confidence.

Summary

We proposed a deep learning method called 3D co-expression matrix analysis (3DCEMA), which uses 3D convolutional neural network to predict gene regulations by classifying 3D co-expression matrices of gene triplets. The unique labels and the third reference genes help to alleviate the influence of noises and zeros on prediction tasks. Techniques that increase scalability enables 3DCEMA to process scRNA-seq data with large gene and cell numbers. 3DCEMA outperforms other existing algorithms both in stability and accuracy, and could serve as a reliable tool that enables researchers to recognize gene regulations more efficiently.

References

Aubin-Frankowski, P.-C.; and Vert, J.-P. 2020. Gene regulation inference from single-cell RNA-seq data with linear differential equations and velocity inference. *Bioinformatics* 36(18): 4774–4780.

Chan, T. E.; Stumpf, M. P.; and Babtie, A. C. 2017. Gene Regulatory Network Inference from Single-Cell Data Using Multivariate Information Measures. *Cell Systems* 5(3): 251 – 267.e3.

Chasman, D.; Fotuhi Siahpirani, A.; and Roy, S. 2016. Network-based approaches for analysis of complex biological systems. *Current Opinion in Biotechnology* 39: 157 – 166.

Deng, Q.; Ramsköld, D.; Reinius, B.; and Sandberg, R. 2014. Single-Cell RNA-Seq Reveals Dynamic, Random Monoallelic Gene Expression in Mammalian Cells. *Science* 343(6167): 193–196.

Deshpande, A.; Chu, L.-F.; Stewart, R.; and Gitter, A. 2019. Network Inference with Granger Causality Ensembles on Single-Cell Transcriptomic Data. *bioRxiv* doi:10.1101/534834.

Hayashi, T.; Ozaki, H.; Sasagawa, Y.; Umeda, M.; Danno, H.; and Nikaido, I. 2018. Single-cell full-length total RNA sequencing uncovers dynamics of recursive splicing and enhancer RNAs. *Nature Communications* 9(1): 619.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Huynh-Thu, V.; Irrthum, A.; Wehenkel, L.; and Geurts, P. 2010. Inferring Regulatory Networks from Expression Data Using Tree-Based Methods. *PLoS ONE* 5(9): e12776.

Kelly, S.; Davide, R.; Fletcher, R. B.; Diya, D.; John, N.; Nir, Y.; Elizabeth, P.; and Sandrine, D. 2018. Slingshot: Cell lineage and pseudotime inference for single-cell transcriptomics. *BMC Genomics* 19(1): 477.

Kim, S. 2015. ppcor: An R Package for a Fast Calculation to Semi-partial Correlation Coefficients. *Communications for statistical applications and methods* 22(6): 665–674.

Kolodziejczyk, A.; Kim, J. K.; Svensson, V.; Marioni, J.; and Teichmann, S. 2015. The Technology and Biology of Single-Cell RNA Sequencing. *Molecular Cell* 58(4): 610 – 620.

Liu, Z.-P.; Wu, C.; Miao, H.; and Wu, H. 2015. RegNetwork: an integrated database of transcriptional and post-transcriptional regulatory networks in human and mouse. *Database* 2015. doi:10.1093/database/bav095.

Matsumoto, H.; Kiryu, H.; Furusawa, C.; Ko, M. S. H.; Ko, S. B. H.; Gouda, N.; Hayashi, T.; and Nikaido, I. 2017. SCODE: an efficient regulatory network inference algorithm from single-cell RNA-Seq during differentiation. *Bioinformatics* 33(15): 2314–2321.

Moerman, T.; Aibar Santos, S.; Bravo González-Blas, C.; Simm, J.; Moreau, Y.; Aerts, J.; and Aerts, S. 2018. GRNBoost2 and Arboreto: efficient and scalable inference of gene regulatory networks. *Bioinformatics* 35(12): 2159–2161.

Nam, A. S.; Chaligne, R.; and Landau, D. A. 2021. Integrating genetic and non-genetic determinants of cancer evolution by single-cell multi-omics. *Nature Reviews Genetics* 22(1): 3–18.

Nestorowa, S.; Hamey, F. K.; Pijuan Sala, B.; Diamanti, E.; Shepherd, M.; Laurenti, E.; Wilson, N. K.; Kent, D. G.; and Göttgens, B. 2016. A single-cell resolution map of mouse hematopoietic stem and progenitor cell differentiation. *Blood* 128(8): e20–e31.

Papili Gao, N.; Ud-Dean, S. M. M.; Gandrillon, O.; and Gunawan, R. 2017. SINCERITIES: inferring gene regulatory networks from time-stamped single cell transcriptional expression profiles. *Bioinformatics* 34(2): 258–266.

Pratapa, A.; Jalihal, A. P.; Law, J. N.; Bharadwaj, A.; and Murali, T. M. 2020. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nature Methods* 17(2): 147–154.

Qiu, X.; Rahimzamani, A.; Wang, L.; Ren, B.; Mao, Q.; Durham, T.; McFaline-Figueroa, J. L.; Saunders, L.; Trapnell, C.; and Kannan, S. 2020. Inferring Causal Gene Regulatory Networks from Coupled Single-Cell Expression Dynamics Using Scribe. *Cell Systems* 10(3): 265 – 274.e11.

Sanchez-Castillo, M.; Blanco, D.; Tienda-Luna, I. M.; Carrion, M. C.; and Huang, Y. 2017. A Bayesian framework for the inference of gene regulatory networks from time and pseudo-time series data. *Bioinformatics* 34(6): 964–970.

Specht, A. T.; and Li, J. 2016. LEAP: constructing gene co-expression networks for single-cell RNA-sequencing data using pseudotime ordering. *Bioinformatics* 33(5): 764–766.

Tang, F.; Barbacioru, C.; Wang, Y.; Nordman, E.; Lee, C.; Xu, N.; Wang, X.; Bodeau, J.; Tuch, B. B.; Siddiqui, A.; Lao, K.; and Surani, M. A. 2009. mRNA-Seq whole-transcriptome analysis of a single cell. *Nature Methods* 6(5): 377–382.

Tasic, B.; Menon, V.; Nguyen, T. N.; Kim, T. K.; Jarsky, T.; Yao, Z.; Levi, B.; Gray, L. T.; Sorensen, S. A.; Dolbeare, T.; Bertagnolli, D.; Goldy, J.; Shapovalova, N.; Parry, S.; Lee, C.; Smith, K.; Bernard, A.; Madisen, L.; Sunkin, S. M.; Hawrylycz, M.; Koch, C.; and Zeng, H. 2016. Adult mouse cortical cell taxonomy revealed by single cell transcriptomics. *Nature Neuroscience* 19(2): 335–346.

Xie, X.; Shi, Q.; Wu, P.; Zhang, X.; Kambara, H.; Su, J.; Yu, H.; Park, S.-Y.; Guo, R.; Ren, Q.; Zhang, S.; Xu, Y.; Silberstein, L. E.; Cheng, T.; Ma, F.; Li, C.; and Luo, H. R. 2020. Single-cell transcriptome profiling reveals neutrophil heterogeneity in homeostasis and infection. *Nature Immunology* 21(9): 1119–1133.

Yuan, Y.; and Bar-Joseph, Z. 2019. Deep learning for inferring gene relationships from single-cell expression data. *Proceedings of the National Academy of Sciences* 116(52): 27151–27158.