Bioinformatics, 2025, **41(6)**, btaf314
https://doi.org/10.1093/bioinformatics/btaf314
Advance Access Publication Date: 23 May 2025
Original Paper

OXFORD

Gene expression

TRENDY: gene regulatory network inference enhanced by transformer

Xueying Tian[1], Yash Patel[2], Yue Wang[3,*]

[1]School of Information, University of California, Berkeley, CA 94720, United States
[2]Department of Mathematics, University of Miami, Coral Gables, FL 33146, United States
[3]Irving Institute for Cancer Dynamics and Department of Statistics, Columbia University, New York, NY 10027, United States

*Corresponding author. Irving Institute for Cancer Dynamics and Department of Statistics, Columbia University, New York, NY 10027, United States. E-mail: yw4241@columbia.edu.

Associate Editor: Laura Cantini

Abstract

Motivation: Gene regulatory networks (GRNs) play a crucial role in the control of cellular functions. Numerous methods have been developed to infer GRNs from gene expression data, including mechanism-based approaches, information-based approaches, and more recent deep learning techniques, the last of which often overlook the underlying gene expression mechanisms.

Results: In this work, we introduce TRENDY, a novel GRN inference method that integrates transformer models to enhance the mechanism-based WENDY approach. Through testing on both simulated and experimental datasets, TRENDY demonstrates superior performance compared to existing methods. Furthermore, we apply this transformer-based approach to three additional inference methods, showcasing its broad potential to enhance GRN inference.

Availability and implementation: Code and data files are available at https://github.com/YueWangMathbio/TRENDY, with DOI: 10.6084/m9.figshare.28236074.

1 Introduction

The expression of genes can be regulated by other genes. We can construct a directed graph, where vertices are genes, and edges are regulatory relationships. This graph is called a gene regulatory network (GRN). GRNs are essential to understanding the mechanisms governing cellular processes, including how cells respond to various stimuli, differentiate, and maintain homeostasis (Wang 2020, Wang *et al.* 2022). Understanding the GRN structure is important for developmental biology (Myasnikova and Spirov 2020, Wang *et al.* 2020, Cheng *et al.* 2022, 2024, Sha *et al.* 2024), disease research (Arshad *et al.* 2016, Angelini *et al.* 2022, Cheng *et al.* 2023, McDonald *et al.* 2023) and even studying macroscopic behavior (Li *et al.* 2021, Vijayan *et al.* 2022, Axelrod *et al.* 2023).

It is difficult to directly determine the GRN structure with experiments. Instead, researchers develop methods that infer the GRN structure through gene expression data. Some methods, such as WENDY (Wang *et al.* 2024) and NonlinearODEs (Ma *et al.* 2020), are mechanism-based: they construct dynamical models for gene expression, and fit with expression data to determine the GRN (Burdziak *et al.* 2023, Wang *et al.* 2023). Some methods, like GENIE3 (Huynh-Thu *et al.* 2010) and SINCERITIES (Papili Gao *et al.* 2018), are information-based: they treat this as a feature selection problem and directly find genes that can be used to predict the level of the target gene (e.g., by linear regression). Then this predictability (information) for the target gene is assumed to imply regulation (Huynh-Thu and Geurts 2018, Zheng *et al.* 2019).

Recently, there are some deep learning-based methods that infer GRN with different neural network frameworks, such as convolutional neural networks (Nauta *et al.* 2019), recurrent neural networks (Kentzoglanakis and Poole 2012), variational autoencoders (Shu *et al.* 2021), graph neural networks (Feng *et al.* 2023, Mao *et al.* 2023), etc. Deep learning frameworks can perform well on many tasks. However, because of the large number of tunable parameters in a neural network, it is difficult to explain why it works. These deep learning-based GRN inference methods have similar problems: they generally apply popular neural networks as black boxes without integration with biological mechanisms, and thus lacking interpretability.

The transformer model is a deep learning architecture designed to handle sequential data (Vaswani *et al.* 2017). It can capture long-range dependencies and relationships (linear and nonlinear) through the self-attention mechanism. The encoder layer of the transformer model can process the input through a neural network, so that the output is close to the given target.

Some researchers have developed GRN inference methods based on the transformer model (Xu *et al.* 2023) or its variants (Shu *et al.* 2022). The STGRNS method (Xu *et al.* 2023) predicts the existence of regulation between two genes solely based on the expression levels of these two genes.

Received: 1 November 2024; **Revised:** 19 January 2025; **Editorial Decision:** 15 May 2025; **Accepted:** 19 May 2025

Since other genes are not considered, this approach makes it difficult to distinguish between direct regulation ($G_i \to G_j$) and indirect regulation through a third gene ($G_i \to G_k \to G_j$). The GRN-transformer method (Shu *et al.* 2022) uses multiple inputs, including the GRN inferred by an information-based method, PIDC (Chan *et al.* 2017), but it also does not consider the biological dynamics of gene regulation.

To combine the powerful deep learning techniques and biological understanding of gene regulation, we propose a novel GRN inference method that applies transformer models to enhance a mechanism-based GRN inference method, WENDY. This new method, TRansformer-Enhanced weNDY, is abbreviated as TRENDY. WENDY method is based on a dynamical model for gene regulation, and an equation for the GRN and the covariance matrix of genes is solved to derive the GRN. In TRENDY, we first use a transformer model to construct a pseudo-covariance matrix that performs better in WENDY. Then we apply another transformer model that directly enhances the inferred GRN.

The idea for the second half of TRENDY can be used to enhance the inferred results by other GRN inference methods. We apply transformer models to three GRN inference methods, GENIE3, SINCERITIES, and NonlinearODEs, to obtain their transformer-enhanced versions, tGENIE3, tSINCERITIES, and tNonlinearODEs.

There have been some research papers that study the idea of denoising networks (e.g. enhancing GRNs inferred by other methods), such as network deconvolution (ND) (Feizi *et al.* 2013), diffusion state distance (Cao *et al.* 2013), BRANE Cut (BC) (Pirayre *et al.* 2015), BRANE Clust (Pirayre *et al.* 2018), and network enhancement (Wang *et al.* 2018).

We test the four traditional methods (WENDY, GENIE3, SINCERITIES, and NonlinearODEs) with their enhanced versions (by transformer, ND, or BC) on two simulated data sets and two experimental data sets. All four transformer-enhanced methods outperform other methods, and TRENDY ranks the first among all methods.

In order to train a deep learning model, such as the transformer model used in this article, we need a sufficient amount of training data. Specifically, since we want the GRN inference method to work well in different situations, the training data should contain many different data sets that correspond to different GRNs. In reality, there are not many available experimental gene expression data sets with known GRNs. Therefore, we consider generating synthetic data sets.

To simulate new gene expression data, the most common approach is to assume that gene expression follows a certain mechanism, such as an ordinary differential equation (ODE) system (Hache *et al.* 2009, Kamimoto *et al.* 2023) or a stochastic differential equation (SDE) system (Schaffter *et al.* 2011, Dibaeinia and Sinha 2020). After determining the parameters of this system, one can simulate a discretized version of this system and obtain new data. Readers may refer to a review (Tripathi *et al.* 2017) for this mechanism approach. Another approach is to apply deep learning to learn from experimental gene expression data and generate new data that mimic the existing data. Such generative neural networks can be variational autoencoders (Shu *et al.* 2021) or generative adversarial networks (Marouf *et al.* 2020).

The advantage of the differential equation approach is that one can change the parameters to generate new data that correspond to another GRN, while the deep learning approach can only reproduce existing data. The disadvantage of the differential equation approach is that the modeled mechanism might not be a perfect fit to reality. Therefore, the generated data might be questionable, especially those generated by an oversimplified model, such as a linear ODE system. Instead, the deep learning approach can generate new data that are indistinguishable from the experimental data.

In this article, we need to feed the transformer models with data from many different GRNs. Therefore, we adopt the mechanism approach with a frequently used nonlinear and stochastic system (Pinna *et al.* 2010, Papili Gao *et al.* 2018, Wang *et al.* 2024):

$$\mathrm{d}X_j(t) = V\left\{\beta\prod_{i=1}^{n}[1+(A_{\text{true}})_{i,j}\frac{X_i(t)}{X_i(t)+1}]-\theta X_j(t)\right\}\mathrm{d}t + \sigma X_j(t)\mathrm{d}W_j(t). \quad (1)$$

The matrix A_{true} is a randomly generated ground truth GRN, where $(A_{\text{true}})_{i,j} > 0/=0/<0$ means that gene i has positive/no/negative regulatory effects on gene j. $X_i(t)$ is the expression level of gene i at time t, and $W_j(t)$ is a standard Brownian motion. For different genes, the corresponding Brownian motions are independent. The same as in previous papers (Pinna *et al.* 2010, Papili Gao *et al.* 2018, Wang *et al.* 2024), the parameter values are $V = 30$, $\beta = 1$, $\theta = 0.2$, and $\sigma = 0.1$. Our goal is to infer A_{true} from $X_j(t)$.

Most deep learning-based GRN inference methods are trained on experimental data sets, whose amount is very limited. Therefore, such methods need to carefully choose the neural network structure and the training procedure under this limitation. In this article, we consider a scenario in which the training data set is sufficiently large, and study how deep learning can achieve the best performance without data limitation.

2 Results

2.1 TRENDY method

WENDY method (Wang *et al.* 2024) uses single-cell gene expression data measured at two time points, each of size $m \times n$ (mRNA counts of n genes for m cells). Here we do not know how cells at different time points correspond to each other, since the measurement of single-cell level gene expression needs to kill the cells (Wang and Wang 2022). For data at two time points, the cell numbers m can be different.

For gene expression data at time points 0 and t, WENDY applies graphical lasso (Friedman *et al.* 2008) to calculate the covariance matrices for different genes, K_0 and K_t, each of size $n \times n$. We can construct a general SDE system like Equation (1) to model the dynamics of gene regulation. After linearization of this system, one can obtain an approximated relation for K_0, K_t, and the ground truth GRN A_{true}, also of size $n \times n$:

$$K_t = (I + tA_{\text{true}}^{\mathrm{T}})K_0(I + tA_{\text{true}}) + D + E. \quad (2)$$

Here I is an $n \times n$ identity matrix, D is an unknown $n \times n$ diagonal matrix, and E is the unknown error introduced by linearization. Then WENDY solves the GRN matrix A as a non-convex optimization problem:

$$\arg\min_A f(A) := \frac{1}{2}\sum_{i\neq j}\{[K_t - (I+tA^{\mathrm{T}})K_0(I+tA)]_{i,j}\}^2. \quad (3)$$

Due to the unknown diagonal matrix D, this summation does not count elements with $i=j$. The result of this optimization problem is denoted as

$$A_0 = \mathrm{WENDY}(K_0, K_t).$$

See Section 1, available as supplementary data at *Bioinformatics* online for details of WENDY.

If WENDY works perfectly, then

$$K_t = (I+tA_0^{\mathrm{T}})K_0(I+tA_0) + D,$$

meaning that $(I+tA_{\mathrm{true}}^{\mathrm{T}})K_0(I+tA_{\mathrm{true}})$ and $(I+tA_0^{\mathrm{T}})K_0(I+tA_0)$ differ by E, the error introduced by the linearization, and A_0 may not match A_{true} accurately. If we want to derive a more accurate and complicated equation of K_0, K_t, and A_{true} than Equation (2), this equation might be numerically unstable to solve. Therefore, we need to find another way to improve WENDY.

Define

$$K_t^* = (I+tA_{\mathrm{true}}^{\mathrm{T}})K_0(I+tA_{\mathrm{true}}). \quad (4)$$

If we replace K_t by K_t^* in Equation (3) and solve A, then this inferred A should be very close to A_{true}. However, in practice, we only have K_0 and K_t, but not K_t^*, since A_{true} is unknown.

The core idea is to train a transformer model TE($k=1$) with input K_t, so that the output K_t' is close to K_t^*. Then we can calculate

$$A_1 = \mathrm{WENDY}(K_0, K_t'),$$

which is closer to A_{true} than A_0. We can further improve A_1 using another transformer model TE($k=3$), along with the information in K_0, K_t, so that the output A_2 is even closer to A_{true}. These two transformer models will be discussed in the next subsection.

The actual K_t is a complicated nonlinear function of A_{true} and K_0. Thus K_t^*, also a function of A_{true} and K_0, can be roughly learned by the TE($k=1$) model as a function of K_t. Given accurate K_0 and K_t^*, we still cannot uniquely determine A_{true}, and the numerical solver for Equation (3) determines which A_1 is chosen. Therefore, the TE($k=3$) model can partially learn the nonlinear behavior of the solver and output a better A_2 from A_1.

Figure 1, Algorithms 1 and 2 describe the training and testing workflow for TRENDY.

2.2 Transformer model TE(k)

We build a deep learning framework TE(k) that is based on transformer encoder layers, where k is the major hyperparameter that describes the number of input matrices. Besides, there are three hyperparameters that can be tuned: the model dimension d; the number of encoder layers l; the number of attention heads h of the encoder layer. We use this TE(k) model with different hyperparameters in TRENDY and other transformer-enhanced methods. See Algorithm 3 for the general structure of TE(k), along with the shape of data after each layer. Besides the standard transformer structure, the segment embedding layer tells the model to treat $k>1$ input matrices separately, and the 2-D positional encoding layer tells the model to remember that the input is originally two-dimensional. Notice that the same transformer encoder layer can handle inputs of different lengths. This means that the gene number n is not a predetermined hyperparameter, and we do not need to train a different model for each n. We will train the TE(k) model with $n=10$ genes and test on data sets with $n=10/18/20$ genes.

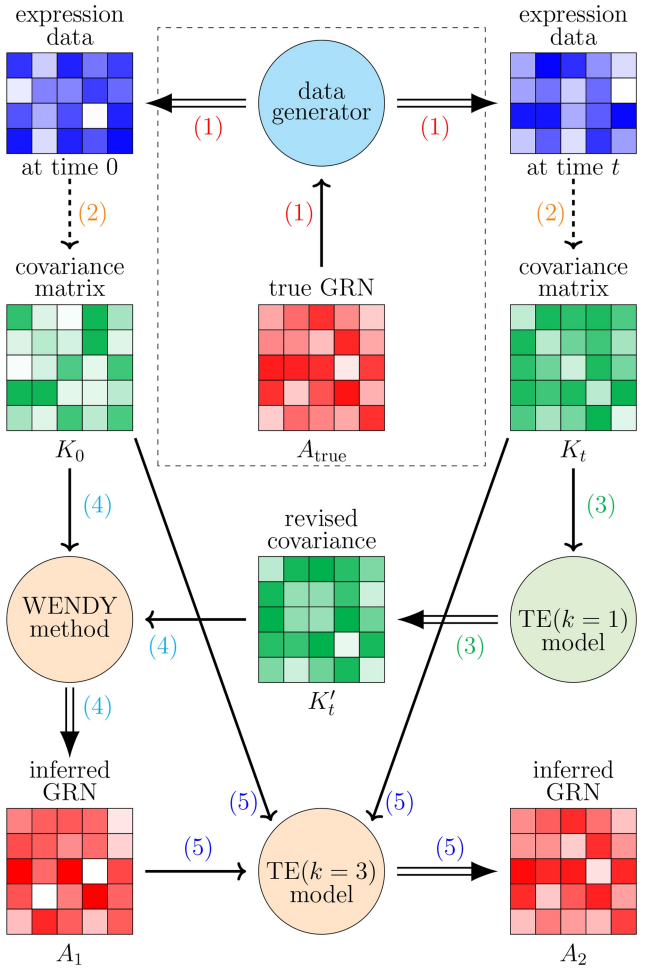

Figure 1. Workflow of training and testing the TRENDY method. Rectangles with grids are numerical matrices, where the color scale represents the numerical value. Circles are mechanisms that take in matrices and produce matrices. Single and double arrows represent inputs and outputs of each mechanism, and dashed arrows represent direct calculations. To apply TRENDY after training, the step in the dashed rectangle is omitted.

For the first half of the TRENDY method, the TE($k=1$) model has $k=1$ input matrix K_t, $d=64$ model dimension, $l=7$ encoder layers, and $h=4$ attention heads. For the second half of the TRENDY method, the TE($k=3$) model has $k=3$ input matrices of the same size, A_1, K_0, K_t, $d=64$ model dimension, $l=7$ encoder layers, and $h=8$ attention heads.

2.3 Other transformer-enhanced methods

The second half of the TRENDY method applies a transformer model to learn the highly nonlinear relation between

Algorithm 1 Training workflow of TRENDY method.

1. **Repeat** generating random A_{true} and corresponding gene expression data
2. **Calculate** covariance matrices K_0 and K_t, and then calculate K_t^* from $K_0, K_t, A_{\text{true}}$
3. **Train** TE($k=1$) model with input K_t and target K_t^* **Call** trained TE($k=1$) model to calculate K_t' from K_t
4. **Call** WENDY to calculate A_1 from K_0, K_t'
5. **Train** TE($k=3$) model with input A_1, K_0, K_t and target A_{true}

Algorithm 2 Testing workflow of TRENDY method.

1. **Input**: gene expression data at two time points
2. **Calculate** covariance matrices K_0 and K_t
3. **Call** trained TE($k=1$) model to calculate K_t' from K_t
4. **Call** WENDY to calculate A_1 from K_0, K_t'
5. **Call** trained TE($k=3$) model to calculate A_2 from A_1, K_0, K_t
6. **Output**: inferred GRN A_2

Algorithm 3 Structure of the TE(k) model. The shape of data after each layer is in the brackets.

1. **Input**: k matrices (k groups of $n\times n$)
2. Linear embedding layer with dimension d (k groups of $n\times n\times d$)
3. Segment embedding layer (omitted when $k=1$) (k groups of $n\times n\times d$)
4. 2-D positional encoding layer (k groups of $n\times n\times d$)
5. Flattening and concatenation ($(n^2)\times(dk)$)
6. l layers of transformer encoder ($(n^2)\times(dk)$)
7. Linear embedding layer ($n^2\times 1$)
8. **Output**: reshaping into a matrix ($n\times n$)

A_1 (result of WENDY) and A_{true}. For other GRN inference methods, we can assume that the inferred GRN and A_{true} have some nonlinear and possibly random relations, since these methods only consider and infer certain forms of regulations. Then we can try to enhance the inferred results by a transformer model.

GENIE3 (Huynh-Thu *et al.* 2010) works on single-cell expression data at one time point. It uses random forest to select genes that have predictability for the target gene, thus being information-based.

SINCERITIES (Papili Gao *et al.* 2018) works on single-cell expression data at multiple time points. It runs regression on the Kolmogorov–Smirnov distance between cumulative distribution functions of expression levels, also being information-based.

NonlinearODEs (Ma *et al.* 2020) works on bulk expression data at multiple time points. This method fits the data with a nonlinear differential equation system, which is thus mechanism-based.

For each of these three methods, we train a TE(k) model to enhance the inferred results, similar to the TE($k=3$) model in the second half of TRENDY. Here each TE(k) model has $d=64, l=7, h=8$, and the training target is A_{true}.

For inferred GRN A_{G} by GENIE3, we train a TE($k=2$) model with A_{G} and the covariance matrix K as input. Then we can use this trained model to calculate a more accurate A_{G}' from A_{G} and K. This method is named tGENIE3.

For inferred GRN A_{S} by SINCERITIES, we train a TE($k=1$) model with A_{S} as input, and name it tSINCERITIES.

For inferred GRN A_{N} by NonlinearODEs, we train a TE($k=1$) model with A_{N} as input, and name it tNonlinearODEs.

2.4 Methods enhanced by ND and BC

Network deconvolution (ND) (Feizi *et al.* 2013) requires the input to be nonnegative and symmetric, so as its output. The idea is to assume that the input is a convolution of a simpler network, and solves this simpler network by deconvolution. BRANE Cut (BC) (Pirayre *et al.* 2015) performs well when we know that some genes are transcription factors and others are not. When we have no knowledge of transcription factors, BC degenerates to setting a threshold for whether one regulation exists. BC requires the input to be nonnegative and symmetric, and the output is a 0–1 matrix. We use ND and BC to enhance those four traditional methods, and name them as nWENDY, nGENIE3, nSINCERITIES, nNonlinearODEs, bWENDY, bGENIE3, bSINCERITIES, bNonlinearODEs.

2.5 Performance of different methods

Certain autoregulations can decrease the variance (Ramos and Reinitz 2019, Giovanini *et al.* 2020, Holehouse *et al.* 2020), and mRNA bursts can increase the gene expression variance (Paré *et al.* 2009, Bothma *et al.* 2014, Leyes Porello *et al.* 2023), which can be approximated by tuning the value of σ in Equation (1). SINC data set is generated by Equation (1) with $\sigma = 0.01/0.1/1$. The synthetic DREAM4 data set (Marbach *et al.* 2012) is commonly used as a benchmark for GRN inference (Papili Gao *et al.* 2018, Wang *et al.* 2024). THP-1 data set measures monocytic THP-1 human myeloid leukemia cells (Kouno *et al.* 2013). hESC data set measures human embryonic stem cell-derived progenitor cells (Chu *et al.* 2016).

For these four data sets, two synthetic and two experimental, we test 16 methods: four traditional methods (WENDY, GENIE3, SINCERITIES, and NonlinearODEs), their transformer variants, ND variants, and BC variants. To compare the inferred GRN A_{pred} and the ground truth GRN A_{true}, we adopt the common practice that calculates two area under curve (AUC) scores: AUROC and AUPRC (Wang *et al.* 2024). Both AUC scores are between 0 and 1, where 1 means perfect matching, and 0 means perfect mismatching. See Section 2, available as supplementary data at *Bioinformatics* online for details of these two scores.

See Figs 2–6 and Tables 1 and 2, available as supplementary data at *Bioinformatics* online for the performance of these methods. Although the training data only has $n=10$ genes, and the THP-1 and hESC data sets have $n=20$ and $n=18$ genes, the four transformer-enhanced methods (TRENDY, tGENIE3, tSINCERITIES, tNonlinearODEs) are better than all other methods. This means that our idea of enhancing GRN inference methods with TE(k) models works well universally. ND and BC generally do not enhance the original method significantly.

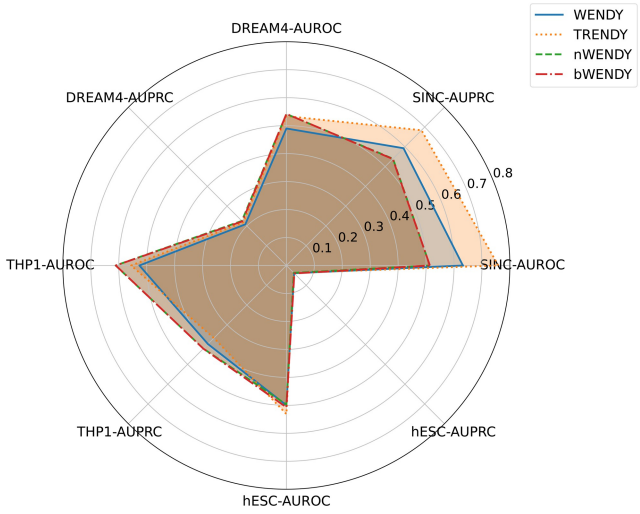

Figure 2. AUROC and AUPRC scores of WENDY and its variants on four datasets.

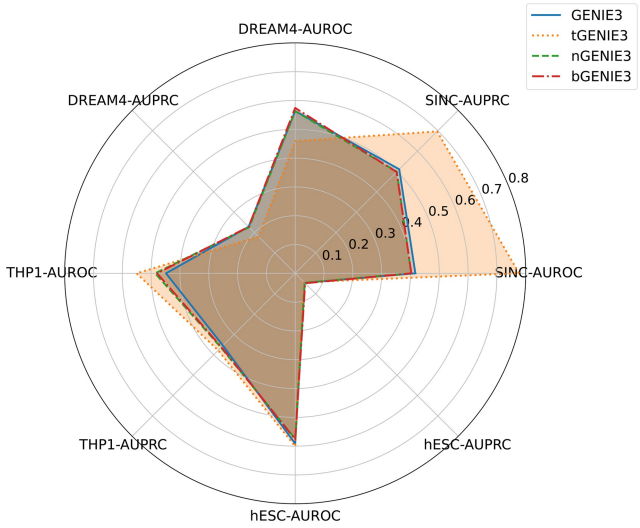

Figure 3. AUROC and AUPRC scores of GENIE3 and its variants on four datasets.

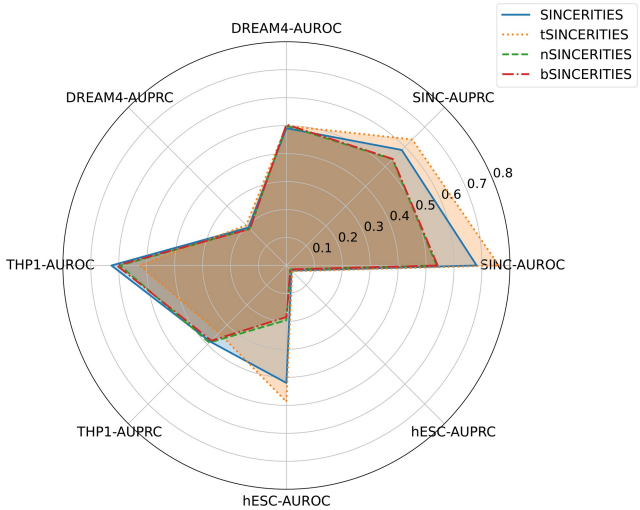

Figure 4. AUROC and AUPRC scores of SINCERITIES and its variants on four datasets.

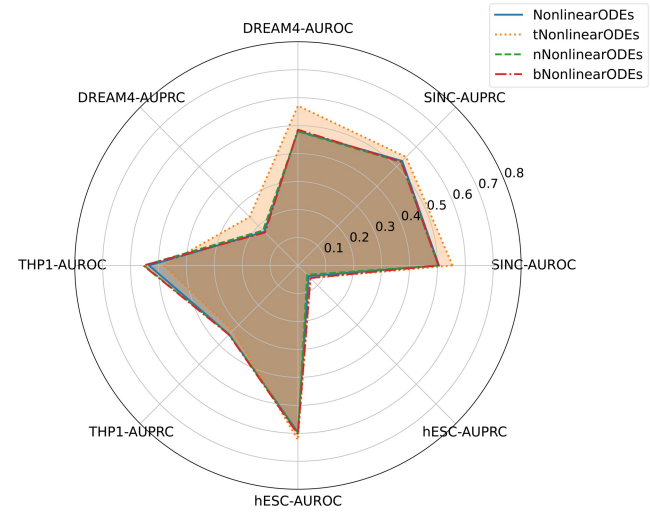

Figure 5. AUROC and AUPRC scores of NonlinearODEs and its variants on four datasets.

The core method of this article, TRENDY, ranks the first among all 16 methods (tGENIE3 is slightly inferior). Besides, the TRENDY method is better than the WENDY method on three data set, and has almost the same performance as WENDY on one data set (THP-1). The other three transformer-enhanced methods are worse than their non-transformer counterparts on at least one data set. This means that TRENDY's performance is robust on different data sets. In sum, TRENDY has satisfactory performance among all methods tested.

All methods have very small AUPRC values on hESC data set. The reason is that the ground truth GRN A_{true} has very few nonzero values (Fig. 3, available as supplementary data at *Bioinformatics* online).

3 Discussion

In this article, we present the TRENDY method, which uses the transformer model TE(k) to enhance the mechanism-based GRN inference method WENDY. TRENDY is tested against three other transformer-enhanced methods and their original forms, and ranks the first. This work explores the potential of deep learning methods in GRN inference when there are sufficient training data. Besides, TRENDY is developed on the biological dynamics of gene regulation, and it is more interpretable than other deep learning-based methods that treat GRN inference as a pure prediction task. Nevertheless, the transformer encoder is directly used as an oracle machine in TRENDY. A future direction is to modify the neural network structure to better represent the biological structure (Liu *et al.* 2020).

The essential difficulty of GRN inference is the lack of data with experimentally verified GRNs. Many deep learning models need significantly more data with known GRNs, and we have to generate new data. Since we do not fully understand the relation between the dynamics of gene expression and the ground truth GRN, it is difficult to evaluate the quality of generated data along with a random GRN. We also cannot guarantee that GRN inference methods that perform well on known experimental data sets can still be advantageous if there are many more experimental data with different GRNs.

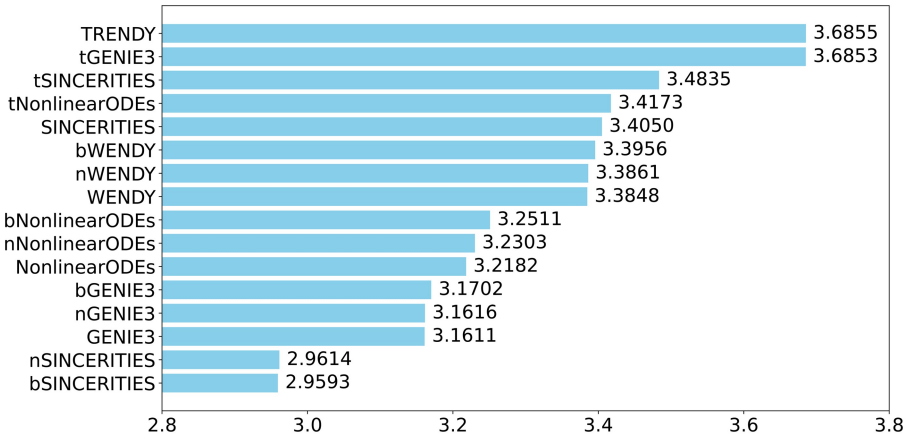

Figure 6. Total scores of all 16 methods, where the *x*-axis starts at 2.8.

We train the models with data of $n = 10$ genes, and they work well on data with $n = 10/18/20$ genes. If we want the models to work on data with many more genes, we need to train the models with corresponding data. However, when the gene number n is large, the number of different GRNs increases exponentially, and the number of training samples should also increase violently. The time cost for generating such data can be quite large. Even with the same amount of training samples, the training speed of the TE(k) model is proportional to n^4, since the input length of the transformer encoder layer is proportional to n^2, and it needs to calculate the attention scores between all pairs of input tokens. Therefore, scaling TRENDY for a much larger n is extremely time-consuming.

In this article, the training data are generated by Equation (1) that simulates the dynamics of mRNA count as a continuous-state process. This might be an oversimplification of gene regulation in reality, where the gene state and the protein count also should be considered (Holehouse *et al.* 2020). Besides, the experimental gene expression data often suffer from incomplete measurement, where many mRNAs are not recorded, and there are many missing values. Therefore, when we train on perfectly measured data and test on data with missing values, the performance is not guaranteed. We can manually add measurement errors and missing values to the training data to solve this problem.

In our testing, we find that adding covariance matrices in the transformer (TRENDY and tGENIE3) is beneficial, since the covariance matrices may contain extra information besides that contained in the inferred GRN.

TRENDY uses covariance matrices instead of raw gene expression data as input. One advantage is that calculating covariance matrices by graphical lasso regularizes the data. One disadvantage is that covariance only concerns second-order moment, while information contained in higher-order moments is abandoned.

4 Methods

4.1 Training data

To generate training data for our TRENDY method, we use Equation (1) to numerically simulate gene expression data with the Euler-Maruyama method (Kloeden *et al.* 1992) for $n = 10$ genes. Every time we generate a random ground truth GRN A_{true}, where each element has probability $0.1/0.8/0.1$ to be $-1/0/1$. Then we use Eq. 1 with this random A_{true} to run the simulation from time 0.0 to time 1.0 with time step 0.01 to obtain one trajectory, where the initial state at time 0 is random. This is repeated 100 times to obtain 100 trajectories, similar to the situation where we measure the single-cell expression levels of n genes for 100 cells (Qian and Cheng 2020). We only record expression levels at time points $0.0, 0.1, 0.2, \ldots, 1.0$. We repeat the simulation to obtain 1.01×10^5 samples, where 10^5 samples are for training, and 10^3 samples are for validation (used in hyperparameter tuning and early stopping).

4.2 Loss function

We use AUROC and AUPRC to measure the difference between A_{true} and the inferred A_{pred}. See Section 2, available as supplementary data at *Bioinformatics* online for details. Unfortunately, for fixed A_{true} and arbitrary A_{pred}, AUROC and AUPRC can only take finitely many values, and are not continuous functions of A_{pred}. We cannot train a neural network with a non-differentiable loss function like AUROC or AUPRC. There have been some surrogate loss functions for AUROC or AUPRC (Qi *et al.* 2021, Yuan *et al.* 2023). We directly use the mean square error loss function, which also performs well. The loss for comparing two GRNs does not count the diagonal elements, since they represent the autoregulation that cannot be inferred by the methods in this article (Wang and He 2023).

4.3 Segment embedding

For TE(k) model with $k > 1$ input matrices, after processing them with the linear embedding layer, we need to add segment embeddings to them, so that the model can differentiate between different input matrices (Devlin *et al.* 2019). We give the segment ID $0/1/2/\ldots$ to the first/second/third/... group of processed inputs, and use an embedding layer to map each segment ID to a d-dimensional embedding vector. Repeat this embedding vector $n \times n$ times to obtain an $(n \times n \times d)$ array, and add it to the $(n \times n \times d)$ arrays after the linear embedding layer. In our numerical simulations, we find that this segment embedding layer slightly improves the performance.

4.4 Positional encoding

For the TE(k) model, the 2-dimensional positional encoding layer (Xu *et al.* 2023) adds an $(n \times n \times d)$ array PE to each of the embedded input, so that the model knows that the input is two-dimensional. For x and y in $1, 2, \ldots, n$ and j in $1, 2, \ldots, d/4$, PE is defined as

$$\begin{aligned}
\mathrm{PE}[x,y,2j-1] &= \mathrm{PE}[x,y,2j-1+d/2] \\
&= \sin[(y-1)\times 10^{-32(j-1)/d}], \\
\mathrm{PE}[x,y,2j] &= \mathrm{PE}[x,y,2j+d/2] \\
&= \cos[(y-1)\times 10^{-32(j-1)/d}], \\
\mathrm{PE}[x,y,2j-1+d/4] &= \mathrm{PE}[x,y,2j-1+3d/4] \\
&= \sin[(x-1)\times 10^{-32(j-1)/d}], \\
\mathrm{PE}[x,y,2j+d/4] &= \mathrm{PE}[x,y,2j+3d/4] \\
&= \cos[(x-1)\times 10^{-32(j-1)/d}].
\end{aligned} \tag{5}$$

In our numerical simulations, we find that this 2-D positional encoding is crucial for the TE(k) model to produce better results.

4.5 Training setting and cost

All TE(k) models used in this article are trained for 100 epochs. For each epoch, we evaluate the model performance on the validation data set. If the performance does not improve for 10 consecutive epochs, we stop training early. For all transformer encoder layers, the dropout rate is 0.1. The optimizer is Adam with learning rate 0.001.

Data generation and model training are conducted on a desktop computer with Intel i7-13700 CPU and NVIDIA RTX 4080 GPU. Data generation takes about one week with CPU parallelization. Training for each TE(k) model takes about two hours with GPU acceleration. After training, the time cost of applying TRENDY on a data set with ~ 10 genes and ~ 100 cells is under one second.

Acknowledgements

The authors would like to thank Dr. Zikun Wang for helpful comments. Y.W. would like to thank Dr. Mingda Zhang, Dr. Lingxue Zhu, and Mr. Vincent Zhang for an intriguing discussion. Y.P. would like to thank Irving Institute for Cancer Dynamics of Columbia University for hosting the Summer Research Program.

Author contributions

Xueying Tian (Methodology [Equal], Software [Equal]), Yash Patel (Methodology [Equal], Software [Equal]), and Yue Wang (Conceptualization [Lead], Methodology [Equal], Software [Equal], Writing—original draft [Lead], Writing—review & editing [Lead])

Supplementary data

Supplementary data are available at *Bioinformatics* online.

Conflict of interest: None declared.

Funding

None declared.

References

Angelini E, Wang Y, Zhou JX *et al.* A model for the intrinsic limit of cancer therapy: duality of treatment-induced cell death and treatment-induced stemness. *PLoS Comput Biol* 2022; **18**:e1010319.

Arshad OA, Venkatasubramani PS, Datta A *et al.* Using Boolean logic modeling of gene regulatory networks to exploit the links between cancer and metabolism for therapeutic purposes. *IEEE J Biomed Health Inform* 2016;**20**:399–407.

Axelrod S, Li X, Sun Y *et al.* The drosophila blood–brain barrier regulates sleep via moody g protein-coupled receptor signaling. *Proc Natl Acad Sci USA* 2023;**120**:e2309331120.

Bothma JP, Garcia HG, Esposito E *et al.* Dynamic regulation of eve stripe 2 expression reveals transcriptional bursts in living drosophila embryos. *Proc Natl Acad Sci USA* 2014;**111**:10598–603.

Burdziak C, Zhao CJ, Haviv D *et al.* Sckinetics: inference of regulatory velocity with single-cell transcriptomics data. *Bioinformatics* 2023; **39**:i394–403.

Cao M, Zhang H, Park J *et al.* Going the distance for protein function prediction: a new distance metric for protein interaction networks. *PLoS One* 2013;**8**:e76339.

Chan TE, Stumpf MPH, Babtie AC. Gene regulatory network inference from single-cell data using multivariate information measures. *Cell Syst* 2017;**5**:251–67.e3.

Cheng RY-H, Hung KL, Zhang T *et al.* Ex vivo engineered human plasma cells exhibit robust protein secretion and long-term engraftment in vivo. *Nat Commun* 2022;**13**:6110.

Cheng Y-C, Stein S, Nardone A *et al.* Mathematical modeling identifies optimum palbociclib-fulvestrant dose administration schedules for the treatment of patients with estrogen receptor–positive breast cancer. *Cancer Res Commun* 2023;**3**:2331–44.

Cheng Y-C, Zhang Y, Tripathi S *et al.* Reconstruction of single-cell lineage trajectories and identification of diversity in fates during the epithelial-to-mesenchymal transition. *Proc Natl Acad Sci USA* 2024;**121**:e2406842121.

Chu L-F, Leng N, Zhang J *et al.* Single-cell RNA-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm. *Genome Biol* 2016;**17**:173.

Devlin J, Chang M-W, Lee K *et al.* Bert: pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Held in Minneapolis, MN, USA. Stroudsburg, PA, USA: Association for Computational Linguistics (ACL), 2019, 4171–86.

Dibaeinia P, Sinha S. SERGIO: a single-cell expression simulator guided by gene regulatory networks. *Cell Syst* 2020;**11**:252–71.e11.

Feizi S, Marbach D, Médard M *et al.* Network deconvolution as a general method to distinguish direct dependencies in networks. *Nat Biotechnol* 2013;**31**:726–33.

Feng K, Jiang H, Yin C *et al.* Gene regulatory network inference based on causal discovery integrating with graph neural network. *Quant Biol* 2023;**11**:434–50.

Friedman J, Hastie T, Tibshirani R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* 2008;**9**:432–41.

Giovanini G, Sabino AU, Barros LRC *et al.* A comparative analysis of noise properties of stochastic binary models for a self-repressing and for an externally regulating gene. *Math Biosci Eng* 2020;**17**:5477–503.

Hache H, Wierling C, Lehrach H *et al.* Genge: systematic generation of gene regulatory networks. *Bioinformatics* 2009;**25**:1205–7.

Holehouse J, Cao Z, Grima R. Stochastic modeling of autoregulatory genetic feedback loops: a review and comparative study. *Biophys J* 2020;**118**:1517–25.

Huynh-Thu VA, Geurts P. dyngenie3: dynamical genie3 for the inference of gene networks from time series expression data. *Sci Rep* 2018;**8**:3384.

Huynh-Thu VA, Irrthum A, Wehenkel L *et al.* Inferring regulatory networks from expression data using tree-based methods. *PLoS One* 2010;**5**:e12776.

Kamimoto K, Stringa B, Hoffmann CM *et al.* Dissecting cell identity via network inference and in silico gene perturbation. *Nature* 2023;**614**:742–51.

Kentzoglanakis K, Poole M. A swarm intelligence framework for reconstructing gene networks: searching for biologically plausible architectures. *IEEE/ACM Trans Comput Biol Bioinform* 2012; **9**:358–71.

Kloeden PE, Platen E. Numerical Solution of *Stochastic Differential Equations*. Berlin: Springer, 1992.
Kouno T, de Hoon M, Mar JC *et al.* Temporal dynamics and transcriptional control using single-cell gene expression analysis. *Genome Biol* 2013;**14**:R118–12.
Leyes Porello EA, Trudeau RT, Lim B. Transcriptional bursting: stochasticity in deterministic development. *Development* 2023; **150**:dev201546.
Li W, Wang Z, Syed S *et al.* Chronic social isolation signals starvation and reduces sleep in drosophila. *Nature* 2021;**597**:239–44.
Liu Y, Barr K, Reinitz J. Fully interpretable deep learning model of transcriptional control. *Bioinformatics* 2020;**36**:i499–507.
Ma B, Fang M, Jiao X. Inference of gene regulatory networks based on nonlinear ordinary differential equations. *Bioinformatics* 2020; **36**:4885–93.
Mao G, Pang Z, Zuo K *et al.* Predicting gene regulatory links from single-cell rna-seq data using graph neural networks. *Brief Bioinform* 2023;**24**:bbad414.
Marbach D, Costello JC, Küffner R, *et al.* Wisdom of crowds for robust gene network inference. *Nature Methods* 2012;**9**:796–804.
Marouf M, Machart P, Bansal V *et al.* Realistic in silico generation and augmentation of single-cell RNA-seq data using generative adversarial networks. *Nat Commun* 2020;**11**:166.
McDonald TO, Cheng Y-C, Graser C *et al.* Computational approaches to modelling and optimizing cancer treatment. *Nat Rev Bioeng* 2023;**1**:695–711.
Myasnikova E, Spirov A. Gene regulatory networks in drosophila early embryonic development as a model for the study of the temporal identity of neuroblasts. *Biosystems* 2020;**197**:104192.
Nauta M, Bucur D, Seifert C. Causal discovery with attention-based convolutional neural networks. *MAKE* 2019;**1**:312–40.
Papili Gao N, Minhaz Ud-Dean SM, Gandrillon O *et al.* SINCERITIES: inferring gene regulatory networks from time-stamped single cell transcriptional expression profiles. *Bioinformatics* 2018; **34**:258–66.
Paré A, Lemons D, Kosman D *et al.* Visualization of individual SCR MRNAS during drosophila embryogenesis yields evidence for transcriptional bursting. *Curr Biol* 2009;**19**:2037–42.
Pinna A, Soranzo N, De La Fuente A. From knockouts to networks: establishing direct cause-effect relationships through graph analysis. *PLoS One* 2010;**5**:e12912.
Pirayre A, Couprie C, Bidard F *et al.* Brane cut: biologically-related a priori network enhancement with graph cuts for gene regulatory network inference. *BMC Bioinformatics* 2015;**16**:368–12.
Pirayre A, Couprie C, Duval L *et al.* Brane clust: cluster-assisted gene regulatory network inference refinement. *IEEE/ACM Trans Comput Biol Bioinform* 2018;**15**:850–60.
Qi Q, Luo Y, Xu Z *et al.* Stochastic optimization of areas under precision-recall curves with provable convergence. In: *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*. Held in virtual conference (originally planned for Vancouver, Canada). Red Hook, NY, USA: Curran Associates, Inc., 2021, 1752–65.
Qian H, Cheng Y-C. Counting single cells and computing their heterogeneity: from phenotypic frequencies to mean value of a quantitative biomarker. *Quant Biol* 2020;**8**:172–6.
Ramos AF, Reinitz J. Physical implications of so (2, 1) symmetry in exact solutions for a self-repressing gene. *J Chem Phys* 2019; **151**:041101.
Schaffter T, Marbach D, Floreano D. GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics* 2011;**27**:2263–70.
Sha Y, Qiu Y, Zhou P *et al.* Reconstructing growth and dynamic trajectories from single-cell transcriptomics data. *Nat Mach Intell* 2024; **6**:25–39.
Shu H, Ding F, Zhou J *et al.* Boosting single-cell gene regulatory network reconstruction via bulk-cell transcriptomic data. *Brief Bioinform* 2022;**23**:bbac389.
Shu H, Zhou J, Lian Q *et al.* Modeling gene regulatory networks using neural network architectures. *Nat Comput Sci* 2021;**1**:491–501.
Tripathi S, Lloyd-Price J, Ribeiro A *et al.* sgnesr: an r package for simulating gene expression data from an underlying real gene network structure considering delay parameters. *BMC Bioinformatics* 2017; **18**:325.
Vaswani A, Shazeer N, Parmar N *et al.* Attention is all you need. In: *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*. Held in Long Beach, CA, USA. Red Hook, NY, USA: Curran Associates, Inc., 2017, 5998–6008.
Vijayan V, Wang Z, Chandra V *et al.* An internal expectation guides drosophila egg-laying decisions. *Sci Adv* 2022;**8**:eabn3852.
Wang Z. *identification of gene expression changes in sleep mutants associated* with reduced longevity in Drosophila. PhD thesis, The Rockefeller University, 2020, 59–74.
Wang Y, He S. Inference on autoregulation in gene expression with variance-to-mean ratio. *J Math Biol* 2023;**86**:87.
Wang Y, Kropp J, Morozova N. Biological notion of positional information/value in morphogenesis theory. *Int J Dev Biol* 2020; **64**:453–63.
Wang Z, Lincoln S, Nguyen AD *et al.* Chronic sleep loss disrupts rhythmic gene expression in drosophila. *Front Physiol* 2022;**13**:1048751.
Wang B, Pourshafeie A, Zitnik M *et al.* Network enhancement as a general method to denoise weighted biological networks. *Nat Commun* 2018;**9**:3108.
Wang L, Trasanidis N, Wu T *et al.* Dictys: dynamic gene regulatory network dissects developmental continuum with single-cell multiomics. *Nat Methods* 2023;**20**:1368–78.
Wang Y, Wang Z. Inference on the structure of gene regulatory networks. *J Theor Biol* 2022;**539**:111055.
Wang Y, Zheng P, Cheng Y-C *et al.* Wendy: covariance dynamics based gene regulatory network inference. *Math Biosci* 2024;**377**:109284.
Xu J, Zhang A, Liu F *et al.* Stgrns: an interpretable transformer-based method for inferring gene regulatory networks from single-cell transcriptomic data. *Bioinformatics* 2023;**39**:btad165.
Yuan Z, Zhu D, Qiu Z-H *et al.* Libauc: a deep learning library for x-risk optimization. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Held in Long Beach, CA, USA. New York, NY, USA: Association for Computing Machinery (ACM), 2023, 5487–99.
Zheng R, Li M, Xiang Chen F-X *et al.* Bixgboost: a scalable, flexible boosting-based method for reconstructing gene regulatory networks. *Bioinformatics* 2019;**35**:1893–900.

Bioinformatics, 2025, 41, 1–8
https://doi.org/10.1093/bioinformatics/btaf314
Original Paper