





METHOD ARTICLE

DeepReGraph co-clusters temporal gene expression and cis-regulatory elements through heterogeneous graph representation learning [version 1; peer review: awaiting peer review]

Jesús Fernando Cevallos Moreno ^{1*}, Peyman Zarrineh^{2*},
Aminael Sánchez-Rodríguez ³, Massimo Mecella¹

¹Dipartimento di Ingegneria Informatica Automatica e Gestionale, Sapienza University of Rome, Roma, RM, 00185, Italy

²School of Medical Sciences, University of Manchester, Manchester, M13 9PT, UK

³Department of Biological Sciences, Universidad Técnica Particular de Loja, Loja, 1101608, Ecuador

* Equal contributors

V1 First published: 13 May 2022, 11:518
<https://doi.org/10.12688/f1000research.114698.1>

Latest published: 13 May 2022, 11:518
<https://doi.org/10.12688/f1000research.114698.1>

Abstract

This work presents DeepReGraph, a novel method for co-clustering genes and cis-regulatory elements (CREs) into candidate regulatory networks. Gene expression data, as well as data from three CRE activity markers from a publicly available dataset of mouse fetal heart tissue, were used for DeepReGraph concept proofing. In this study we used open chromatin accessibility from ATAC-seq experiments, as well as H3K27ac and H3K27me3 histone marks as CREs activity markers. However, this method can be executed with other sets of markers. We modelled all data sources as a heterogeneous graph and adapted a state-of-the-art representation learning algorithm to produce a low-dimensional and easy-to-cluster embedding of genes and CREs. Deep graph auto-encoders and an adaptive-sparsity generative model are the algorithmic core of DeepReGraph. The main contribution of our work is the design of proper combination rules for the heterogeneous gene expression and CRE activity data and the computational encoding of well-known gene expression regulatory mechanisms into a suitable objective function for graph embedding. We showed that the co-clusters of genes and CREs in the final embedding shed light on developmental regulatory mechanisms in mouse fetal-heart tissue. Such clustering could not be achieved by using only gene expression data. Function enrichment analysis proves that the genes in the co-clusters are involved in distinct biological processes. The enriched transcription factor binding sites in CREs prioritize the candidate transcript factors which drive the temporal changes in gene

Open Peer Review

Approval Status *AWAITING PEER REVIEW*

Any reports and responses or comments on the article can be found at the end of the article.

expression. Consequently, we conclude that DeepReGraph could foster hypothesis-driven tissue development research from high-throughput expression and epigenomic data. Full source code and data are available on the DeepReGraph [GitHub](#) project.

Keywords

Developmental Gene Regulatory Networks, Cis-Regulatory Elements, Heterogeneous Graph Representation Learning, Deep Graph Auto-encoders



This article is included in the [Bioinformatics gateway](#).

Corresponding author: Aminael Sánchez-Rodríguez (asanchez2@utpl.edu.ec)

Author roles: **Cevallos Moreno JF:** Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Resources, Software, Visualization, Writing – Original Draft Preparation; **Zarrineh P:** Conceptualization, Data Curation, Formal Analysis, Investigation, Resources, Supervision, Validation, Visualization, Writing – Original Draft Preparation; **Sánchez-Rodríguez A:** Conceptualization, Methodology, Project Administration, Supervision, Visualization, Writing – Review & Editing; **Mecella M:** Funding Acquisition, Project Administration, Supervision, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: ASR would like to thank the Universidad Técnica Particular de Loja for its financial support under project PROY\INV\CCBIO\2021\3150. JC; MM would like to acknowledge the financial support from ELIS Innovation Hub. PZ was supported by Biotechnology and Biological Sciences Research Council (BBSRC) grant BB/T007761/1
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2022 Cevallos Moreno JF *et al.* This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Cevallos Moreno JF, Zarrineh P, Sánchez-Rodríguez A and Mecella M. **DeepReGraph co-clusters temporal gene expression and cis-regulatory elements through heterogeneous graph representation learning [version 1; peer review: awaiting peer review]** F1000Research 2022, 11:518 <https://doi.org/10.12688/f1000research.114698.1>

First published: 13 May 2022, 11:518 <https://doi.org/10.12688/f1000research.114698.1>

Introduction

Gene regulatory networks drive gene expression during cell development. Studying the regulatory networks' effects on gene expression has become an essential topic in evolutionary developmental biology. Recently, a wide range of temporal high-throughput experiments has become available for both normal development and disease-related investigations.¹⁻³ For example, the ENCODE consortium has provided comprehensive fetal development datasets for 12 tissues in mice from the E10.5 stage to the birth time P0.¹ These datasets include both gene expression data and major transcriptional and epigenetic features. Some of the latter come from ATAC-seq experiments that measure chromatin accessibility, whole-genome shotgun bisulfite sequencing experiments measuring DNA methylation, and histone modifications.¹ Similar datasets, though in lesser amount, exist to study aging (see a recent review by Pagiatakis *et al.*²). Cancer, instead, is among the diseases with the largest amount of available gene regulatory datasets (see a recent review by Zboril *et al.*³).

Gene expression regulation occurs via non-coding segments, also known as cis-regulatory elements (CRE).⁴ Associating CREs to the genes they regulate is the key to deciphering temporal dynamic changes through development. As CREs and genes are distinct entities located on different genome coordinates, a concurrent study of these has been challenging. Many methods have been developed to study genes and CREs separately through development in the last years. Infinite Gaussian Process Mixture Model⁵ and Convolutional Neural Networks⁶ are among the most successful methods to find temporal clusters of either genes or CREs. Generally, high-throughput datasets are highly clusterable, and even conventional clustering methods like K-Means clustering can also generate satisfactory results when used for a single source dataset (*i.e.* genes or CREs). However, combining or aligning the clusters across data sources is still a major challenge.

Insights about developmental regulatory networks are typically gained by the application of Multiview learning⁷ to align single-cell RNA sequencing (scRNA-seq) and single-cell ATAC sequencing (scATAC-seq).^{8,9} Such methods permit the classification of different cells through the combination of various high-throughput experiments made over the same cell population.^{10,11} The Seurat method,¹² for example, makes different reduced dimension representations of single-cell gene expression and CREs data. Such a method uses Canonical Correlation Analysis to place multiple datasets on a common latent space with reduced dimension and anchor a population of cells across different modality datasets. Other methods rely on stronger prior knowledge about cell alignment to anchor multiple modalities. This is the case of the Multi-Omics Factor Analysis v2 (MOFA+), a statistical framework for the comprehensive and scalable integration of single-cell multi-modal data.¹³ More recently, multi-modal dataset alignment methods have been developed based on state-of-the-art deep learning techniques.^{14,15} MAGAN,¹⁶ for example, successfully aligned scRNA-seq and scATAC-seq datasets using Generative Adversarial Networks (GAN). Yang *et al.*,¹⁷ used deep auto-encoders to map multiple data modalities of a population of cells to a common latent space. They used prior knowledge and adversarial learning to drive the correct alignment of data. Comprehensive surveys on multi-modal or multi-view learning applied to multi-omics datasets can be found in.^{8,18}

The works mentioned above explore gene expression regulatory mechanisms by combining multiple views of the same cell population. This work investigates gene regulatory networks by clustering two diverse entity types together: We sought to group CREs with their correspondent regulated genes. In this sense, we performed a heterogeneous network clustering task. Heterogeneous data interaction was also modelled by Huang *et al.*¹⁹ for predicting transcription factor interactions with their target genes. Wang²⁰ and Zhou²¹ instead, modelled gene-disease interactions through a heterogeneous-network model. Other recent problems solved through heterogeneous graph-based models include Gene Ontology Representation Learning,²² Gene prioritization for rare diseases²³ and drug repurposing.²⁴ The main novelty of this work is the introduction of a model for regulatory networks discovery which is based on the concept of "Heterogeneous Graphs".²⁵ We modelled a graph containing two classes of nodes: genes and their potential regulatory elements, now on termed as candidate-CREs (cCRE), and multiple types of relations or edges between nodes. One such relation type is the similarity between temporal gene expression profiles. Another relation type is the similarity between temporal cCREs activity patterns. Finally, we introduced a third edge type: the base-pair distance between genes and cCREs in the genome. Having defined such relations, we propose the usage of graph representation learning (graph RL) to group genes and cCREs as a function of the probability of the existence of gene regulation mechanisms between them.

To this end, we extended a state-of-the-art graph RL algorithm presented in Ref. 26 to make it capable of learning representations of heterogeneous graphs and applied it to the ENCODE heart dataset in Ref. 1. We used open chromatin regions from ATAC-seq as cCREs. Besides ATAC-seq, we used H3K27ac and H3K27me3 histone marks, well-known markers of enhancer activity, and polycomb repression. Our framework produced a relation type between genes and cCREs which is aware of both same-class pattern similarities and base-pair proximities between elements of different classes. Moreover, through the usage of the adaptive neighbors model presented in Ref. 26 we produced a "clustering-friendly" embedding where we could straightforwardly perform clustering. The resulting clusters contained both genes and cCREs and tended to identify possible regulatory mechanisms. We demonstrated this claim by evaluating such

clusters' semantic power with external expert criteria. We concluded that the proposed methodology is able to learn suitable combinations of multiple entity relations to form a unique relation under a graph that resembles a gene regulatory network. Our framework is called DeepReGraph, because it is a *Deep* learning-based algorithm for identification of gene expression *Regulatory* mechanisms through heterogeneous *Graph* RL. Our full code, the pre-processed datasets used, and the values used for all the parameters and hyperparameters used in our experiment are online available at our repository^[1].

Methods

To demonstrate DeepReGraph's ability to cluster genes and cCREs simultaneously, we applied this method to a mouse heart fetal developmental dataset from the ENCODE project.¹ Gene expression and cCRE activities were time-series formatted. We used three well-characterized epigenetic markers: ATAC-seq, H3K27ac, and H3K27me3 measurements as features of cCRE temporal activities. We aimed to create a low-dimensional and clustered representation of the whole dataset, where clusters represent candidate gene-expression regulatory mechanisms (GERM). In other words, we aimed to create clusters containing both genes and cCREs, with the working hypothesis that cCREs on a cluster are plausibly regulating the expression of the genes in the same cluster. Moreover, genes on the same GERM cluster should have similar gene expression profiles, and cCREs should have similar activity profiles. Lastly, the base-pair distances between genes and cCREs on the same cluster should be relatively small with respect to distances between elements in different clusters. We validated heterogeneous clusters of genes and cCREs to study how gene regulatory networks (GRN) drive changes in gene expression during mouse heart fetal development. To validate gene expression clusters, we performed enrichment analysis of biological process gene ontology terms using the ClusterProfiler bioconductor package.²⁷ To validate the cCRE clusters, we looked for the Enriched Homer Motifs.²⁸

Data pre-processing

Data pre-processing was done in two stages. Firstly, gene expression and cCRE datasets were normalized across temporal data. The second stage included filtering-out temporal profiles where the correlation across replicates or the variance across time was below some pre-defined thresholds. We explain the pre-processing stages in greater detail below.

We firstly downloaded the gene expression values, called peaks, and bam files of ATAC-seq, H3K27ac, and H3K27me3 from the ENCODE database¹ (mice heart fetal tissue). We used logFPKM values of gene expression throughout this paper. To detect cCRE regions, we re-centered ATAC-seq called peaks across time points and replicates using the bioconductor DiffBind package.²⁹ To determine chromatin accessibility in each cCRE region, we took regions of 500 nucleotides long (250 nucleotides on each side from the center of cCREs). We recorded binding intensities for each replicate of each time point separately by using the bioconductor Rsubread package.³⁰ We counted H3K27ac and H3K27me3 marks for cCREs in the same manner, but used regions 3000 nucleotides long instead (1500 nucleotides each side from the center of cCREs). We took the median value of non-peak regions in each experiment separately to remove background counts from real intensity counts. Then, we subtracted these median values from the intensities of each experiment separately. We used logRPKM values of denoised counts across replicates and time points. Removing the background noise caused the distribution of each experiment to resemble a Gaussian distribution.

At the end of this process, we had two replicas of gene expression time-series profiles for a set of genes and two replicas of cCRE activity time-series for each activity marker over a set of cCREs. We then proceeded to filter out invalid genes and cCREs based on two criteria. The first group of filters we applied to this data were proposed in Ref. 31. For each gene, we measured the correlation of the gene expression profiles of both replicates and discarded every gene whose replicate had a negative correlation value. We did the same with the cCREs for each activity marker profile. If only one activity marker had a negative correlation between replicates, the cCRE was discarded. We also discarded every element (gene or cCRE) where two consecutive time probes had an absolute ratio greater than 2.

We also discarded low-expressed genes and low activity-characterized cCREs.³² We computed the mean gene expression value for each gene and discarded every gene whose mean expression value was under the 0.8 percentile of such mean value distribution. We did the same with cCREs: we computed the mean activity value for each activity marker and discarded every cCRE where the mean value was below a minimum percentile threshold. Such thresholds were 0.8 for ATAC-seq and 0.7 for H3K27ac and H3K27me3. Lastly, for gene expression and each activity marker dataset, we created a 3-rd degree polynomial regression to estimate the variance of each time series as a function of its mean value. We discarded all the elements where the actual variance was below the predicted variance. Our pre-processed and cleaned data consisted of 607 genes and 5239 cCREs from the mouse heart dataset. This dataset is available online on our public repository. Our objective was to shed light on regulatory mechanisms among genes and cCREs. Having cleaned our

¹<https://github.com/QwertyJacob/DeepReGraph>

dataset, modeling it as a heterogeneous graph is the strategy we used to combine all the data and learn a new relation type between elements that gives information about gene expression regulation. We explain the graph modelling in the next paragraph.

Heterogeneous graph model

Many industrial data are represented in a graph, as multiple entities with quantifiable relationships between them. Nowadays, graph modelisation is widely used in bioinformatics.³³ One strategy to extract added-value from graphs is the use of graph RL algorithms. Graph RL has been widely studied by the research community over the past few years.^{25,34–36} Graph RL algorithms seek to map the information of a graph on a reduced-dimension latent space where the geometrical distances between elements in such space resemble the relations in the original graph. In other words, graph RL, also referred to as “graph embedding”, consists in finding a reduced dimensional representation of the nodes of a graph while conserving most of the semantic information of such a graph.^{37,38}

A graph can be homogeneous if it has a unique type of node and a unique type of relation defined between them, or heterogeneous, if it introduces multiple edge and node types. We argue that regulatory networks during embryonic development can also benefit from being modelled as a heterogeneous graph. Reduced-dimension representations of such a graph could contain multiple types of information that, if well combined, could semantically express regulatory mechanisms of gene expression.

We defined a heterogeneous graph and described it as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$, where \mathcal{V} is the set of nodes, \mathcal{E} is the set of edges and $\mathcal{T} = \mathcal{T}_{\mathcal{E}} \cup \mathcal{T}_{\mathcal{V}}$ is the set containing all the node types and edge types. The set of node types consists of genes and cCREs, and is denoted by $\mathcal{T}_{\mathcal{V}} = \{genes, ccre\}$. Each gene $g_i, \forall i \in |\mathcal{V}_{genes}|$ in the dataset is characterized by a gene expression time-series profile that will be encoded in a vector denoted by \mathbf{g}_i . Note that the length of \mathbf{g}_i coincides with the number of time-points considered in the gene expression data. The vectors $\mathbf{g}_i, \forall i \in |\mathcal{V}_{genes}|$ can be concatenated as rows to form the feature matrix represented in red in panel A of Figure 1.

Every candidate cis-regulatory element, instead, will be denoted as $c_j, \forall j \in |\mathcal{V}_{ccres}|$ and will be characterized by M activity time-series profiles, that will be encoded in corresponding vectors denoted as $\mathbf{c}_j^1, \mathbf{c}_j^2, \dots, \mathbf{c}_j^M$. The time-points under which gene expression and cCRE activity values are available are the same. In this work, $M = 3$. Specifically, we considered the ATAC-seq, H3K27ac, and H3K27me3 activity time-series profiles for each cCRE, but one could include more cCRE

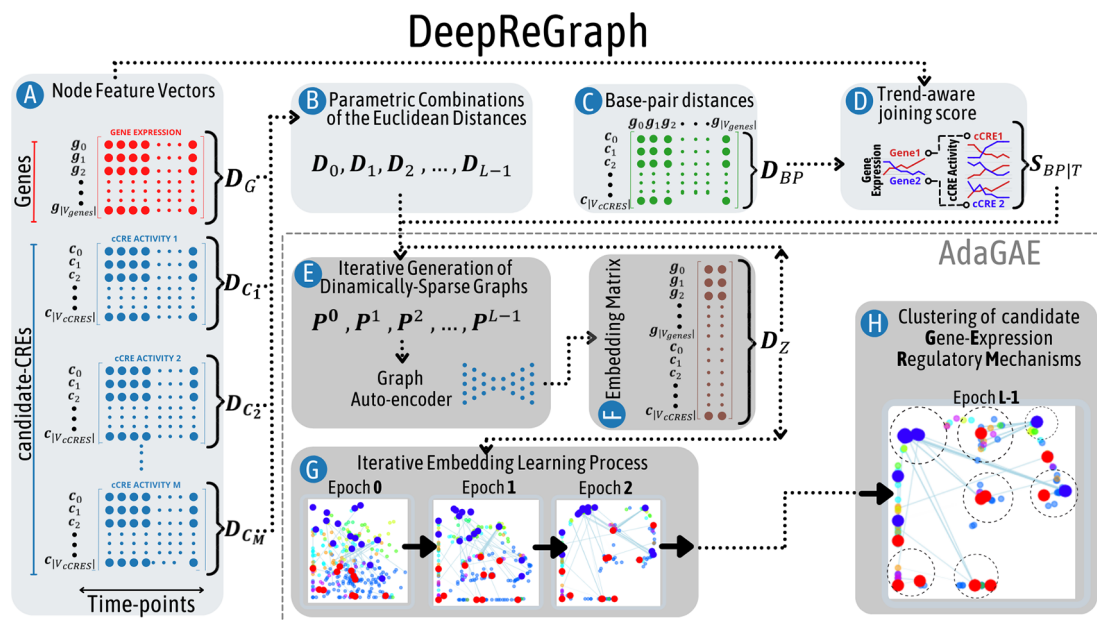


Figure 1. Schematic representation of DeepReGraph. Our framework combines multiple temporal enhancer activity markers with temporal gene expression data and base-pair distances to build a heterogeneous graph of genes and cis-regulatory elements (CRE). The proposed framework then applies an adapted version of AdaGAE²⁶ to find a low-dimensional, easy-to-cluster embedding representation of data through an iterative optimisation process. In the final embedding produced by our method, the spatial distribution of genes and cCREs resembles candidate gene-expression regulatory Mechanisms.

activity markers in the set of features. Note that the vectors $\mathbf{c}_j^m, \forall j \in |\mathcal{V}_{cCREs}|$ can be concatenated as rows to form the matrices represented in blue in panel A of [Figure 1](#).

Each node of our graph was associated to one or multiple feature vectors that can be seen as points on a multi-dimensional feature space. The set of feature-spaces defined for the nodes in \mathcal{G} is denoted as $R = \{G, C_1, C_2, \dots, C_M\}$ where G is the gene expression feature space and C_1, C_2, \dots, C_M are the feature spaces of the M cCRE activity data. Each one of these feature spaces will be referred to as *original feature space*, and the vectors $\mathbf{g}_i, \mathbf{c}_j^1, \mathbf{c}_j^2, \dots, \mathbf{c}_j^M$ as original feature vectors from now on. Note that such vectors correspond to the rows of the matrices in panel A of [Figure 1](#). It is well-known that the probability of the existence of a regulatory mechanism between a gene and a cis-regulatory element is inversely proportional to the base-pair distance between them.³⁹ For this reason, in our work, we used another data source, called the Link Matrix, that gives us information about the base-pair distance between genes and cCREs. We represented such a matrix in panel C of [Figure 1](#). Interaction between cCREs and genes is generally possible in the range of 1 Mega base-pair.⁴⁰ Consequently, we considered only distances lower than 10^6 base-pairs. For practical purposes, we scaled the base-pair distance values in the interval $[0, 1]$ to define a scaled base-pair distance function, D^{BP} :

$$D^{BP} : (g_i, c_j) \rightarrow d_{ij}^{BP} \in [0, 1], \\ \forall g_i \in \mathcal{V}_{genes}, c_j \in \mathcal{V}_{cCREs}$$

Given the scaled distance function, we created a base-pair proximity relationship S_{BP} using a parametric transformation:

$$S_{BP}(i, j) = \frac{1}{\left(d_{ij}^{BP} / \beta_c\right)^{\beta_d} + 1}, \quad \forall i, j \in |\mathcal{V}| \quad (1)$$

where β_c and β_d are fixed hyper-parameters. Notice that the domain and co-domain of S^{BP} is defined in the interval $[0, 1]$. The gene-to-cCREs relationships defined by S_{BP} are sparse in the sense that, for each gene, only a few cCREs will have a non-zero proximity value for it.

The interaction between CREs and the genes they regulate is complex. Some CRE markers are correlated with gene expression and some others are anti-correlated.⁴¹ This correlation or anti-correlation is governed by the mechanism that a specific marker affects the gene expression. For example, chromatin should be opened prior to the gene expression. Therefore, chromatin accessibility is commonly directly correlated with gene expression. H3K27ac is also directly correlated with gene expression, as this epigenetic modification happens in active transcription regions. On the other hand, H3K27me3 shows epigenetic modifications that happen at polycomb repressed regions, and are consequently inversely correlated with the expression of the corresponding regulated genes. We needed to capture the relevance of these and other temporal covariances between gene expression and the correspondent regulatory elements' activity markers.

To this end, we created a joining score, J_T , as a function of the temporal slopes of gene expression and cCRE activity time-series:

$$J_T(g_i, c_j) = \left| \gamma_i^g + \frac{\sum_{m \in M} \omega_m \gamma_j^m}{\sum_{m \in M} \omega_m} \right|, \quad \text{where :} \\ \gamma_i^g = \text{sgn} \left(\frac{\partial \mathbf{g}_i}{\partial t} \right), \quad \gamma_j^m = \text{sgn} \left(\frac{\partial \mathbf{c}_j^m}{\partial t} \right) \\ \forall g_i \in \mathcal{V}_{genes}, c_j \in \mathcal{V}_{cCREs} \quad (2)$$

where the temporal slope of gene expression vector \mathbf{g}_i is denoted by $\frac{\partial \mathbf{g}_i}{\partial t}$, the slope of cCRE activity vectors is $\frac{\partial \mathbf{c}_j^m}{\partial t}, \forall m \in \{0, 1, \dots, M-1\}$, $\text{sgn}(\cdot)$ is the signum function, and $\omega_m \in [-1, 1]$ is a fixed parametric weight for the \mathbf{c}^m trend slope.

We computed a trend-aware score multiplying (1) and (2) for each gene-cCRE pair in our graph:

$$S_{BP|T}(g_i, c_j) = S_{BP}(i, j) \cdot J_T(g_i, c_j), \\ \forall g_i \in \mathcal{V}_{genes}, c_j \in \mathcal{V}_{cCREs} \quad (3)$$

We represented the computation of [Equation \(3\)](#) in panel D of [Figure 1](#). Having defined the trend-aware score, we could differentiate between any pair of genes, g_j and g_k , when these were equally proximal to any cCRE c_i but had gene

expression vectors with different temporal trends. In this case, the original base-pair proximity score in (1) assigned the same value to the association (g_j, c_i) and (g_k, c_i) . Notice that the trend-aware score in (2), instead, distinguishes the most plausible association of c_i with respect to the alternatives g_j and g_k as a function of the temporal slopes of the gene expressions elements. In our experiment, we found the best results setting $\omega_{ATAC} = 1$, $\omega_{H3K27ac} = 0$ and $\omega_{H3K27me3} = 0$ in Equation (2).

The set of edge types in the graph is denoted as $\mathcal{T}_{\mathcal{G}} = \{S_{BP|T}, S_G, S_{C_1}, S_{C_2}, \dots, S_{C_M}\}$. As explained before, $S_{BP|T}$ indicates the trend-aware base-pair proximity relationship explained above; S_G stands for the gene expression profile similarity between genes, and $S_{C_1}, S_{C_2}, \dots, S_{C_M}$ are the relationships that express the different cCRE activity time-series similarities. We aimed to combine all edge types in $\mathcal{T}_{\mathcal{G}}$ to find a unique multi-semantic edge type that resembled gene expression regulation mechanisms. The process of combining edge types is represented in panel E of Figure 1 and is explained later in this section. To perform the combination of all edge types in $\mathcal{T}_{\mathcal{G}}$, we modelled the data as a heterogeneous graph and designed a graph RL algorithm capable of extracting such a dependency between elements using prior biological knowledge and the data available. We explain the graph RL algorithm we used to fulfil our objective in the next paragraph.

Graph representation learning

Graph RL algorithms aim at finding a matrix: $\mathbf{Z} \in \mathbb{R}^{|\mathcal{V}| \times d}$ that contains information about every feature-space in R . In our case, we used a graph RL algorithm to find a unique feature space for the nodes of \mathcal{G} that comprises most of the information encoded by the feature spaces contained in R . The rows of \mathbf{Z} correspond to a new set of feature vectors for each node in \mathcal{G} . Each row of \mathbf{Z} is called an “embedding vector” and is a reduced-dimension representation of the whole set of original feature vectors of a given node.^{37,34} The embedding vectors are “multi-semantic” in that they contain information from every feature space in R . The dimension of embedding vectors, d , is generally lower with respect to the dimension of the original feature spaces. The embedding matrix \mathbf{Z} is represented in panel F of Figure 1.

Graph RL is also referred to as “graph embedding”. Giving a node $v \in \mathcal{V}$, and a specific edge-type $\tau \in \mathcal{T}_{\mathcal{G}}$, the neighborhood of v in τ is denoted as $\mathcal{N}_{\tau}(v)$ and corresponds to the set of nodes that are connected to v by τ -type edges. We denote with $\mathcal{N}(v)$ instead the set containing every neighborhood of v :

$$\mathcal{N}(v) = \{\mathcal{N}_{\tau}(v), \forall \tau \in \mathcal{T}_{\mathcal{G}}\}$$

The heterogeneous graph embedding process can be represented by an encoding function that takes in input a node $v \in \mathcal{V}$ and its neighborhood set, $\mathcal{N}(v)$, and returns a new representation for that node:

$$f : (v, \mathcal{N}(v)) \rightarrow z \in \mathbb{R}^d \quad (4)$$

One of the most common methodologies for implementing (4) is the encoder-decoder paradigm.²⁵ This paradigm also models an auxiliary decoding function. Considering a homogeneous graph, i.e. a graph with a unique node feature space X , the encoder function in (4) takes as input original feature vectors of X , and outputs the corresponding embedding vectors:

$$f(\mathbf{x}_i) = \mathbf{z}_i, \forall i \in |\mathcal{V}|$$

The decoding function, instead, takes as input a pair of node embeddings produced by the encoder, e.g. \mathbf{z}_i and \mathbf{z}_j , and outputs a real number. Such a number is the prediction of the value of a pre-defined pair-wise relationship between \mathbf{x}_i and \mathbf{x}_j :

$$\text{Dec} : (\mathbf{z}_i, \mathbf{z}_j) \rightarrow \hat{v}_{i,j} \in \mathbb{R} \quad (5)$$

where $\hat{v}_{i,j}$ is the predicted value of any pre-defined pair-wise relationship between \mathbf{x}_i and \mathbf{x}_j . For example, $\hat{v}_{i,j}$ could be the predicted cosine similarity between such vectors, or any other similarity/distance function.

The encoding and decoding functions should minimize a reconstruction loss of the form:

$$\mathcal{L} = \sum_{i,j \in |\mathcal{V}|} \Delta(\hat{v}_{i,j}, v_{i,j}) \quad (6)$$

where $v_{i,j}$ is the real value of the pair-wise relationship between \mathbf{x}_i and \mathbf{x}_j that (5) predicts, and Δ is a pre-defined discrepancy function that takes as input the predicted and real values for any pair of nodes. Once the encoding and

decoding functions that minimize (6) have been found, the original feature vectors can be fed to f to obtain the definitive graph embedding \mathbf{Z} . This embedding consists of a group of points in a low-dimension embedding space. In our experiment, we adapt the encoder/decoder paradigm to a heterogeneous graph context. Our main objective was for the the spatial distribution of these points in the embedding space to reflect plausible gene-expression regulation mechanisms. To this end, we proposed to find a parametric encoding function, and in particular, we chose to implement (4) using a deep graph auto-encoder as explained in the next paragraph.

Graph auto-encoders

A convenient framework that implements the encoding-decoding paradigm is the deep graph auto-encoder (deep GAE).⁴² In this framework, (4) was implemented through a graph neural network (GNN), an artificial neural network that takes the original feature vectors and the adjacency matrix of \mathcal{G} as input and produces the node embedding vectors. GNNs are non-linear parametric functions, and after a pre-defined initialization schema, the parameters of these functions can be optimized to minimize (6) through gradient descent.

In this work, we proposed the use of the basic GNN model presented in Ref. 25. In a homogeneous graph context, the basic GNN takes as input the node feature matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times m}$ and a pre-defined graph adjacency matrix denoted by $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$, and performs non-linear parametric operations with this information to produce \mathbf{Z} . In matrix notation, the operations made by the basic GNN model can be represented as follows:

$$\mathbf{Z} = \sigma \left(\mathbf{W}_{self} \mathbf{X} + \mathbf{W}_{neigh} \tilde{\mathbf{A}} \mathbf{X} \right) \quad (7)$$

where σ is a non-linear operator, the learnable parameter matrices \mathbf{W}_{self} and \mathbf{W}_{neigh} are responsible for combining the features of each node with the features of an aggregation over its neighborhood, and $\tilde{\mathbf{A}}$ is the symmetric-normalized adjacency matrix of \mathcal{G} :

$$\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}} (\mathbf{A}) \mathbf{D}^{-\frac{1}{2}}$$

where \mathbf{D} is the degree matrix of \mathbf{A} .

We defined a parametric “architecture” for the encoding function using (7), but still needed to define \mathbf{A} and \mathbf{X} , to drive the optimization of the parameters of f to minimize (6). The initialization of these matrices can follow multiple strategies. For example, in the neural message passing framework,⁴³ given a homogeneous graph context, one generally constructs \mathbf{X} stacking the original feature vectors as the rows of such a matrix. However, it is not the only valid strategy. In our heterogeneous graph context, we decided to initialize \mathbf{X} as the identity matrix because we had multiple original feature matrices. We were confident to use the identity matrix to initialize \mathbf{X} because we carefully initialized the adjacency matrix \mathbf{A} combining the information of every relationship type in our graph, as we will explain in the next paragraph.

Finally, the similarity measure that the decoder is meant to predict needs to be specified to implement (5). In this paper, we proposed the use of the decoding similarity function proposed by Ref. 26 to converge into a clustering-friendly embedding, i.e., an embedding with dense and easily identifiable clusters. In the next paragraph, we give a brief explanation of the algorithm in,²⁶ which performs graph RL on a single-feature space. For a more detailed exposition of the work of Xuelong *et al.*, the reader is referred to the original paper.²⁶

Adaptive-sparsity graph auto-encoder

In this work, we proposed the use of AdaGAE,²⁶ which is a state-of-the-art method for RL and clustering through graph modelling of a dataset. The whole AdaGAE algorithm is wrapped in a gray rectangle in Figure 1. This algorithm is defined in a homogeneous data context. In other words, given a unique original feature matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times m}$, the objective was to produce a low-dimensional embedding matrix $\mathbf{Z} \in \mathbb{R}^{|\mathcal{V}| \times d}$ where $d \ll m$. Authors in²⁶ used a deep GAE to embed high-dimensional datasets in a graph-like, low-dimensional format. Xuelong *et al.* generated a sparse graph as a function of the Euclidean distances between the original feature vectors using a k-nearest neighbors⁴⁴ (k-NN) model.

The main novelty of AdaGAE is the adaptiveness in the sparsity parameter k . The graph auto-encoder is, in fact, iteratively optimized with respect to various objective functions like (6), and each objective function is constructed using a different sparsity parameter. With a custom decoding function, such a dynamic model leads to a sparse graph. In other words, they produce a clustering-friendly embedding of the original data.

In other words, at each iteration, $l \in \{0, 1, 2, \dots, L-1\}$, a sparse and weighted graph is generated. In this graph, the nodes correspond to the samples of the original dataset. Instead, the weights of the edges are inversely proportional to the

Euclidean distances between the original feature vectors. Specifically, at iteration l , a graph is generated in which the weight of the link between nodes v_i and v_j is denoted as $p_{i,j}^l$ and is a function of a sparsity parameter k^l :

$$p_{i,j}^l = \left(\frac{d_{i,k^l+1} - d_{i,j}}{\sum_{j=1}^{k^l} (d_{i,k^l+1} - d_{i,j})} \right)_+, \quad \forall i,j \in |\mathcal{V}| \quad (8)$$

where $(\cdot)_+ = \max(\cdot, 0)$ is the rectifier operator, $d_{i,j}$ denotes the Euclidean distance between the feature vectors of v_i and v_j , and d_{i,k^l+1} denotes the distance between the feature vectors of v_i and its (k^l+1) -th nearest neighbour.

Notice that, in (8), the parameter k^l might have a different value at each iteration. Such a parameter induces the sparsity of the generated graph: for each node v_i , only up to k^l elements will have a non-zero weighted link with that node. Thus, at each iteration l , AdaGAE generates a k^l -sparse graph. Notice also that the function (8) is a discrete probability density function (PDF):

$$\begin{aligned} \mathbf{P}_i^l : (v_j) &\rightarrow p_{i,j} \in [0, 1] \\ \sum_{j \in |\mathcal{V}|} p_{i,j}^l &= 1 \end{aligned}$$

where a different PDF \mathbf{P}_i^l is defined for each single node in the dataset. Thus, at each iteration l , AdaGAE generates a probability distribution family \mathbf{P}^l :

$$\mathbf{P}^l = \left\{ \mathbf{P}_0^l, \mathbf{P}_1^l, \dots, \mathbf{P}_{|\mathcal{V}|}^l \right\}$$

where each PDF in \mathbf{P}^l contains information about the weights of the links between one node and the rest of the nodes in the graph. Authors in Ref. 26 proposed to re-initialize the adjacency matrix of the auto-encoder at each iteration using the family \mathbf{P}^l . In other words, at each iteration l , AdaGAE stacks the different distributions $\mathbf{P}_i^l, \forall i \in |\mathcal{V}|$ to form \mathbf{A} .

The embedding matrix \mathbf{Z} produced by AdaGAE was iteratively optimized. We denoted with \mathbf{Z}^l the embedding matrix after iteration l . The embedding vector of node v_i after iteration l , instead, coincides with the i -th row of \mathbf{Z}^l and is denoted as \mathbf{z}_i^l . At each iteration, AdaGAE proposes to implement the decoding function in (5) as a function of the embeddings of the previous iteration:

$$\mathbf{Q}^l(i,j) = q_{i,j}^l = \frac{\exp -\tilde{d}_{i,j}^{l-1}}{\sum_{j=1}^{|\mathcal{V}|} \exp -\tilde{d}_{i,j}^{l-1}} \quad (9)$$

where $\tilde{d}_{i,j}^{l-1}$ denotes the euclidean distance between the embedding vectors \mathbf{z}_i^{l-1} and \mathbf{z}_j^{l-1} . Notice that this encoding function is also a point-wise probability distribution function:

$$\begin{aligned} \mathbf{Q}_i^l : (v_j) &\rightarrow q_{i,j} \in [0, 1] \\ \sum_{j \in |\mathcal{V}|} q_{i,j}^l &= 1 \end{aligned}$$

and that we can group every PDF $\mathbf{Q}_i^l, \forall i \in |\mathcal{V}|$ in a PDF family \mathbf{Q}^l as we did with \mathbf{P}^l .

At each iteration l , AdaGAE proposes to use the Kullback-Leibler divergence (KL-divergence)⁴⁵ of \mathbf{P}^l with respect to \mathbf{Q}^l to implement (6). Explicitly, the reconstruction loss that AdaGAE seeks to minimize is the following:

$$\begin{aligned} \mathcal{L}^l &= \sum_{i \in |\mathcal{V}|} \text{KL}(\mathbf{P}_i^l \| \mathbf{Q}_i^l) \\ \text{where :} & \\ \text{KL}(\mathbf{P}_i^l \| \mathbf{Q}_i^l) &= \sum_{j \in |\mathcal{V}|} p_{i,j}^l \cdot \log \frac{p_{i,j}^l}{q_{i,j}^l} \end{aligned} \quad (10)$$

Xuelong *et al.* relied on the graph convolutional network (GCN) model⁴⁶ to implement the encoder of their deep GAE. However, in this paper, we used a simpler GNN model, as explained in the previous paragraph. We represented the iterative computation of (8) and (9) and the iterative optimization of (10) in panel E of Figure 1.

Notice that the main dynamic criterion that causes the objective functions to change is the sparsity parameter k^l . However, the sparsity is not the only thing that changes from iteration to iteration: In the first iteration, the distribution family \mathbf{P}^0 is generated through (8) using the Euclidean distances between the original feature vectors. After the first iteration, instead, the distribution family \mathbf{P}^l , $\forall l < L - 1$ is generated using (8) as a function of the Euclidean distances between the embedding vectors of the \mathbf{Z}^{l-1} embedding matrix. By doing so, AdaGAE considers high-order neighborhoods of each node to construct the final embedding matrix. In other words, AdaGAE is said to “exploit the high-level information” present in the data.

Different embeddings for various iterations are plotted in panel G of Figure 1. In those plots, one can see the embedding of a small set of genes and cCREs extracted from the original mouse heart dataset. Large points represent genes, while cCREs are the small points. Point colors, instead, correspond to single-modality or homogeneous cluster assignments, i.e., cluster assignments of genes and cCREs, taking into account their corresponding feature spaces separately. Note that the embedding turns more cluster-friendly at each iteration. In other words, the density and separation of clusters are increased at each iteration, and clusters can be caught visually in the final embedding in panel H of Figure 1. In AdaGAE, the sparsity parameter initialization, k_0 , the increment of this parameter from iteration to iteration, δk , and the number of iterations, L , regulate the number of clusters in the final embedding. In our experiment, we tested various parameter configurations and found eight significant GERM clusters with $L = 11$, $k_0 = 350$, $\delta k = 25$.

Notice that the original work of Xuelong *et al.* defined (8) as a function of the Euclidean distances over a unique feature space. We extended the AdaGAE framework to deal with multiple node feature spaces. In particular, at each iteration l , we combined the Euclidean distances of multiple node feature spaces - the gene expression and the cCRE activity time-series - to generate a unique distance function that is given to (8) to generate \mathbf{P}^l . In the next paragraph, we explain how we combined the feature spaces to adapt AdaGAE to heterogeneous graph RL.

Extending AdaGAE to heterogeneous graphs

As explained previously, we modelled the whole set of gene expression and cCRE-activity data as a heterogeneous graph. We proposed to extend the work in Ref. 26 to heterogeneous graph RL. We ran the same process described in the previous paragraph, with some differences. The first difference was the construction of the reference distance function to feed to (8). At each iteration l , we combined the Euclidean distances defined in each one of the original feature spaces to form a unique distance function \mathbf{D}^l that generated the sparse distribution family \mathbf{P}^l .

Recall that $R = \{G, C_1, C_2, \dots, C_M\}$ is the set of feature spaces defined, where G is the gene-expression feature space and C_1, C_2, \dots, C_M are the cCRE activity feature spaces, for example ATAC-seq, H3k27ac, aH3k27me3, etc. At each iteration l , for each feature space $r \in R$, we computed the Euclidean distances between the original feature vectors in r , scaled these distances into the interval $[0, 1]$, and denoted them with d_{ij}^r , $\forall i, j \in |\mathcal{V}|$. Notice that the Euclidean distances between the original feature vectors were only defined between same-class elements: The Euclidean distances in the gene-expression feature space G , were defined between genes, and the distances in C_1, C_2, \dots, C_M , only between cCREs. Consequently, we set the distance between different-class elements to the maximum value, i.e., 1, for each feature space. Finally, we create a custom linear combination of these distance functions to form a unique distance function $\hat{\mathbf{D}}^l$:

$$\hat{\mathbf{D}}^l : (v_i, v_j) \rightarrow \tilde{d}_{ij}^l \in \mathbb{R}, \forall i, j \in |\mathcal{V}|$$

where:

$$\tilde{d}_{ij}^l = \begin{cases} 1 - \alpha_G^l \cdot (1 - d_{ij}^G), & \text{if } v_i, v_j \in \mathcal{V}_{genes} \\ \frac{\sum_{m=0}^M 1 - \alpha_{C_m}^l \cdot (1 - d_{ij}^{C_m})}{M}, & \text{if } v_i, v_j \in \mathcal{V}_{cCREs} \\ 1, & \text{otherwise} \end{cases} \quad (11)$$

where \mathcal{V}_{genes} is the set of genes and \mathcal{V}_{cCREs} is the set of cCREs, d_{ij}^G is the Euclidean distance between the expression profile of gene i and gene j , $d_{ij}^{C_m}$ is the Euclidean distance between the C_m activity profiles of cCREs i and j , and α_G^l and $\alpha_{C_m}^l$ are fixed parameters that indicate the importance weights of the distances of gene-expression and cCRE-activity at iteration l , respectively.

At the beginning of the l -th iteration, the Euclidean distances between the embedding vectors of \mathbf{Z}^{l-1} are denoted by \mathbf{D}_Z^{l-1} . We combined such information with $\widehat{\mathbf{D}}$ defined in (11), to generate the unified distance function, \mathbf{D}^l as follows:

$$\mathbf{D}^l = \frac{\widehat{\mathbf{D}}^l + \widehat{\mathbf{D}}_Z^{l-1}}{2} \quad (12)$$

where $\widehat{\mathbf{D}}_Z^{l-1}$ is a *weighted version* of \mathbf{D}_Z^{l-1} :

$$\widehat{\mathbf{D}}_Z^{l-1} = 1 - \alpha_Z^l \cdot (1 - \mathbf{D}_Z^{l-1}) \quad (13)$$

In (13), α_Z^l is the fixed parametric importance weight of the Euclidean distances in \mathbf{D}_Z^{l-1} . Notice that, in (12), $\widehat{\mathbf{D}}_Z^{l-1}$ allows to exploit the high-level information in the data and $\widehat{\mathbf{D}}$ helps to reduce the loss of information with respect to the original feature spaces at each iteration.

At each iteration $l, \forall l \in [0, L-1]$, we computed \mathbf{D}^l using (12). This computation is represented in panel E of Figure 1. We then used \mathbf{D}^l and the corresponding sparsity parameter k^l to compute a connectivity distribution family $\widehat{\mathbf{P}}^l$ using (8). Notice that this distribution family was generated using only the same-class relation types in $\mathcal{T}_{\mathcal{G}} - \{S_{BP|T}\}$. In other words, we only used the node feature spaces in R to create $\widehat{\mathbf{P}}^l$. We needed to add the trend-aware base-pair proximity relationship $S_{BP|T}$ defined in (3) to compute the definitive distribution family \mathbf{P}^l that takes into account all information we have:

$$\mathbf{P}^l = \widehat{\mathbf{P}}^l + \omega_{BP}^l \cdot S_{BP|T} \quad (14)$$

where ω_{BP}^l is a parametric importance weight of the base-pair proximity information $S_{BP|T}$ during iteration l .

After computing $\widehat{\mathbf{P}}^l$, we computed \mathbf{Q}^l using (9), and optimized the embedding matrix \mathbf{Z} , minimizing a reconstruction loss of the form (6). AdaGAE proposed the use of (10) to implement the reconstruction loss. In this paper, however, we propose to extend such a loss function to better separate elements that are distant in the original feature spaces, as explained in the next paragraph.

Regularization of the loss function

The minimization of (10) has the same effect of an attractive force that tends to collapse points in the embedding space, reducing the distances between z_i and z_j proportionally to p_{ij}^l . When considering a single node v_i , such an objective function could lead to errors in the manipulation of the position of the most distant neighbors of such node.⁴⁷ Inspired by Ref. 48, in this work, we added a regularization term that acts as a repulsive force. Such a repulsive force tends to increase $\widehat{d}_{i,j}^l$ proportionally to the inverse of p_{ij}^l . In other words, we pushed elements away from each other proportionally to their distance \mathbf{D}^l in (12). To create this force, we added the KL-divergence of $1 - \mathbf{P}_i^l$ with respect to $1 - \mathbf{Q}_i^l$ to (10).

Notice that the union of the attractive and repulsive force is equivalent to the minimization of the binary cross-entropy of \mathbf{P}^l with respect to \mathbf{Q}^l as defined in Ref. 48. Lastly, we added parametric importance weights to the attractive and repulsive force components and constructed our final objective function:

$$\mathcal{L}^l = CE(\mathbf{P}^l, \mathbf{Q}^l) = \sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} [\psi_A^l \cdot \text{KL}(\mathbf{P}^l \| \mathbf{Q}^l) + \psi_R^l \cdot \text{KL}(1 - \mathbf{P}^l \| 1 - \mathbf{Q}^l)] \quad (15)$$

where ψ_A^l and ψ_R^l are the importance weights of the attractive and repulsive term during iteration l , respectively.

We iteratively optimized the parameters of our GNN-based encoder. At each iteration, we minimized (15) with a different set of parameters. After running the optimization for a predefined number of iterations, we processed to a clustering-friendly embedding where the clusters reflect plausible gene regulatory networks. We then ran k-means clustering on the embedding to find plausible gene expression regulatory mechanisms. This clusterization is represented in panel H of Figure 1.

Results

DeepReGraph performs the co-clustering of genes and cCREs together. Consequentially, resulting clusters are heterogeneous in the class of elements they might contain. DeepReGraph helped us identify eight co-clusters when applied to

developmental fetal mouse heart datasets. We assessed the quality of gene expression and cCREs clusters from a computational point of view. We also analyzed these clusters from a biological perspective as described below.

DeepReGraph generated high-quality co-clusters

We employed a principle component analysis (PCA) to visualize the cCRE clusters on dimension reduced plots as shown in panel A of Figure 2. This figure highlights a clear separation of clusters on *PC0*, *PC1*, and *PC2* which explains the 57% variability in the cCRE datasets. We also performed k-means clustering using $k=8$. Panel B of Figure 2 shows a confusion matrix that compares cCRE clusters from DeepReGraph with the uni-modal clustering produced with k-means clustering. It is clear from this confusion matrix that the result of the multi-modal clustering of DeepReGraph was highly similar to the uni-modal clustering one. This similarity indicates that DeepReGraph clustering does take into account the same-class pattern similarities when defining clusters. We also visualized the PCA plot of gene expression data in panel C of Figure 2. *PC0* by itself explains 76% of the variability in gene expression. Basically, gene expression has two distinct patterns throughout mouse fetal heart tissue development: genes with increasing expression and genes with decreasing expression. Interestingly, these two major patterns have been divided into sub-patterns based on the cCRE clusters that plausibly drive their regulation.

The clustered patterns produced by DeepReGraph are presented in Figure 3. In this figure, the y-axis contains the mean-centered values for gene expression in the left-most column, while mean-centered CRE features are in the last three columns. The mean value reduction process mentioned was done as follows: given a time-series vector $\mathbf{x} = [x_0, x_1, \dots, x_r]$, with a mean value $\hat{x} = \frac{\sum_{i=0}^r x_i}{|x|}$, the mean-reduced vector is $\tilde{\mathbf{x}} = [x_0 - \hat{x}, x_1 - \hat{x}, \dots, x_r - \hat{x}]$. The x-axes of the plots instead correspond to the considered time-points of mouse fetal heart development. Notice that each cluster contains a set of genes and cCREs. The area between the first and 0.75 quantile is colored for each trend plot, to help visualize the trend of each cluster.

DeepReGraph revealed the regulatory signature of mouse fetal heart development

We investigated the enriched function of gene expression clusters and enriched transcription factor binding site motifs of cCRE clusters to decipher the general signature of mouse fetal heart development. Figure 4 summarizes these enrichment

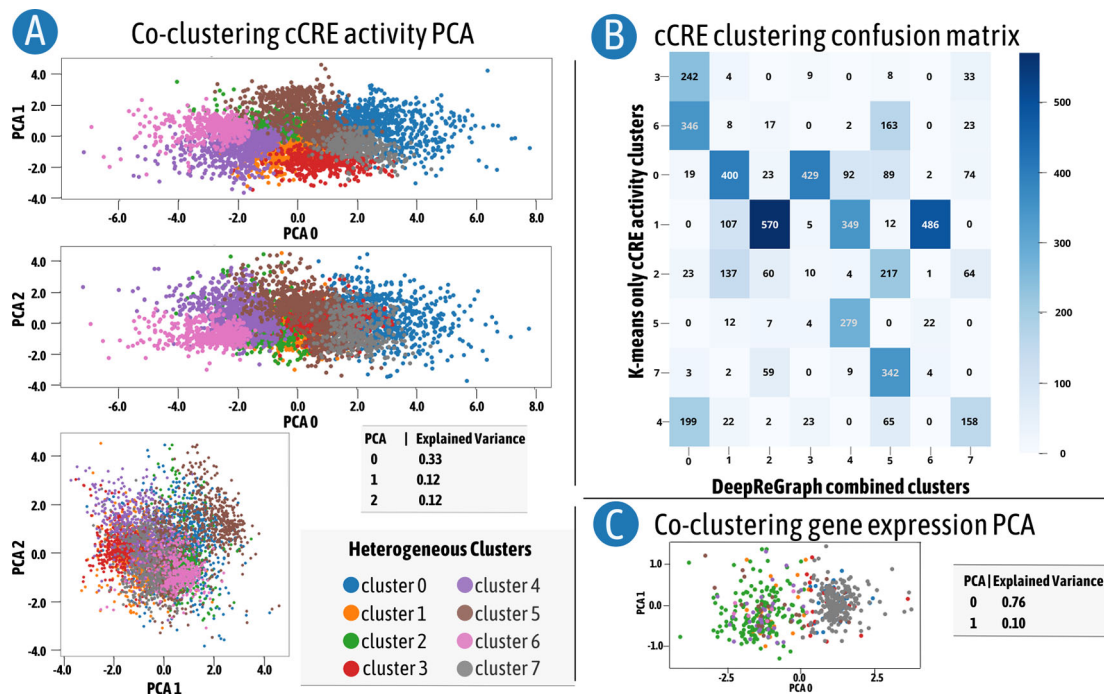


Figure 2. Intersections between single-modality k-means clusters and co-clusters induced by DeepReGraph. A) Principal component analysis (PCA) reduced dimension of candidate cis-regulatory elements (cCRE) profiles colored by the correspondent DeepReGraph cluster. B) Intersection between only-cCRE agglomerative clustering and cCRE extracted from DeepReGraph heterogeneous clusters. C) PCA reduced dimension of gene expression profiles colored by the corresponding DeepReGraph cluster.

DeepReGraph Co-clustering Trends

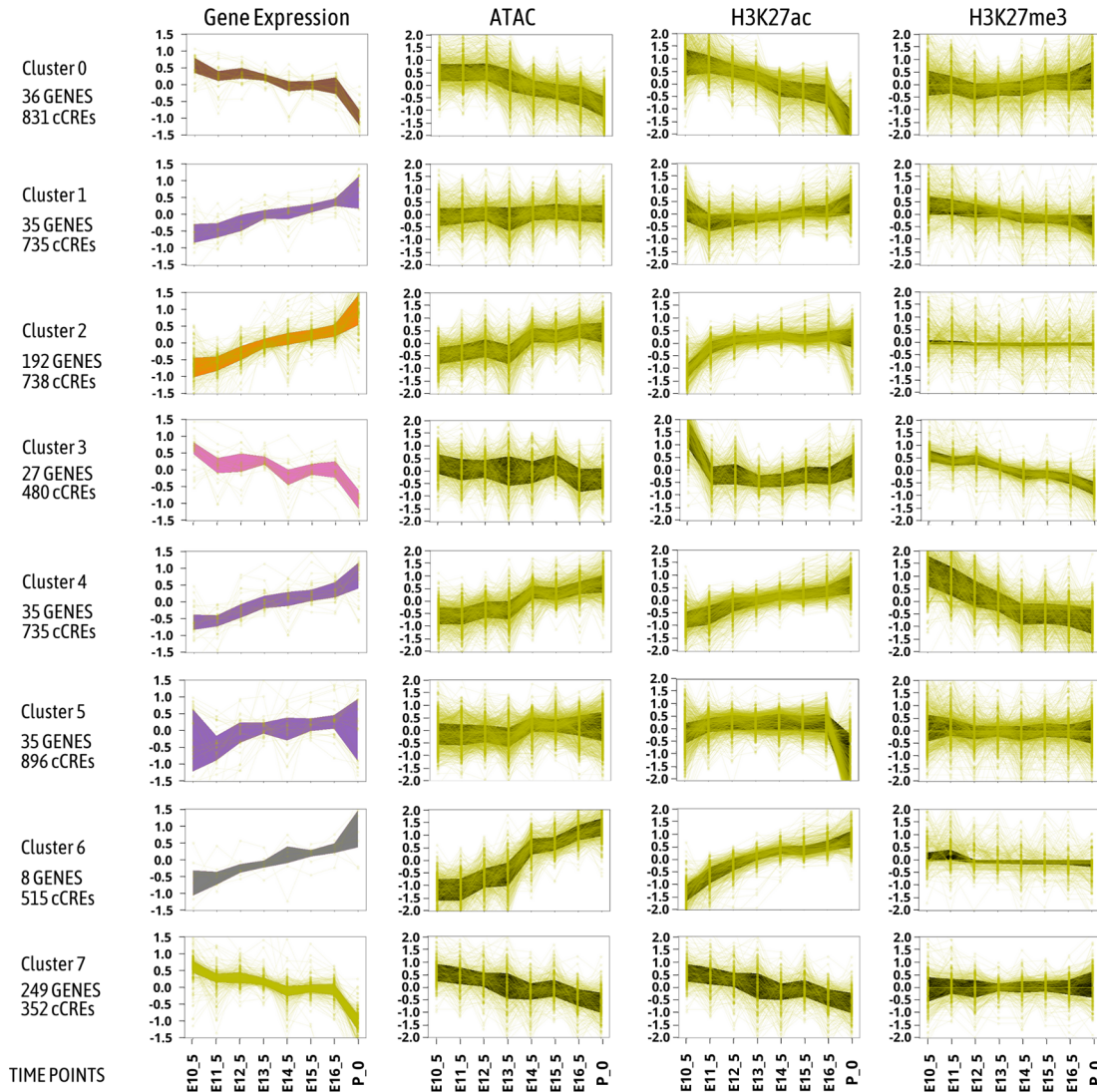


Figure 3. Co-clusters identified by DeepReGraph and the relative profile trends. The first column of plots contains gene expression time profiles, and the rest of the columns contain enhancer activity time profiles. The space between the first and third quartile for each plot was colored to better show the trend.

analysis. It clearly shows that all gene expression clusters have clear functional annotation and all cCRE clusters entail clear transcription factor binding site motifs.

In general, expression of genes related to cell proliferation functions decrease during mouse fetal development, while expression of genes related to heart functions increase.⁴ However, if we look at the enriched terms for the detected gene expression clusters in panel A of Figure 3, and the pattern of gene expression the cCREs that drive them in Figure 3, we can observe that the story is not so simple. Two of the largest gene expression clusters are cluster2 (192 genes) which represents the heart functional genes (enriched for heart contraction function), and cluster7 (249 genes) which represents the cell proliferation genes. The smaller gene expression clusters have more specific enriched functions. Considering the other gene expression clusters down-regulated during development, cluster0 was more enriched for DNA replication, while cluster3 was enriched for non-heart developmental processes. Similarly, the smaller gene expression clusters up-regulated during development gained specific functional enrichment: cluster1 was enriched in metabolic processes to generate energy for the heart to function. Cluster4 was enriched for ventricular cardiac muscle cell membrane repolarization, and cluster6 was enriched for heart contraction. Cluster5 contained genes enriched for regulation of

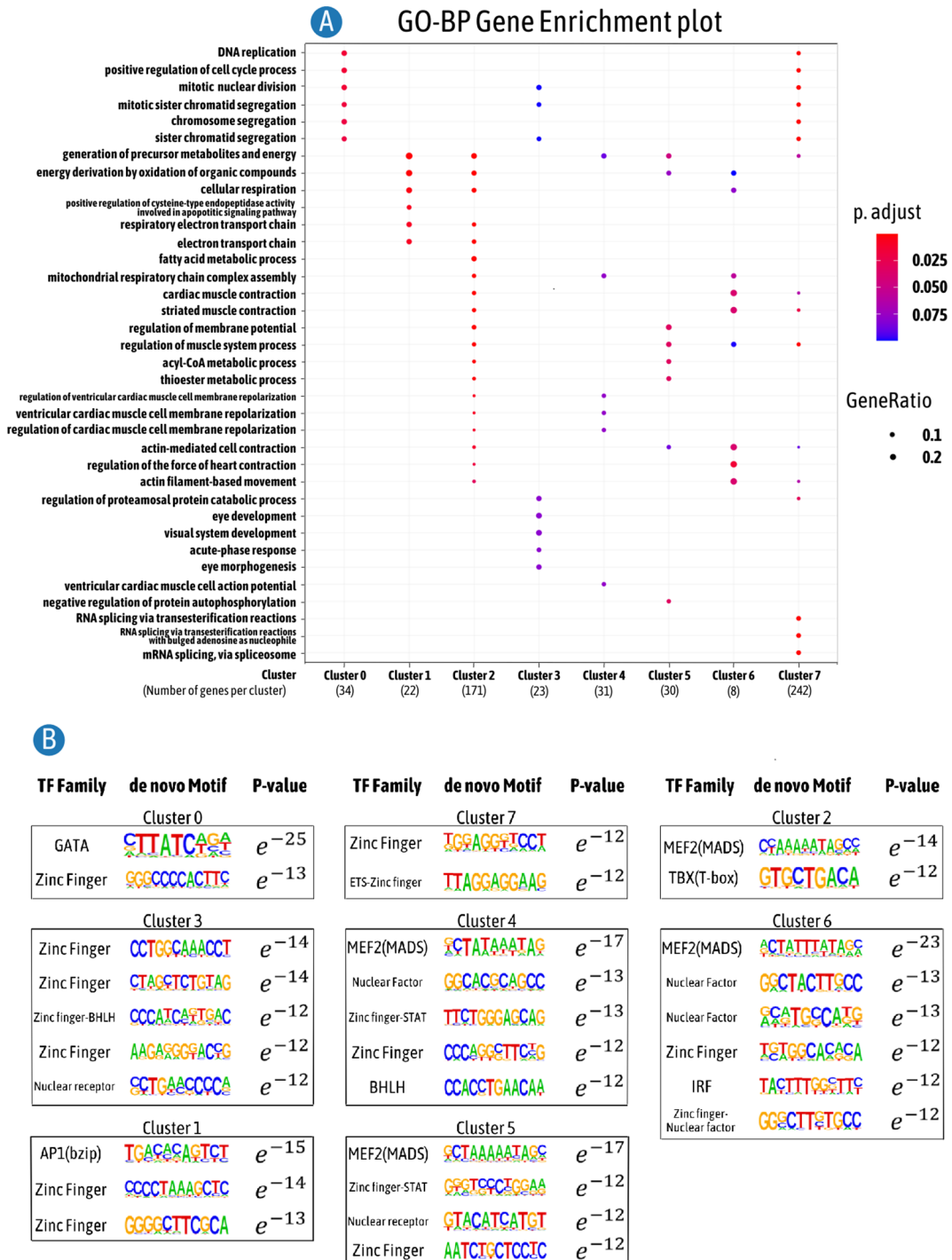


Figure 4. A) Enriched function of gene expression extracted from DeepReGraph clusters. B) Enriched transcription factor binding site motifs of cCRE for the same clusters.

muscle system process. Here, co-clustering of gene expression and cCREs enabled us to derive smaller and more specific clusters. Otherwise, as it is clear from Figure 2 panel C, the smaller clusters of gene expression with more focused functions could not be deduced from gene expression alone.

Multi-modal clustering can also describe how cCREs can drive gene expression during development. First, smaller gene expression clusters gained more cCREs per genes, as Figure 3 shows. Secondly, cCREs with different epigenetic patterns

have been linked to similar pattern of gene expression, as can be seen also in [Figure 3](#). For example, in the major cluster with up-regulated genes during development, i.e. cluster2, the trends of ATAC-seq and H3K27ac increased as expected. Similarly, for cluster7 which entails a large set of down-regulated genes throughout development, the trend of ATAC-seq and H3K27ac decreased. However, the H3K27me3 pattern for both cluster2 and cluster7 showed no changes in [Figure 3](#). We can observe polycomb removal events in cluster1, cluster3, and cluster4 as H3K27me3 levels decreased in these clusters.

Multi-modal clustering can further clarify how gene expression changes through development, as cCRE clusters exhibited clearly enriched motifs in panel B of [Figure 4](#). These enriched motifs can prioritize the candidate transcription factors which drive the development. For example, the MEF2 motif was enriched in the clusters related to up-regulated genes. The only exception was cluster1, for which the AP1 motif was enriched. We can assume that a MEF2 binding transcription factor, and more probably MEF2C, was the major transcription factor which caused the increase in gene expression values. Other binding site motifs were been enriched for the clusters entailing temporally up-regulated genes in [Figure 4](#) panel B. Interestingly, these motifs were highly cluster-specific. A similar dynamic was seen for clusters with temporally decreased gene expression. Although zinc finger motifs were enriched in these clusters, the GATA motif was only enriched in cluster0 and the basic helix-loop-helix (BHLH), nuclear receptor motifs are only enriched in cluster3.

Discussion

This study introduced a novel method called DeepReGraph to perform multi-modal clustering of gene expression and cCREs. DeepReGraph allows a cluster-friendly embedding, where clusters contain genes and CREs and tend to identify gene regulation mechanisms. Interestingly, the results of multi-modal clusters derived by DeepReGraph for cCREs were highly similar to the uni-modal clustering using k-means. However, DeepReGraph generated gene expression clusters that could not be derived by using gene expression data alone. Such a result might be expected if we consider cCRE and gene expression changes in a “cause and effect” manner. cCREs are part of the regulatory network and are among the driving causes of alternation in gene expression. Therefore, we can expect cCRE uni-modal clustering to be similar to co-clustering gene expression and cCREs together. However, gene expression is controlled by cCREs. In mouse fetal heart development, we have shown that similar gene expression (similar effect) can be divided into different clusters based on the controlling cCREs. This result shows the added values of the multi-modal clustering method to understand the signature of development.

Developmental regulatory networks can be straightforwardly modelled as heterogeneous graphs. The main reason for such a claim is because they are made of two distinct classes of elements (genes and CREs) whose interaction tends to be highly correlated with multiple features like gene expression, base-pair distance, and cCRE activity. Modelling such regulatory networks as heterogeneous graphs is key to using graph RL algorithms to converge to low-dimensional embeddings for such systems. The spatial distribution of nodes in the embedding might resemble complex relationships between nodes.

We undertook the challenge of converging to an embedding where gene expression regulation mechanisms are easily identifiable. To do so, we created our own heterogeneous graph RL algorithm by carefully designing an extension of AdaGAE.²⁶ First, we designed a dynamic combination schema of multiple node feature spaces to create a unique node feature space. This unification was a necessary step to adapt AdaGAE to a heterogeneous graph scenario. We also created a repulsive force by extending the loss function in Ref. 26. This repulsive force has proven essential to separate elements with different gene expression or activity trends. Third, we introduce a trend-aware regularization of the base-pair distance relationship between nodes. This regularization proved essential to produce more compact clusters. The resulting schema is responsible for producing a clustering-friendly embedding space that sheds light on regulatory mechanisms.

In this work, we extended the algorithm presented in Ref. 26 to produce an algorithm capable of embedding a heterogeneous graph into a low-dimensional, easy-to-cluster embedding. Other approaches to heterogeneous graph embedding that we could further investigate exist³⁴; for example, the relational graph convolutional networks (RGCN).⁴⁹ Such a model implies a greater number of parameters; various parameter-sharing approaches have been proposed, some of them making use of the attention mechanism.^{50,51} We could further investigate attention-based prioritization of nodes and relationships for learning embeddings.⁵² If we consider the production of a unified embedding of heterogeneous data as a first step, we could conceive other offline clustering algorithms. The clustering-friendly embedding we present resembles a differentiable version of agglomerative clustering. However, other algorithms like the ones in Refs. 53,54 use a differential expectation-maximization schema, where a distance-to-centroid loss is minimized to reach final embedding with compact clusters. Consequently, the use of different deep clustering approaches should be further investigated.

Regulatory networks and gene regulation are dynamic processes. Therefore, temporal datasets can potentially describe them better than static ones. However, initial efforts to associate regulatory networks and chromatin states to the gene expression were made based on limited data. ChromHMM⁵⁵ is the most used method to assign states of the chromatin to genes. However, with the advent of large temporal datasets like ENCODE, which we used in this paper, it is possible to move beyond a static view of regulatory networks and gene expression. This study used chromatin accessibility, H3K27ac, and H3K27me3, three well-known epigenetic markers with a well-characterized effect on gene expression. This framework can be further expanded to other epigenetic markers. Such an expansion could have two main advantages. The first advantage is the potential improvement of clustering quality. The second consists in better deciphering the combinatorial trends of epigenetic changes and their effects on gene expression dynamics.

Data and software availability

Source code, Data and Interactive Notebook available at: <https://github.com/QuertyJacob/DeepReGraph>. Archived source code, data and notebook at time of publication: <https://doi.org/10.5281/zenodo.6416055>

Data are available under the terms of the Apache License, Version 2.0

Acknowledgements

The authors wish to thank Prof. L. Casasús and Dr. Raúl P. Caulier for their valuable insights.

References

- Gorkin DU, Barozzi I, Zhao Y, *et al.*: **An atlas of dynamic chromatin landscapes in mouse fetal development.** *Nature*. July 2020; **583**(7818): 744–751.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Pagiatakis C, Musolino E, Gornati R, *et al.*: **Epigenetics of aging and disease: a brief overview.** *Aging Clin. Exp. Res.* April 2021; **33**(4): 737–745.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Zboril E, Yoo H, Chen L, *et al.*: **Dynamic interactions of transcription factors and enhancer reprogramming in cancer progression.** *Front. Oncol.* September 2021; **11**: 0 753051.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Wittkopp PJ, Kalay G: **Cis-regulatory elements: molecular mechanisms and evolutionary processes underlying divergence.** *Nat. Rev. Genet.* December 2011; **13**(1): 59–69.
[PubMed Abstract](#) | [Publisher Full Text](#)
- McDowell IC, Manandhar D, Vockley CM, *et al.*: **Clustering gene expression time series data using an infinite gaussian process mixture model.** *PLoS Comput. Biol.* January 2018; **14**(1): e1005896.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Özgül OF, Bardak B, Tan M: **A convolutional deep clustering framework for gene expression time series.** *IEEE/ACM Trans. Comput. Biol. Bioinform.* November 2021; **18**(6): 2198–2207.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Sun S, Liang M, Dong Z, *et al.*: *Multiview Machine Learning*. Springer; January 2019.
- Nguyen ND, Wang D: **Multiview learning for understanding functional multiomics.** *PLoS Comput. Biol.* April 2020; **16**(4): e1007677.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Li Y, Fang-Xiang W, Ngom A: **A review on machine learning principles for multi-view biological data integration.** *Brief. Bioinform.* March 2018; **19**(2): 325–340.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Stanley JS 3rd, Gigante S, Wolf G, *et al.*: **Harmonic alignment.** *Proc. SIAM Int. Conf. Data Min.* 2020; **2020**: 316–324.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Liu J, Huang Y, Singh R, *et al.*: **Jointly embedding multiple single-cell omics measurements.** *Algorithms Bioinform.* September 2019; **143**.
- Stuart T, Butler A, Hoffman P, *et al.*: **Comprehensive integration of Single-Cell data.** *Cell*. June 2019; **177**(7): 1888–1902.e21.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Argelaquet R, Arnol D, Bredikhin D, *et al.*: **MOFA+: a statistical framework for comprehensive integration of multi-modal single-cell data.** *Genome Biol.* May 2020; **21**(1): 111.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Yuan Y, Bar-Joseph Z: **Deep learning for inferring gene relationships from single-cell expression data.** *Proc. Natl. Acad. Sci. U. S. A.* December 2019; **116**: 27151–27158.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Eraslan G, Avsec Ž, Gagneur J, *et al.*: **Deep learning: new computational modelling techniques for genomics.** *Nat. Rev. Genet.* July 2019; **20**(7): 389–403.
[Publisher Full Text](#)
- Amodio M, Krishnaswamy S: **MAGAN: Aligning biological manifolds.** Dy J, Krause A, editors. *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*. PMLR; 2018; pages 215–223.
- Yang KD, Belyaeva A, Venkatachalapathy S, *et al.*: **Multi-domain translation between single-cell imaging and sequencing data using autoencoders.** *Nat. Commun.* January 2021; **12**(1): 31.
[Publisher Full Text](#)
- Xi J, Yu Z: *Unsupervised Learning Models for Unlabeled Genomic, Transcriptomic & Proteomic Data*. Frontiers Media SA; January 2022.
- Huang Y-A, Pan G-Q, Wang J, *et al.*: **Heterogeneous graph embedding model for predicting interactions between TF and target gene.** *Bioinformatics.* March 2022; **38**: 2554–2560.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Wang X, Gong Y, Yi J, *et al.*: **Predicting gene-disease associations from the heterogeneous network using graph embedding.** *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. November 2019; pages 504–511.
- Zhou K, Zhang S, Wang Y, *et al.*: **High-quality gene/disease embedding in a multi-relational heterogeneous graph after a joint matrix/tensor decomposition.** *J. Biomed. Inform.* February 2022; **126**: 103973.
[Publisher Full Text](#)
- Nourani E: **GoVec: Gene ontology representation learning using weighted heterogeneous graph and Meta-Path.** *J. Comput. Biol.* December 2021; **28**(12): 1196–1207.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Rao A, Vg S, Joseph T, *et al.*: **Phenotype-driven gene prioritization for rare diseases using graph convolution on heterogeneous networks.** *BMC Med. Genet.* July 2018; **11**(1): 57.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Mei X, Cai X, Yang L, *et al.*: **Relation-aware heterogeneous graph transformer based drug repurposing.** *Expert Syst. Appl.* March 2022; **190**: 116165.
[Publisher Full Text](#)

25. Hamilton WL: *Graph Representation Learning*. Morgan & Claypool Publishers; September 2020.
26. Li X, Zhang H, Zhang R: **Adaptive graph Auto-Encoder for general data clustering**. *IEEE Trans. Pattern Anal. Mach. Intell.* 2021; 1–1. [PubMed Abstract](#) | [Publisher Full Text](#)
27. Wu T, Hu E, Xu S, *et al.*: **clusterprofiler 4.0: A universal enrichment tool for interpreting omics data**. *Innovation (N Y)*. August 2021; 2(3): 100141. [PubMed Abstract](#) | [Publisher Full Text](#)
28. Heinz S, Benner C, Spann N, *et al.*: **Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities**. *Mol. Cell*. May 2010; 38(4): 576–589. [Publisher Full Text](#)
29. Ross-Innes CS, Stark R, Teschendorff AE, *et al.*: **Differential oestrogen receptor binding is associated with clinical outcome in breast cancer**. *Nature*. January 2012; 481(7381): 389–393. [Publisher Full Text](#)
30. Liao Y, Smyth GK, Shi W: **The R package rsubread is easier, faster, cheaper and better for alignment and quantification of RNA sequencing reads**. *Nucleic Acids Res.* May 2019; 47(8): e47. [PubMed Abstract](#) | [Publisher Full Text](#)
31. Ernst J, Nau GJ, Bar-Joseph Z: **Clustering short time series gene expression data**. *Bioinformatics*. June 2005; 21 Suppl 1: i159–i168. [PubMed Abstract](#) | [Publisher Full Text](#)
32. Holmes SH, Huber W: *Modern Statistics for Modern Biology*. Cambridge University Press; 2018.
33. Yi B, Wang X, Li K, *et al.*: **A comprehensive survey of network function virtualization**. *Comput. Netw.* March 2018; 133: 212–262. [Publisher Full Text](#)
34. Shi C, Wang X, Yu PS: *Heterogeneous Graph Representation Learning and Applications*. Singapore: Springer; January 2022.
35. Chen F, Wang Y-C, Wang B, *et al.*: **Graph representation learning: a survey**. *APSIPA Transactions on Signal and Information Processing*. 2020; 9. [Publisher Full Text](#)
36. Yi H-C, You Z-H, Huang D-S, *et al.*: **Graph representation learning in bioinformatics: trends, methods and applications**. *Brief. Bioinform.* September 2021.
37. Xie Y, Bin Y, Lv S, *et al.*: **A survey on heterogeneous network representation learning**. *Pattern Recogn.* August 2021; 116: 107936. [Publisher Full Text](#)
38. Hamilton WL, Ying R, Leskovec J: **Representation learning on graphs: Methods and applications**. September 2017.
39. Zheng B, Sage M, Sheppard EA, *et al.*: **Engineering mouse chromosomes with Cre-loxP: range, efficiency, and somatic applications**. *Mol. Cell. Biol.* January 2000; 20(2): 648–655. [PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
40. Mora A, Sandve GK, Gabrielsen OS, *et al.*: **In the loop: promoter-enhancer interactions and bioinformatics**. *Brief. Bioinform.* November 2015; 17(6): 980–995. [PubMed Abstract](#) | [Publisher Full Text](#)
41. Dong X, Weng Z: **The correlation between histone modifications and gene expression**. *Epigenomics*. April 2013; 5(2): 113–116. [PubMed Abstract](#) | [Publisher Full Text](#)
42. Baldi P: **Autoencoders, unsupervised learning, and deep architectures**. Guyon I, Dror G, Lemaire V, *et al.*, editors. *Proceedings of ICML Workshop on Unsupervised and Transfer Learning, volume 27 of Proceedings of Machine Learning Research*. Bellevue, Washington, USA: PMLR; July 2012; pages 37–49.
43. Gilmer J, Schoenholz SS, Riley PF, Vinyals O, *et al.*: **Neural message passing for quantum chemistry**. Precup D, Teh YW, editors. *Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research*. PMLR; 2017; pages 1263–1272.
44. Kramer O: **K-Nearest neighbors**. Kramer O, editor. *Dimensionality Reduction with Unsupervised Nearest Neighbors*. Berlin Heidelberg, Berlin, Heidelberg: Springer; 2013; pages 13–23.
45. Kapoor R, Gupta R, Son LH, *et al.*: **Boosting performance of power quality event identification with KL divergence measure and standard deviation**. *Measurement*. October 2018; 126: 134–142. [Publisher Full Text](#)
46. Wu Z, Pan S, Chen F, *et al.*: **A comprehensive survey on graph neural networks**. *IEEE Trans Neural Netw Learn Syst.* January 2021; 32(1): 4–24. [Publisher Full Text](#)
47. Oskolkov N: **How exactly umap works**. Nov 2019. [Reference Source](#)
48. McInnes L, Healy J, Melville J: **UMAP: Uniform manifold approximation and projection for dimension reduction**. February 2018.
49. Schlichtkrull M, Kipf TN, Bloem P, *et al.*: **Modeling relational data with graph convolutional networks**. *The Semantic Web*. Springer International Publishing; 2018; pages 593–607.
50. Marcheggiani D, Titov I: **Encoding sentences with graph convolutional networks for semantic role labeling**. March 2017.
51. Sinha K, Sodhani S, Dong J, *et al.*: **CLUTRR: A diagnostic benchmark for inductive reasoning from text**. 2019.
52. Zhou S, Jiajun B, Wang X, *et al.*: **HAHE: Hierarchical attentive heterogeneous information network embedding**. January 2019.
53. Yang B, Xiao F, Sidiropoulos ND, *et al.*: **Towards k-means-friendly spaces: Simultaneous deep learning and clustering**. Precup D, Teh YW, editors. *Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research*. PMLR; 2017; pages 3861–3870.
54. Sadeghi M, Armanfard N: **Deep clustering with self-supervision using pairwise data similarities**. June 2021.
55. Ernst J, Kellis M: **ChromHMM: automating chromatin-state discovery and characterization**. *Nat. Methods*. February 2012; 9(3): 215–216. [PubMed Abstract](#) | [Publisher Full Text](#)

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research