# Survey of Graphical Neural Nets in Biology

Dylan Molho

Michigan State University

1. scDeepsort
2. scGNN
3. DSTG
4. Imputing scRNA via combining GCN and AE NN (GraphSCI)
5. Disease Prediction
   - Graph Convolutional Filtering for Breast Cancer Subtype Classification
   - DeepPaN: Deep Patient GCN

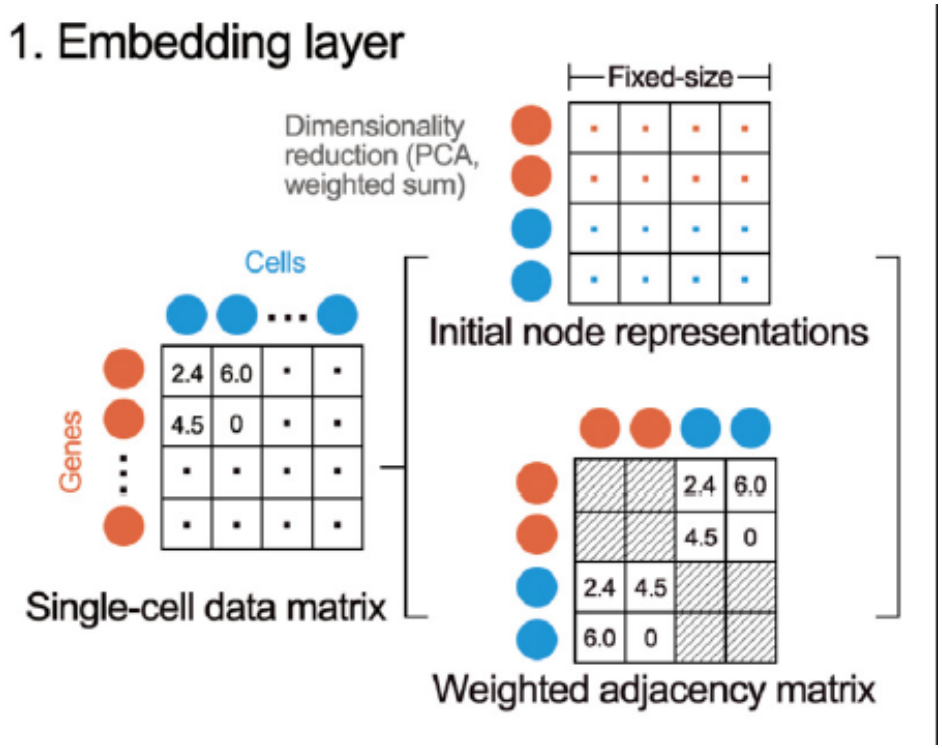
# scDeepSort

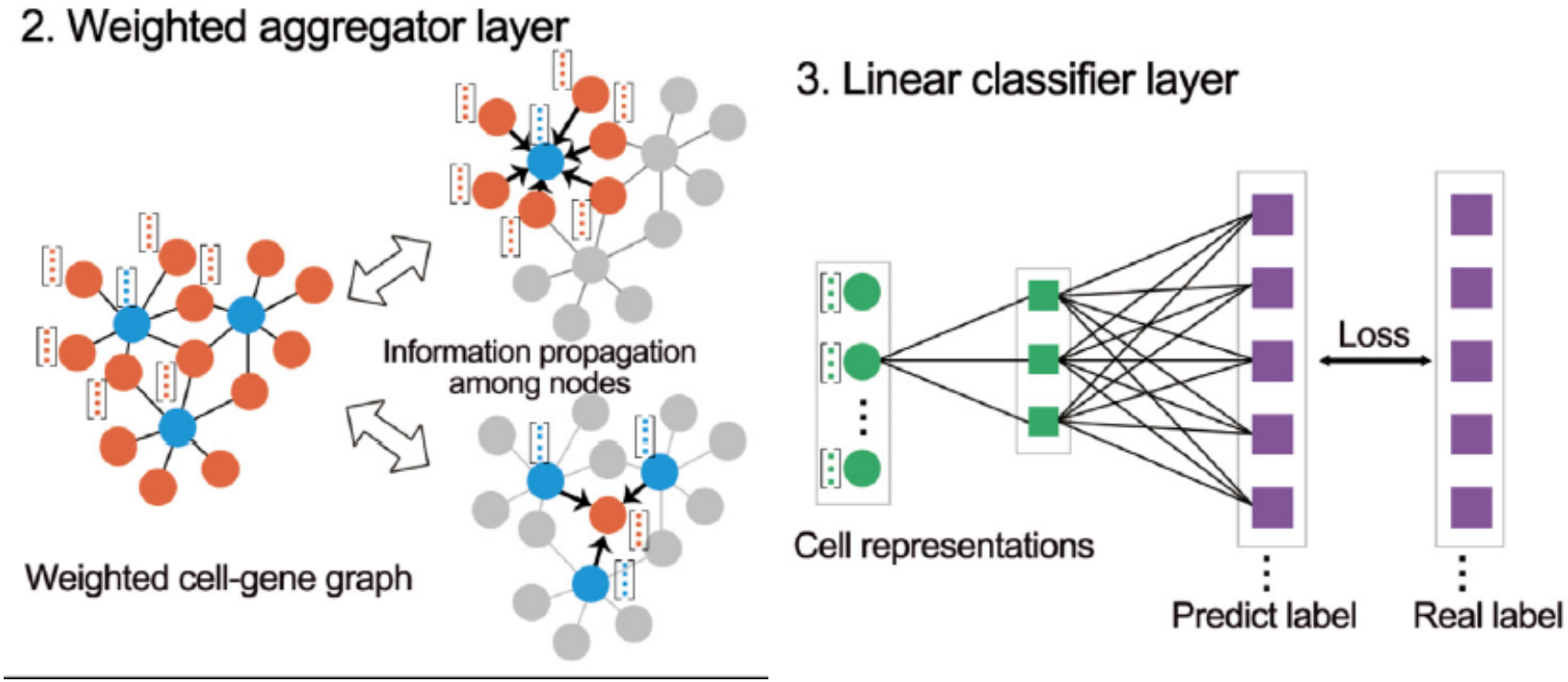
- A main goal in dealing with scRNA data is to do cell type identification (annotation), imputation often serves as a subgoal for this.
- scDeepSort is a method for annotation which uses a weighted GNN framework leveraging cell and genes relationships which produces a natural graph structure.
- scDeepSort consists of three components: an embedding layer, weighted graph aggregator, and linear classifier layers.
- The model can be trained on large reference datasets before running on query dataset with same genes using learned relationships on genes to produce superior node embeddings for classification.

# Embedding Layer

- To construct weighted cell-gene graph, cells and genes are both treated as nodes.
- The G.E. of a cell for a certain gene gives the weighted edge between the two nodes.
- Each node needs a feature vector.
- Gene nodes are given features via PCA on the G.E. matrix: if $D \in \mathbb{R}^{m \times n}$ is the $m$ genes and $n$ cell G.E. matrix, then PCA gives $D' \in \mathbb{R}^{m \times k}$ (k = 200, 400...).
- Then the cell nodes are given features by performing a weighted summation of these gene vectors, where each weight is given by the G.E. of that gene for that cell:

$$\text{cell}_j = \sum_{i=1}^{m} D_{ij} D'^{\top}_{[i,:]} = D'^{\top} D$$


1. Embedding layer
Fixed-size
Dimensionality reduction (PCA, weighted sum)
Cells
Genes
2.4
6.0
4.5
0
Single-cell data matrix
Initial node representations
Weighted adjacency matrix


2. Weighted aggregator layer
Information propagation among nodes
Weighted cell-gene graph
3. Linear classifier layer
Loss
Cell representations
Predict label
Real label

# Graph Filter

- Once the graph representation is aggregated into $X \in \mathbb{R}^{(m+n)\times d}$ and $A \in \mathbb{R}^{(m+n)^2}$ we put this structure through the layers of a GNN.
- In GNN's one seeks to place the nodes through iterated transformations that take into account the graph structure; one either performs spectral or spatial transformations, both called graph filters.
- scDeepSort uses a well-known spatial filter called GraphSAGE, which transforms a node along with information taken from its local neighborhood.
- The filter must be altered to take into account the information of the weighted edges.

- Let $h_i^k$ be the embedding of node $i$ in the $k$th layer.
- The weighted graph aggregator layer is

$$h_i^k = \sigma\left(W^{k-1} AGG\left(h_i^{k-1}, h_{N(i)}^{k-1}\right) + b^{k-1}\right)$$

  where $N(i)$ is the set of one-hop neighbors of node $i$, and $\sigma$ is a standard activation function (ReLU).
- The choice of the $AGG$ function creates different filters; choices include mean aggregator, pooling (max) operator, or LSTM aggregator. In the operations, one can first apply a (learned) linear transformation, or a full set of NN layers before applying the aggregation.

- The authors use the mean $AGG$ with some alterations: first the weighted adjacency matrix is normalized by

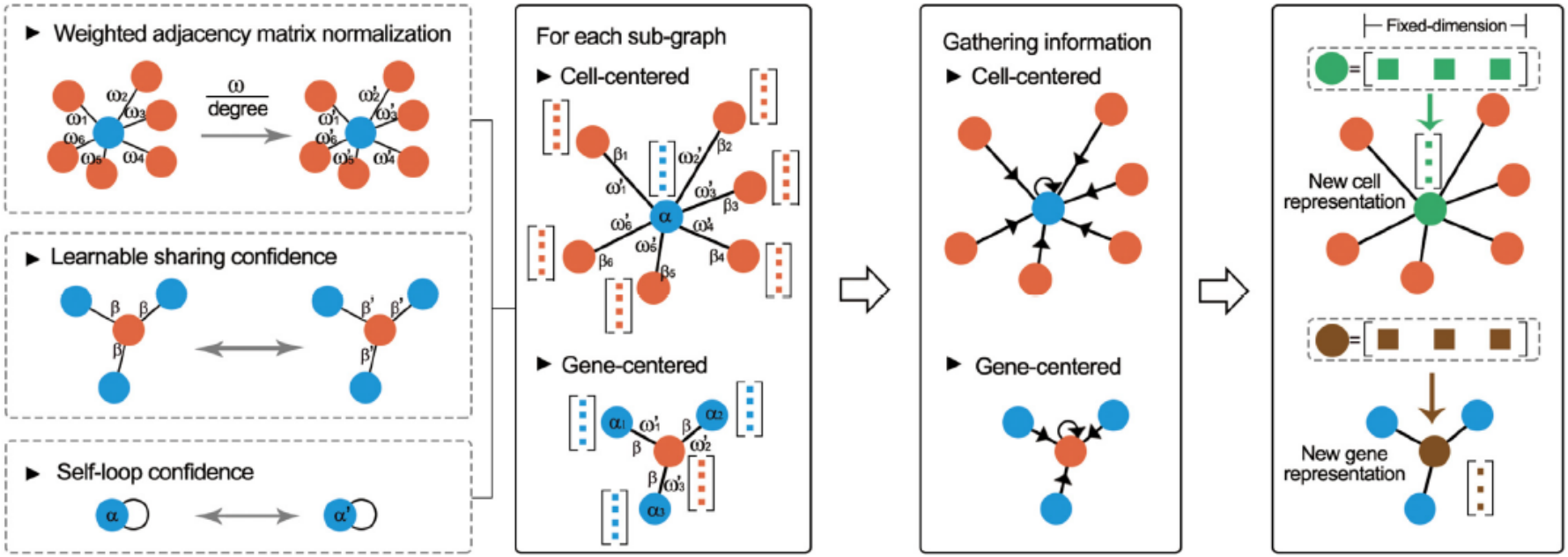
$$a'_{ij} = d_i \cdot \frac{a_{ij}}{\sum_{j \in N(i)} a_i j}$$

where $d_i$ is number of connected genes to cell $i$.

- The second change is adding learnable confidence parameters. Each edge will have a corresponding parameter weighing the connection of the neighbor, including a parameter for nodes themselves (as if there was a self loop.)

$$h_i^k = \left( W^{k-1} \frac{\alpha h_i^{k-1} + \sum_{j \in N(i)} \beta_j a'_{ij} h_j^{k-1}}{1 + |N(i)|} + b^{k-1} \right).$$

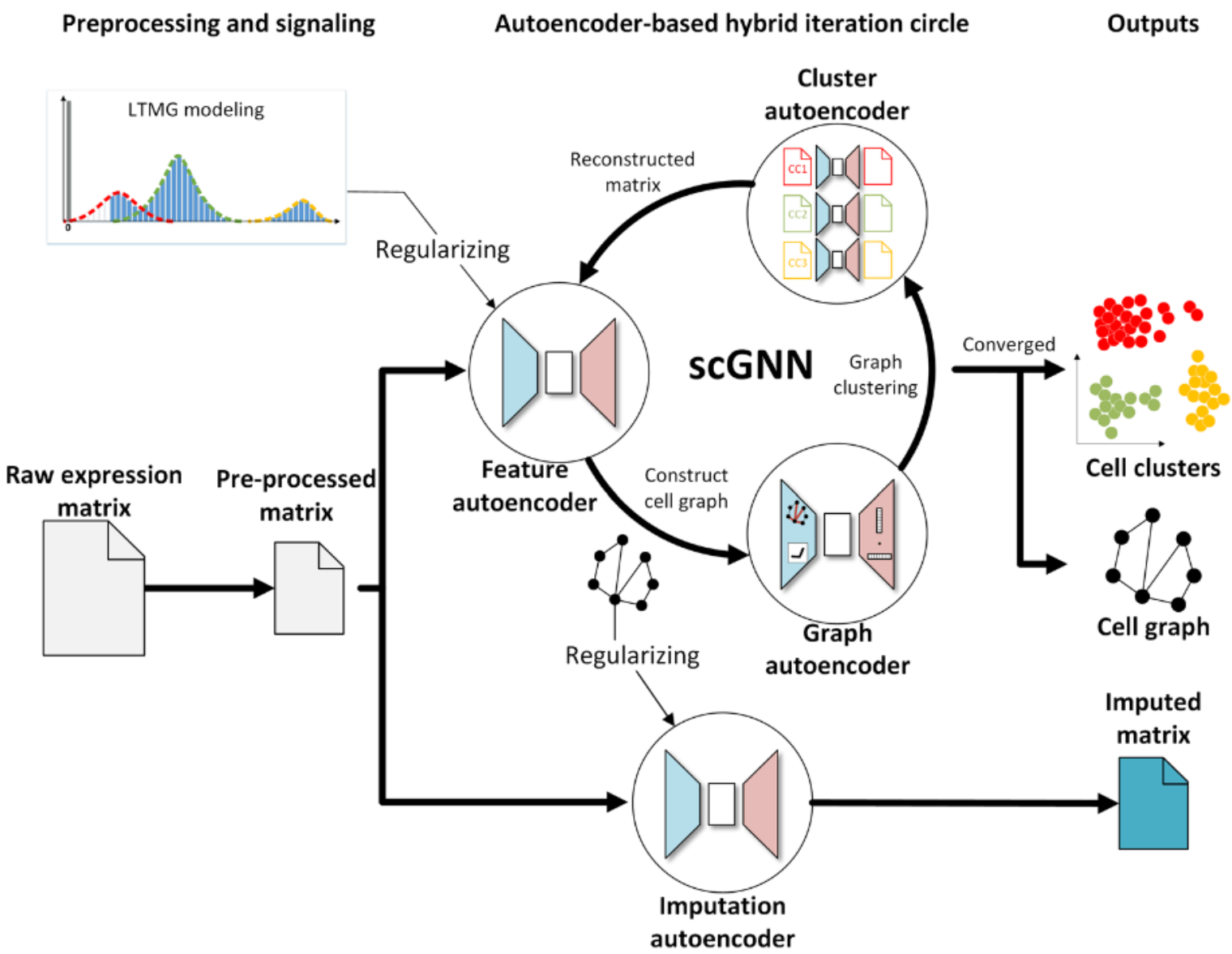
Here the $\beta_j$ are the shared confidence parameters for edges to genes $j$, and $\alpha$ is a shared confidence parameter for nodes.


Weighted adjacency matrix normalization
ω
degree
Learnable sharing confidence
Self-loop confidence
For each sub-graph
Cell-centered
Gene-centered
Gathering information
Cell-centered
Gene-centered
Fixed-dimension
New cell representation
New gene representation

- Lastly a linear classification layer produces classes $\widehat{y}_i = \text{softmax})(Wh_i^k + b)$.
- Cross entropy loss is used giving objective function $\widehat{\theta} = \text{argmin}_\theta - \sum_{c=1}^{C} y_c \log \widehat{y}_c$.
- To annotate the new dataset after training, new cells are connected to gene nodes of trained cell-gene graph and the gene expression of new cells are given as the edge between new cell node and gene node.
- The learned confidence parameters inform which genes are discriminatory for annotation.

# scGNN

- Methods like Phenograph, MAGIC, and Seurat, which use KNN graph to model relationship between cells, may over-simplify the complex cell and gene relationships of cell pop.
- Autoencoder methods give an alternative means to construct an effective representation of scRNA-Seq data via reconstructing its input.
- A unique feature of graph autoencoder is that they can learn a low-dimensional representation of the graph topology and train node relationships in a global view of the whole graph.


Preprocessing and signaling
Autoencoder-based hybrid iteration circle
Outputs
LTMG modeling
Regularizing
Cluster autoencoder
Reconstructed matrix
scGNN
Graph clustering
Converged
Cell clusters
Raw expression matrix
Pre-processed matrix
Feature autoencoder
Construct cell graph
Graph autoencoder
Cell graph
Regularizing
Imputed matrix
Imputation autoencoder

# Method

- Using scRNA-seq G.E. as input, data filtering and quality control are done as standard preprocessing - only genes expressed as non-zero in more than 1% of cells, and cellls expressed as non-zero in more than 1% of genes are kept. Then ranking by s.d., top 2000 genes kept. All data log-transformed.
- A mixed Gaussian model with left truncation is used to explore regulatory signals of g.e. The normalized expression values of gene $X$ over $N$ cells denoted $X = \{x_1, \ldots, x_N\}$, where $x_j \in X$ is assumed to follow mixture of $k$ Gaussians, corresponding to $k$ possible gene regulatory signals (TRSs).

- Density function of $X$ is

$$p(X;\Theta) = \prod_{j=1}^{N} p(x_j;\Theta) = \prod_{j=1}^{N}\sum_{i=1}^{k} \alpha_i p(x_j;\theta_i)$$

$$= \prod_{j=1}^{N}\sum_{i=1}^{k} \alpha_i \frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{-(x_j-\mu_i)^2}{2\sigma_i^2}} = L(\Theta;X)$$

  where $\alpha_i$ is mixing weight, and $\mu_i$ and $\sigma_i$ can be estimated by $\Theta^* = \text{argmax}_\Theta\, L(\Theta;X)$.

- With left truncation assumption, the gene expression profile is split into $M$, "truly measured expression of values", and $N - M$ left-censored g.e.'s for $N$ conditions. The number of Gaussian components is selected by the Bayesian Information Criterion, then original g.e. values labeled to most likely dist. under each cell.

- Then, the probability that $x_j$ belongs to distribution $i$ is given by

$$p(x_j \in \text{TRS}i | K, \Theta^*) \propto \frac{\alpha_i}{\sqrt{2\pi\sigma_i^2}} e^{\frac{-(x_j-\mu_i)^2}{2\sigma_i^2}}$$

and $x$ is assigned TRS $i$ if the probability is max of $i = 1, \ldots, K$.

# Feature Autoencoder

- The autoencoder learns the representative embedding of scRNA expression through stacked two layers of dense networks in both the encoder and decoder.
- Encoder gives low-dim embedding of $X'$ from input gene expression $X$, and decoder reconstructs expression $\widehat{X}$ from embedding, $X, \widehat{X} \in \mathbb{R}^{N \times M}$, $X' \in \mathbb{R}^{N \times M'}$..
- Objective of training is achieving max similarity between original and reconstructed through loss function MSE $\sim \sum (X - \widehat{X})^2$.

- Regularization is adopted to integrate gene regulation info during autoencoding process that is gene specific. The full loss is then defined as

$$\text{Loss} = (1-\alpha)\sum(X-\widehat{X})^2 + \alpha\sum\left((X-\widehat{X})^2 \circ \text{TRS}\right)$$

  where $\text{TRS} \in \mathbb{R}^{N\times M}$, $\alpha \in [0,1]$, $\circ$ is element-wise mult.
- In encoder, first and second layers have output dim of 512, 128 resp., and each layer is followed by ReLU. In decoder output is same reversed, with sigmoid activation function.
- The cluster autoencoder has same architecture as feature autoencoder but without regularization in loss.

# Cell Graph and Pruning

- The cell graph is created using KNN on the embedding learned by feature autoencoder.
- $K$ is predefined parameter used to control scale of interaction.
- Pruning process selects an adaptive number of neighbors for each node and keeps more biologically meaningful cell graph.
- Isolation Forest is applied to prune graph to detect outliers in $K$-neighbors of each node.

# Graph Autoencoder

- The graph autoencoder learns to embed and represent the topological information from the pruned cell graph.
- For input $G = (V, E)$, with $N = |V|$, and $A$ and $D$ as adjacency and degree matrices.
- The node feature matrix of the graph autoencoder is the learned embedding $X'$ from the feature autoencoder.
- Then the graph convolution network (GCN) is defined $\text{GCN}(X', A) = \text{ReLU}(\widetilde{A}X'W)$, where $W$ is weight matrix learned in training, and $\widetilde{A} = D^{-1/2}AD^{-1/2}$ symmetrically normalized adjacency matrix.
- The encoder of graph autoencoder is composed of two layers of GCN, giving $Z$

$$Z = \text{ReLU}(\widetilde{A}\text{ReLU}(\widetilde{A}X'W_1)W_2).$$

- The decoder of the graph autoencoder is defined as an inner product between the embedding:

$$\widehat{A} = \text{sigmoid}(ZZ^{\top})$$

where $\widehat{A}$ is reconstructed adjacency matrix, and $\text{sigmoid}(t) = 1/(1 + e^{-t})$.

- The goal of the graph autoencoder is to minimize the cross-entropy $L$ between $A$ and $\widehat{A}$:

$$L(A, \widehat{A}) = -\frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (a_{ij} * log(\widehat{a}_{ij}) + (1 - a_{ij}) * \log(1 - \widehat{a}_{ij}))$$

- After output from graph embedding is given, the $k$-means clustering method is used to cluster cells embedding, with number of clusters determined by Louvain algorithm. Then expression matrix on each cell cluster is reconstructed with cluster autoencoder.

- This collection of tasks goes through an iterative process aimed at building a single-cell graph that converges.

$$\widetilde{A} = \lambda L_0 + (1-\lambda)\frac{A_{ij}}{\sum_j A_{ij}}$$

where $L_0 = D_0^{-1/2} A_0 D_0^{-1/2}$ is the normalized adjacency matrix of the initial pruned graph, $\lambda \in [0, 1]$.

- If $\widetilde{A} - A < \gamma_1$ or if cell types are similar enough, that ARI $< \gamma_2$ then stop.
- $ARI = \frac{RI - \mathbb{E}[RI]}{\max(RI) - \mathbb{E}[RI]}$ for unadjusted rand index (RI) defined as

$$RI = \frac{a+b}{C_n^2}$$

where $a$ is number of pairs correctly labeled in same set, $b$ number of pairs correctly labeled as in different sets, $C_n^2$ total number of pairs. $\mathbb{E}[RI]$ expected RI of random labeling.

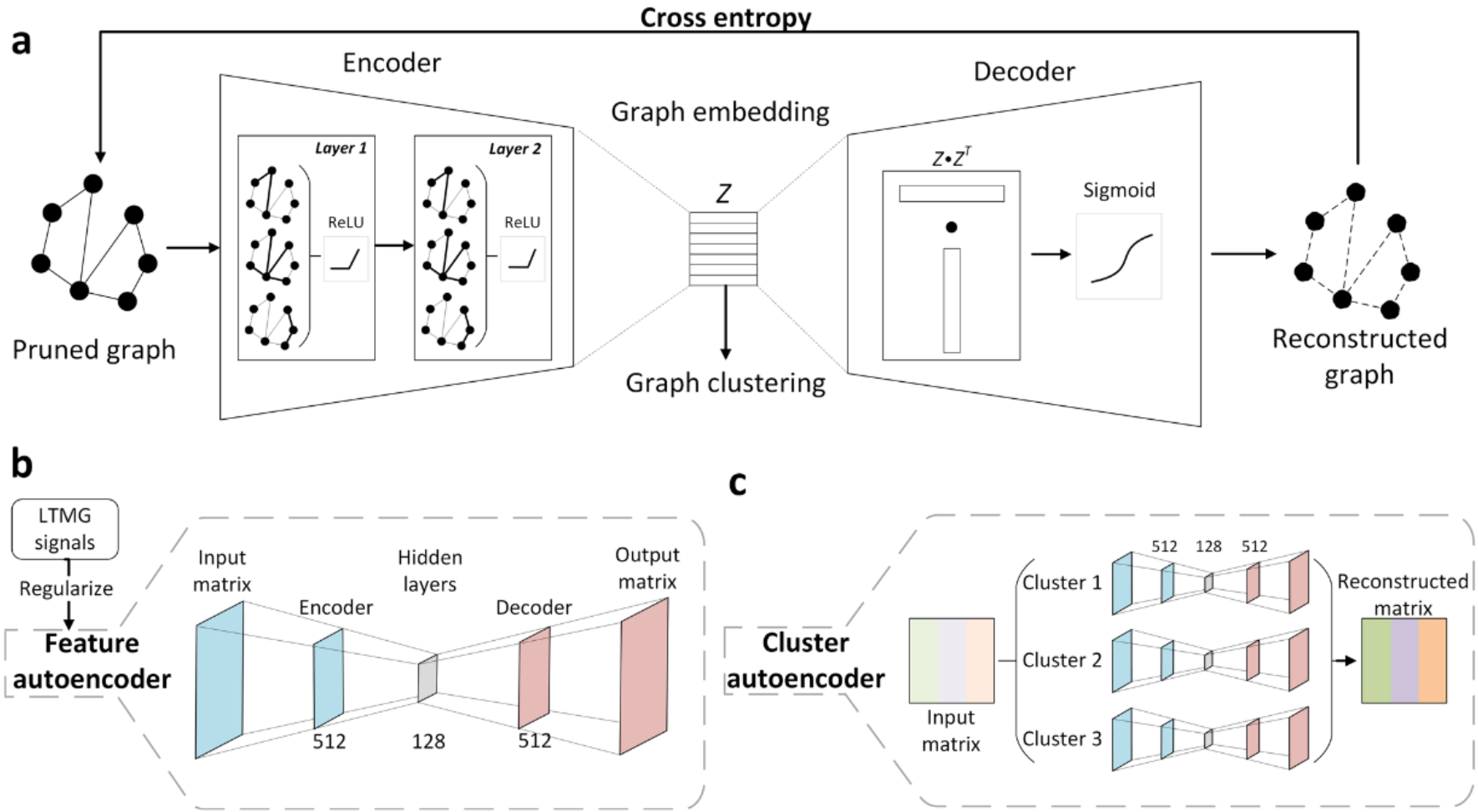
# Imputation Autoencoder

- After iterative process stops, the imputation AE imputes and denoises raw expression matrix within inferred cell-cell relationship.
- Has same arch. as feature AE, but also uses three additional regularizers from the cell graph.
- First reg is $\gamma_1 \sum(A \cdot (X - \widehat{X})^2)$ where $A \in \mathbb{R}^{N \times N}$ is the adjacency matrix from converged pruned cell graph, and $\cdot$ is matrix mult
- Cells within the same inferred cell type will be penalized via $\gamma_2 \sum(B \cdot (X - \widehat{X})^2)$ for $B_{ij} = 1$ if $i$ and $j$ are in same cell type, and 0 o/w.
- Lastly an $L_1$ regularizer $\beta \sum |w|$.

- Gives

$$\text{Loss} = (1-\alpha)\sum(X-\widehat{X})^2 + \alpha\sum\left((X-\widehat{X})^2 \circ \text{TRS}\right) + \beta\sum|w|$$

$$+\gamma_1\sum\left(A\cdot(X-\widehat{X})^2\right) + \gamma_2\sum\left(B\cdot(X-\widehat{X})^2\right).$$


a
Cross entropy
Encoder
Graph embedding
Decoder
Layer 1
Layer 2
ReLU
ReLU
Z
Z•Z^T
Sigmoid
Pruned graph
Graph clustering
Reconstructed graph
b
LTMG signals
Regularize
Feature autoencoder
Input matrix
Encoder
Hidden layers
Decoder
Output matrix
512
128
512
c
Cluster autoencoder
Input matrix
Cluster 1
Cluster 2
Cluster 3
512
128
512
Reconstructed matrix

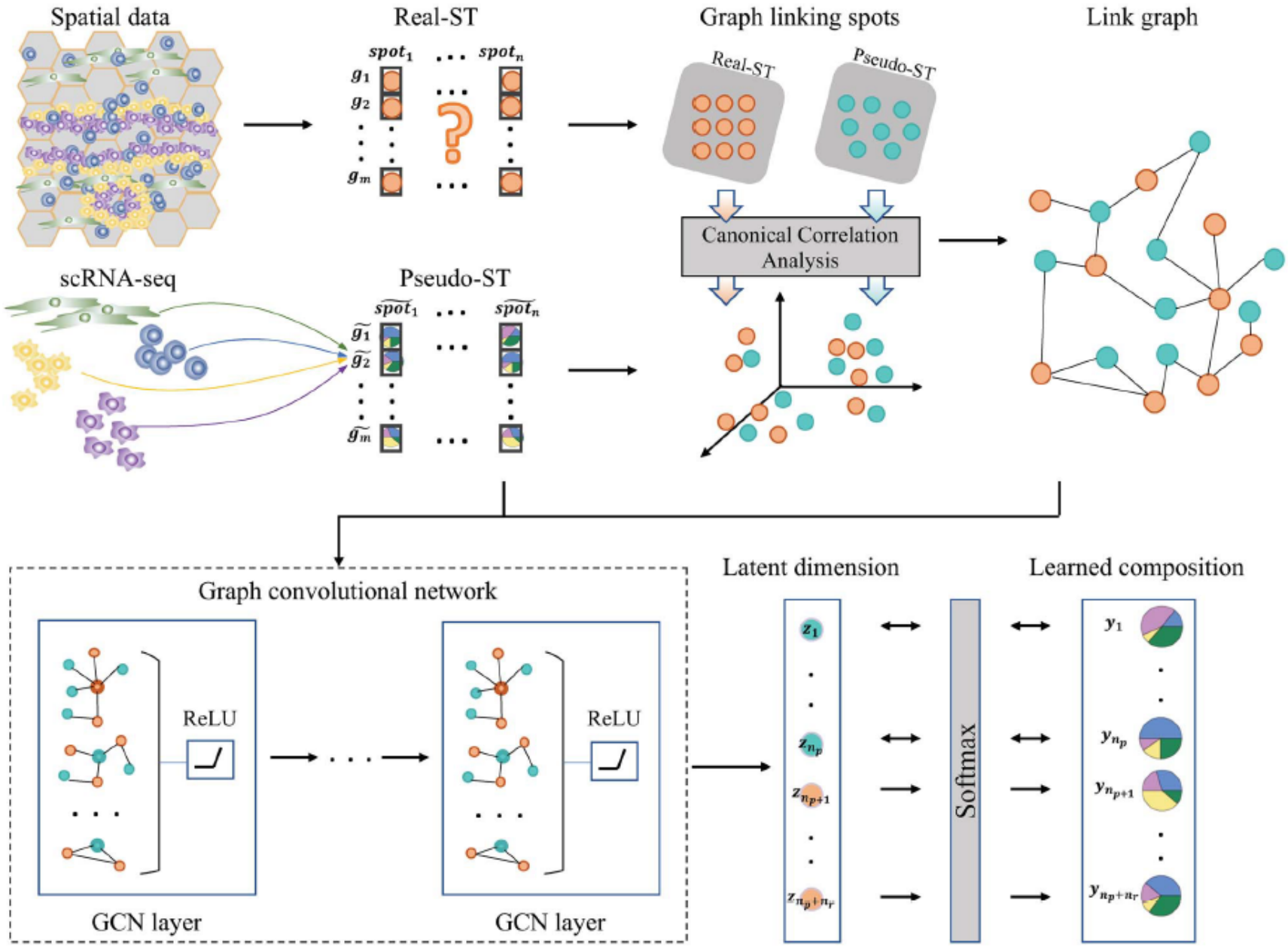
# Deconvolving Spatial Transcriptomics data via GNN

- Spatially localized cells can have complex spatial architecture in heterogeneous tissues.
- Understanding architecture yields insight into the cellular mechanisms and functions in diseases.
- scRNA-seq data alone cannot identify structural organization.
- Spatial transcriptomic (ST) profiling methods use spatially indexed barcodes with RNA-seq.
- However, resolution of ST data is lower than single-cell level data.
- 'Spot' sampling of cells (10-20 cells) usually have heterogeneous cell types.

- An identified goal is to identify cell compositions within each spot.
- Graph convolutional networks are proposed to capture topological information of cells.
- Authors develop the deconvoluting spatial transcriptomics data through graph-based convolutional netwroks (DSTG).
- DTSG is able to learn the precise composition of the ST data using semi-supervised GCN.

# Method

- Synthetic data produced from scRNA data after standard preprocessing is done (e.g. choosing top 2000 variable genes), then mixing cell expressions into 'spots'.
- This Pseudo-ST data is used alongside Real-ST data for deconvoluting.
- Then further preprocessing on ST data - normalization, log transformation, then GE normalization over spots.
- Link graph produced with spots as nodes - CCA to reduce dimension of Real-ST data and Pseudo-ST data.
- Mutual Nearest Neighbors used to determine links between Real-ST nodes and Pseudo-ST nodes.
- As well MNN's of Real-ST are found and used to generate links.


Spatial data
Real-ST
Graph linking spots
Link graph
Real-ST
Pseudo-ST
Canonical Correlation Analysis
scRNA-seq
Pseudo-ST
Graph convolutional network
ReLU
ReLU
GCN layer
GCN layer
Latent dimension
Softmax
Learned composition

- DSTG takes two inputs - link graph given by adj. matrix $A$ and $X = [X_{\text{pseudo}}\, X_{\text{real}}] \in \mathbb{R}^{m \times n}$, $n = n_p + n_r$ size of pseudo and real sets.
- Adjacency is normalized $\widetilde{A} = D^{-1/2}\widehat{A}D^{-1/2}$ for $\widehat{A} = A + I$.
- Each graph convolutional layer is defined

$$H^{(l+1)} = f\left(H^{(l)}, \widetilde{A}\right) = \text{ReLU}\left(\widetilde{A}H^{(l)}W^{(l)}\right)$$

- The three-layer DSTG uses soft-max for classification, giving

$$\widehat{Y} = f(X, \widetilde{A}) = \text{softmax}(\widetilde{A}\text{ReLU}(\widetilde{A}X^{\top}W^{(0)})W^{(1)})$$

for $\widehat{Y} = \begin{bmatrix} \widehat{Y}_p \\ \widehat{Y}_r \end{bmatrix} \in \mathbb{R}^{N \times F}$ for $F$ cell classes.

- The loss function is cross-entropy at pseudo-ST spots, $\mathcal{L} = -\sum_{i=1}^{n_p}\sum_{f=1}^{F} y_{i,f}\ln(\widehat{y}_{i,f})$.

# GraphSCI

- Existing imputation methods to solve GE dropout try to take similarity of cells and genes into account, but do not consider gene-to-gene or cell-to-cell correlations simultaneously, resulting in a lack of biological variation across cells or genes.
- As well, as imputation occurs, an iterative setup is possible to understand relations better between genes, which leads to better imputation.
- The Single-Cell Imputation method combines GCN and AE networks, called GraphSCI.

- A graph is constructed from the GE data with genes as nodes, and edges given by Pearson correlation coefficient (PCC) of (log-normalized) expression data $X$, so that if PCC is $> .3$ or $< -.3$, an edge is put between the nodes determing $A$.
- The GCN encodes the gene-to-gene network with GE matrix $X$ to latent vector $Z$, then reconstructs the edges.
- The AE encodes the GE matrix with gene-to-gene network and samples $Z$ from ZIMB or NB distributions to reconstruct GE matrix.
- Using $M$ cells and $N$ genes, let $\mathcal{N}$ and $\mathcal{M}$ be the set of genes and samples resp. then $\mathcal{G} = (\mathcal{N}, \mathcal{M}, E)$ for $E$ the edges, determined by $A \in \mathbb{R}^{N \times N}$.
- As usual, preprocessing is performed on $X \in \mathbb{R}^{N \times M}$, normalizing by cell library size then taking log-trans.
- $A_{ij} = \rho_{X_i, X_j} = \frac{\text{cov}(X_i, X_j)}{\sigma_{X_i} \sigma_{X_j}}$

- The GCN has an inference model $f_\phi$ and generative model $g_\varphi$ used to construct model for probabilistic encoder $q_\phi$ and prob. encoder $p_\varphi$.
- A two-layer GCN and two-layer fully connected NN maps $A$ and $X$ to low-d representation of posterior dist. of Gaussian or ZINB dist. respectively.
- The two layer GCN is defined $H_{\mathcal{N}}^{(1)} = \text{ReLU}(\widetilde{A}XW_{\mathcal{N}}^{(0)})$ and $[\mu_{\mathcal{N}}, \sigma_{\mathcal{N}}^2] = \widetilde{A}H_{\mathcal{N}}^{(1)}W_{\mathcal{N}}^{(1)}$ giving mean and variance of Gaussian.
- The two-layer encoder of AE for inferring ZINB dist of samples is $H_{\mathcal{M}}^{(1)} = \tanh(X^\top(W_{\mathcal{M}}^{(0)} \odot A) + b^{(0)})$, and $[\mu_{\mathcal{M}}, \theta_{\mathcal{M}}, \pi_{\mathcal{M}}] = \sigma(H_{\mathcal{M}}^{(1)}W_{\mathcal{M}}^{(1)} + b^{(1)})$ for mean, dispersion and dropout probability of ZINB.

$$ZINB(X|\mu;\theta;\pi) = \pi\delta_0(X) + (1-\pi)NB(X|\mu;\theta).$$

$$NB(X|\mu;\theta) = \frac{\Gamma(X+\theta)}{\Gamma(\theta)\Gamma(X+1)}\left(\frac{\theta}{\theta+mu}\right)^{\theta}\left(\frac{\theta}{\theta+mu}\right)^{X}.$$

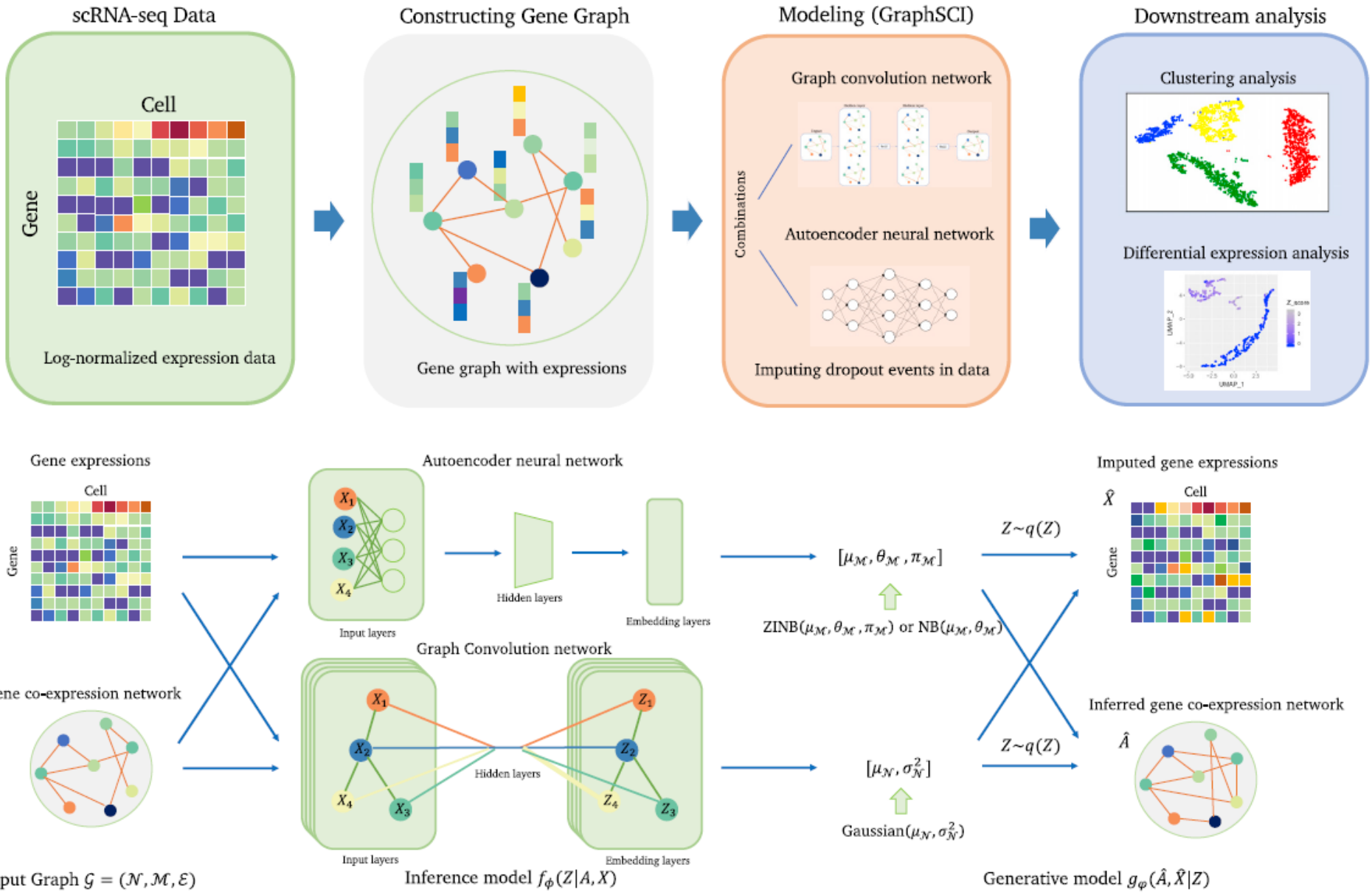
- The estimation of the ZINB parameters from the hidden layer are

$$\mu_{\mathcal{M}} = \exp(H^{(1)}_{\mathcal{M}} W^{(1)}_{\mathcal{M}} + b^{(1)})$$

$$\theta_{\mathcal{M}} = \text{softplus}(H^{(1)}_{\mathcal{M}} W^{(1)}_{\mathcal{M}} + b^{(1)})$$

$$\pi_{\mathcal{M}} = \text{sigmoid}(H^{(1)}_{\mathcal{M}} W^{(1)}_{\mathcal{M}} + b^{(1)})$$

- Then using latent space of $Z^{\mathcal{N}}$ and $Z^{\mathcal{M}}$, a decoder $g_\varphi$ is used to determine new parameters of cell distribution $[\mu'_{\mathcal{M}}, \theta'_{\mathcal{M}}, \pi'_{\mathcal{M}}] = g_{\varphi_1}(Z^{\mathcal{N}}_i, Z^{\mathcal{M}}_j)$ modeling $p_{\varphi_1}(\widehat{X}_{ij} | Z^{\mathcal{N}}_i, Z^{\mathcal{N}}_j) = ZINB(\mu'_{\mathcal{M}_{ij}}, \theta'_{\mathcal{M}_{ij}}, \pi'_{\mathcal{M}_{ij}})$.
- Then the implementation is $\widehat{X}_{ij} = g_{\varphi_1}(Z^{\mathcal{N}}_i, Z^{\mathcal{M}}_j) = diag(\vec{s}_j) \times Z^{\mathcal{M}}_j$ for $\vec{s}_j$ the size factor of cell $j$, and $\widehat{A}_{ij} = g_{\varphi_2}(Z^{\mathcal{N}}_i, Z^{\mathcal{N}}_j) = \text{sigmoid}((Z^{\mathcal{N}}_i)^\top Z^{\mathcal{N}}_j)$.


scRNA-seq Data
Cell
Gene
Log-normalized expression data
Constructing Gene Graph
Gene graph with expressions
Modeling (GraphSCI)
Graph convolution network
Combinations
Autoencoder neural network
Imputing dropout events in data
Downstream analysis
Clustering analysis
Differential expression analysis
Gene expressions
Cell
Gene
Autoencoder neural network
Input layers
Hidden layers
Embedding layers
Imputed gene expressions
$\hat{X}$
Cell
Gene
$[\mu_{\mathcal{M}}, \theta_{\mathcal{M}}, \pi_{\mathcal{M}}]$
$Z \sim q(Z)$
$\text{ZINB}(\mu_{\mathcal{M}}, \theta_{\mathcal{M}}, \pi_{\mathcal{M}})$ or $\text{NB}(\mu_{\mathcal{M}}, \theta_{\mathcal{M}})$
Gene co-expression network
Graph Convolution network
Hidden layers
Input layers
Embedding layers
Inferred gene co-expression network
$\hat{A}$
$[\mu_{\mathcal{N}}, \sigma^2_{\mathcal{N}}]$
$Z \sim q(Z)$
$\text{Gaussian}(\mu_{\mathcal{N}}, \sigma^2_{\mathcal{N}})$
Input Graph $\mathcal{G} = (\mathcal{N}, \mathcal{M}, \mathcal{E})$
Inference model $f_\phi(Z|A, X)$
Generative model $g_\varphi(\hat{A}, \hat{X}|Z)$

- The optimization is performed to obtain accurate embeddings of both genes and cells in unsupervised way.
- $Z^{\mathcal{N}}$ and $Z^{\mathcal{M}}$ are optimized by variational lower bound $\mathcal{L}$

$$\mathcal{L}(\phi, \varphi) := \mathbb{E}_{q_\phi}\left[\sum_{i \in \mathcal{N}, j \in \mathcal{M}} \log p_{\varphi_1}(\widehat{X}_{ij} | Z_i^{\mathcal{N}}, Z_j^{\mathcal{M}})\right]$$

$$+\mathbb{E}_{q_\phi}\left[\sum_{i,j \in \mathcal{N}} \log p_{\varphi_2}(\widehat{A}_{ij} | Z_i^{\mathcal{N}}, Z_j^{\mathcal{N}})\right]$$

$$-D_{KL}(q_\phi(Z^{\mathcal{M}}|A, X^\top)||p(Z^{\mathcal{M}})) - D_{KL}(q_\phi(Z^{\mathcal{N}}|A, X^\top)||p(Z^{\mathcal{N}}))$$

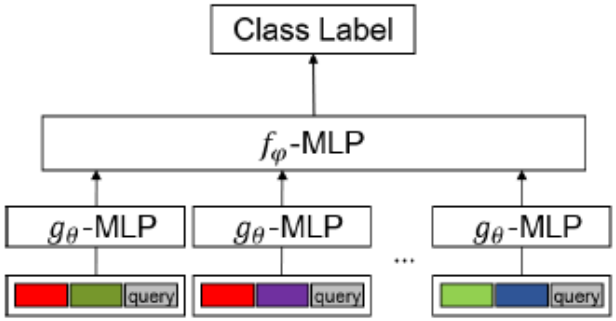
# GCN for Cancer Classification

- Breast cancer (BC) is a very common cancer, with many factors that effect it. A goal is to define subtypes based on phenotype, molecular behavior, and histopathology
- There exists some classifications of BC subtypes, but are suboptimal due to not investigating underlying mechanisms of cancer cells, namely the complex GE interdependencies.
- To address this, one should use biological networks.
- Graph based deep learning techniques allow non-grid data, so can leverage network biology.
- Authors develop a hybrid method of relational network (RN) and GCN.

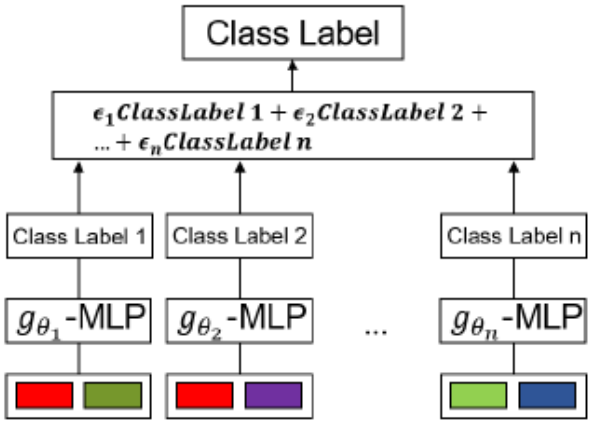
- The GCN uses spectral filters to transform the data, so that for graph Laplacian $L$ with eigendecomposition $L = U\Lambda U^\top$, the convolution of signals $x$ by $y$ is $Uy(\Lambda)U^\top x$ for $y(\Lambda)$ diagonal matrix.
- The Chebyshev polynomial is used in the Cheby-filter so that $y_{\theta'}(\Lambda) = \sum_{k=0}^{K-1} \theta'_k T_k(\widehat{\Lambda})$ for $\widehat{\Lambda} = 2\Lambda/\lambda_{max} - I_n$. Thus the filter is

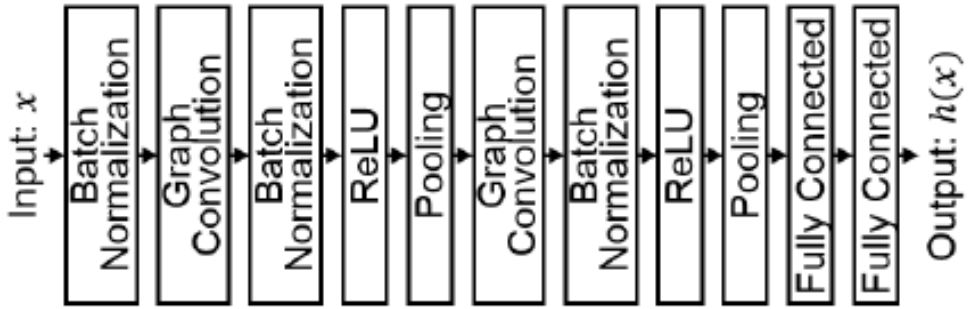
$$x *_G y_{\theta'} = \sum_{k=0}^{K-1} \theta'_k T_k(\widehat{L})x.$$

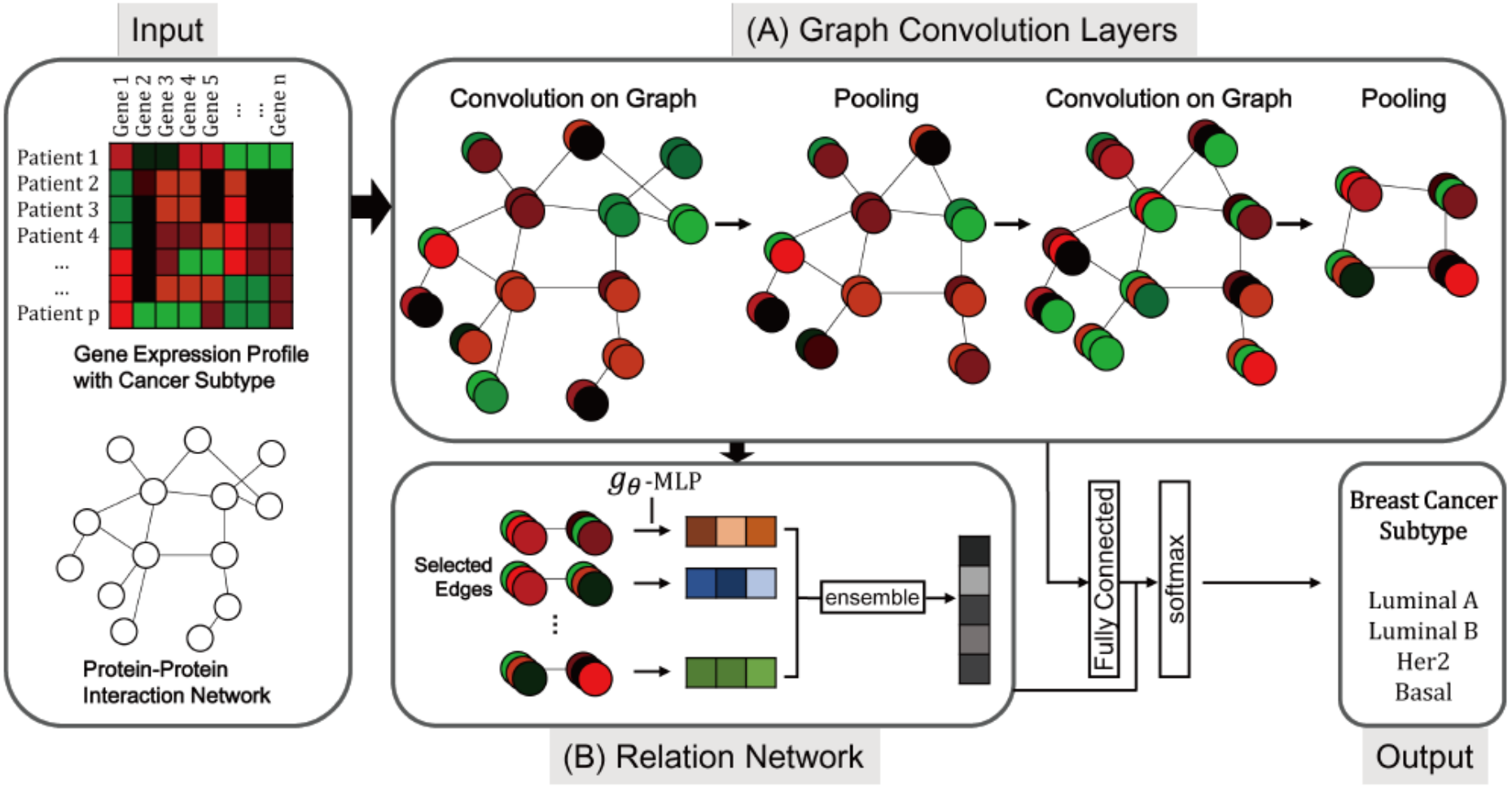
- The relation network is a NN that makes classifications of relations between objects $O = \{o_1, \ldots, o_n\}$, originally defined as $RN(O) = f_\psi(\sum_{i,j} g_\theta(o_i, o_j))$ for MLPs $f_\psi, g_\theta$.
- In original each pair had a query, but no query is available here, so is modified: top $k$ edges are selected via edge weights, and then these determine object pairs. And each pair will have separate parameters.
- Linear ensemble summation is used for $f$ function, yielding $RN(O) = \sum_{i,j} e_{ij} g_{\theta_{ij}}(o_i, o_j)$.




(a) Original RN




Input
Gene 1
Gene 2
Gene 3
Gene 4
Gene 5
...
...
Gene n
Patient 1
Patient 2
Patient 3
Patient 4
...
...
Patient p
Gene Expression Profile with Cancer Subtype
Protein-Protein Interaction Network
(A) Graph Convolution Layers
Convolution on Graph
Pooling
Convolution on Graph
Pooling
$g_\theta$-MLP
Selected Edges
ensemble
Fully Connected
softmax
Breast Cancer Subtype
Luminal A
Luminal B
Her2
Basal
(B) Relation Network
Output

- The final output of the model is $\widehat{y} = softmax(h(x_i) + \sum_{i,j} e_{ij} g_{\theta_{ij}}(o_i, o_j))$, where $h$ is the GCN.
- The output at the last graph convolution layer is also used as a input of the relation network.
- Cross-entropy between $\widehat{y}$ and true $y$ is loss function.
- Method was tested on synthetic and empirical datasets: synthetic data has two classes with means $\mu_1, \mu_2$ giving Multivariate Normal dists.
- Each class has underlying graph structure giving by covariance matrix $\Sigma$ with random entries $\sigma_{ij} \sim \mathcal{N}(0.1, 0.1)$ and then each $\sigma_{ij}$ is given random sign $\{-1, 1\}$.
- The empirical dataset had RNA-seq data from BC cells, and had classifications from PAM50 molecular subtype, and topology of graph determined from STRING database of protein-protein interaction network.

# DeepPaN

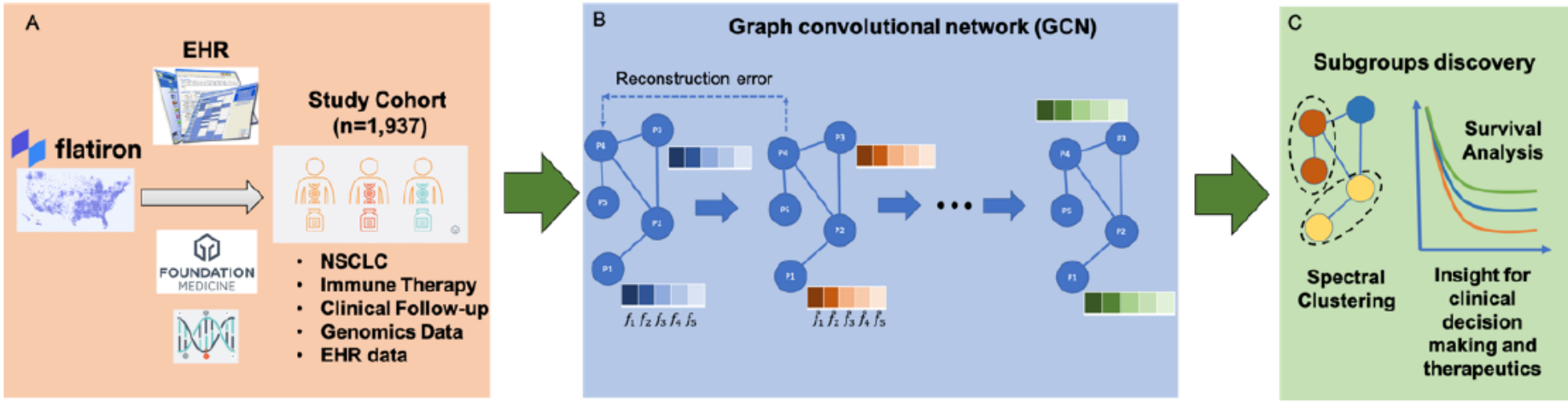
- Immuno-oncology (IO) therapies are used to treat non-small cell lung cancer (NSCLC).
- Responses to IO in NSCLC are highly variable.
- Recent findings suggest heterogeneous collection of genomic alterations and clinical phenotypes can change IO response.
- A goal is to find NSCLC subgroups (cancer types) across both clinical and genomic landscapes.
- Multi-modal data is available in the form of electronic health records (EHRs) and genomic data including tumor mutational burden (TMB).
- Patient similarity networks (PSNs) have shown promise for subtyping.
- DeePaN (deep patient graph convolutional network) is a unsupervised GNN for clustering NSCLC patients.

- A graph is created with patients as nodes - combined gene and EHR data, and links from KNN.
- After getting graph embedding from GAE, graph based spectral clustering is applied to discover patient subgroups with differential IO-treatment benefit.
- The reconstructed feature representation can be achieved by training an marginalized graph autoencoder (MGAE) on the patient network using the objective function

$$\frac{1}{m}\sum_{i=1}^{m}\|X-\widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}\widetilde{X}_i W\|^2+\lambda\|W\|_F^2$$

where $\widetilde{X}=[\widetilde{X}_1,\ldots,\widetilde{X}_m]$ are $m$ corrupted copies of $X\in\mathbb{R}^{n\times d}$, $\widetilde{A}=A+I$.

- The learned representation $Z_0$ of patients graph from MGAE is then put through spectral clustering after getting kernel matrix $Z_1=Z_0Z_0^\top$ and normalizing via $Z_2=\frac{1}{2}(|Z_1|+|Z_1^\top|)$.


A
EHR
flatiron
FOUNDATION MEDICINE
Study Cohort (n=1,937)
NSCLC
Immune Therapy
Clinical Follow-up
Genomics Data
EHR data
B
Graph convolutional network (GCN)
Reconstruction error
C
Subgroups discovery
Spectral Clustering
Survival Analysis
Insight for clinical decision making and therapeutics