Spatially contrastive variational autoencoder for deciphering tissue heterogeneity from spatially resolved transcriptomics

Yaofeng Hu[1], Kai Xiao[4,5], Hengyu Yang[1], Xiaoping Liu[1*], Chuanchao Zhang[1*], Qianqian Shi[2,3*]

[1]Key Laboratory of Systems Health Science of Zhejiang Province, School of Life Science, Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou, 310024, China

[2]Hubei Engineering Technology Research Center of Agricultural Big Data, Huazhong Agricultural University, Wuhan 430070, Hubei, China

[3]Hubei Key Laboratory of Agricultural Bioinformatics, College of Informatics, Huazhong Agricultural University, Wuhan 430070, China

[4]State Key Laboratory of Cell Biology, Shanghai Institute of Biochemistry and Cell Biology, Center for Excellence in Molecular Cell Science, Chinese Academy of Sciences, Shanghai 200031, China

[5]University of Chinese Academy of Sciences, Beijing 100049, China

* To whom correspondence should be addressed. Qianqian Shi. Email: qqshi@mail.hzau.edu.cn; Chuanchao Zhang, Email: chuanchaozhang@ucas.ac.cn; Xiaoping Liu, Email: xpliu@ucas.ac.cn

Section S1. Supplementary Notes:

1.1 Graph convolution and graph deconvolution

***Graph convolution*:** Convolutional operation in graph is a special form of Laplacian smoothing. Graph convolution on signal X^0 and X^1 with a filter g_c is defined as

$$H_0 = X^0 * g_c = X^0 U diag\big(g_c(\lambda_1), \dots, g_c(\lambda_N)\big)U^T, \quad (1)$$

$$H_1 = X^1 * g_c = X^1 U diag\big(g_c(\lambda_1), \dots, g_c(\lambda_N)\big)U^T, \quad (2)$$

where $\{\lambda_i\}_{i=1}^N$ and U represent the eigenvalues and eigenvectors of normalized Laplacian matrix $L = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = U\Lambda U^T$, respectively. Matrix A is the relationship of samples. D denotes the degree matrix. * denotes convolutional operator. We use the convolutional operator of GCN as the convolution operation of this model (i.e., $g_c(\lambda_i) = \lambda_i$).

***Graph deconvolution*:** As an inverse to convolution, graph deconvolution aims to recover the gene expression X^0 and X^1 from the smoothed representation H_0 and H_1. From the spectral perspective, graph deconvolution on a smoothed representation H_0 and H_1 with filter g_d is defined as:

$$\widehat{X^0} = H_0 * g_d = H_0 U diag\big(g_d(\lambda_1), \dots, g_d(\lambda_N)\big)U^T, \quad (3)$$

$$\widehat{X^1} = H_1 * g_d = H_1 U diag\big(g_d(\lambda_1), \dots, g_d(\lambda_N)\big)U^T, \quad (4)$$

In general, the deconvolutional operator g_d is the inverse function of g_c, e.g., $g_d(\lambda_i) = 1/\lambda_i$. Due to the randomness of the latent representation in VAE, the general deconvolutional operators are not suitable for this model.

Combining with the graph convolution, the representation H_0 and H_1 with stochastic disturbance are similarly defined as:

$$H_0 = X^0 U g_c(\Lambda)U^T + \epsilon_0, \quad (5)$$

$$H_1 = X^1 U g_c(\Lambda)U^T + \epsilon_1, \quad (6)$$

where ϵ_0 and ϵ_1 are stochastic disturbance (i.e., $E((\epsilon_0)_i) = 0,\ E((\epsilon_1)_i) = 0;\ VAR((\epsilon_0)_i) = E\big((\epsilon_0)_i{}^2\big) = \sigma_0^2, VAR((\epsilon_1)_i) = E\big((\epsilon_1)_i{}^2\big) = \sigma_1^2$) from the representation of VAE.

Naturally, the gene expression data recovered by graph deconvolution are formulated by

$$\widetilde{X^0} = X^0 U g_d(\Lambda) g_c(\Lambda)U^T + \epsilon_0 U g_d(\Lambda)U^T, \quad (7)$$

$$\widetilde{X^1} = X^1 U g_d(\Lambda) g_c(\Lambda)U^T + \epsilon_1 U g_d(\Lambda)U^T. \quad (8)$$

The reconstructed error is defined as

$$\begin{aligned} MSE &= E\left\|X^0 - \widetilde{X^0}\right\|_2^2 + E\left\|X^1 - \widetilde{X^1}\right\|_2^2 = E\left\|X^0 U - \widetilde{X^0} U\right\|_2^2 + E\left\|X^1 U - \widetilde{X^1} U\right\|_2^2 \\ &= \sum_{i=1}^{N} \big(g_d(\lambda_i) g_c(\lambda_i) - 1\big)^2 E\left[(X^0 U)_i{}^2\right] + g_d^2(\lambda_i)\, E\left[(\epsilon_0 U)_i{}^2\right] \\ &+ \sum_{i=1}^{N} \big(g_d(\lambda_i) g_c(\lambda_i) - 1\big)^2 E\left[(X^1 U)_i{}^2\right] + g_d^2(\lambda_i)\, E\left[(\epsilon_1 U)_i{}^2\right]. \end{aligned} \quad (9)$$

Because of the existence of random perturbation terms (i.e., $\epsilon_0,\ \epsilon_1$), the inverse function of convolution as deconvolution (i.e., $g_d(\lambda_i) g_c(\lambda_i) = 1$) cannot minimize the reconstruction errors. Considering the convexity of Eq. 9, MSE is minimized by setting the derivative with respect to $g_d(\lambda_i)$ to zero and thus we obtain the graph deconvolution filter $g_d(\lambda_i)$ as

$$g_d(\lambda_i) = \frac{g_c(\lambda_i)}{g_c^2(\lambda_i) + \dfrac{\sigma_0^2 + \sigma_1^2}{E\left[(X^0 U)_i{}^2\right] + E\left[(X^1 U)_i{}^2\right]}} = \frac{g_c(\lambda_i)}{g_c^2(\lambda_i) + \dfrac{\sigma_0^2 + \sigma_1^2}{E\left[(X^0)_i{}^2\right] + E\left[(X^1)_i{}^2\right]}}, \quad (10)$$

where $\sigma_0^2 = VAR((\epsilon_0)_i) = E\left((\epsilon_0)_i{}^2\right) = E\left((\epsilon_0 U)_i{}^2\right), \sigma_1^2 = VAR((\epsilon_1)_i) = E\left((\epsilon_1)_i{}^2\right) = E\left((\epsilon_1 U)_i{}^2\right)$. The average values (i.e., $\frac{1}{N}\sum_{i=1}^{N} E\left[(X^0)_i{}^2\right]$ and $\frac{1}{N}\sum_{i=1}^{N} E\left[(X^1)_i{}^2\right]$) are used to approximate the values of $E\left[(X^0)_i{}^2\right]$ and $E\left[(X^1)_i{}^2\right]$, respectively. Due to the following statistical formula

$$\sum_{i=1}^{N} E\left[(X^0)_i{}^2\right] = \sum_{i=1}^{N} E[(X^0)_i]^2 + VAR[(X^0)_i], \tag{9}$$

$$\sum_{i=1}^{N} E\left[(X^1)_i{}^2\right] = \sum_{i=1}^{N} E[(X^1)_i]^2 + VAR[(X^1)_i], \tag{10}$$

The values of $E\left[(X^0)_i{}^2\right]$ and $E\left[(X^1)_i{}^2\right]$ are estimated as follows:

$$E\left[(X^0)_i{}^2\right] = \frac{1}{ND'}\left(\|X^0\|_F^2 + \left\|X^0 - \frac{1}{N}X^0\mathbf{1}\right\|_F^2\right), \tag{11}$$

$$E\left[(X^1)_i{}^2\right] = \frac{1}{ND'}\left(\|X^1\|_F^2 + \left\|X^1 - \frac{1}{N}X^1\mathbf{1}\right\|_F^2\right), \tag{12}$$

where $\mathbf{1} \in \mathrm{R}^{\mathrm{N}\times\mathrm{N}}$ is all ones matrix and D' is the feature size of the signal X^0 and X^1. The variance σ_0^2, σ_1^2 are estimated by considering their neighborhoods as

$$\sigma_0^2 = \frac{1}{ND'}\|X^0 - D^{-1}AX^0\|_F^2, \tag{13}$$

$$\sigma_1^2 = \frac{1}{ND'}\|X^1 - D^{-1}AX^1\|_F^2. \tag{14}$$

1.2 Graph embedding contrastive variational autoencoder: We build a graph embedding contrastive variational autoencoder to contrast the biological signals of each spot and its spatial neighbors for learning representation. There are three principal components: graph convolutional encoder, graph deconvolutional decoder and deep contrastive variational autoencoder. The main procedures can be stated as follows.

- *Graph convolutional encoder*: We use graph convolutional network (GCN) as the encoder to learn representation with spatial constraint from the original and augmented expression data (i.e., X_0 and X_1). In general, graph convolutional operation can be interpreted as a special form of Laplacian smoothing [17]. From the spectral perspective, the t-th layer of the graph convolutional encoder for X_0 and X_1, denoted as $H_0^{(t)}$ and $H_1^{(t)}$, is defined as:

$$H_0^{(t)} = \phi_0^{(t)}\left(W_0^{(t)}H_0^{(t-1)} * g_c(L) + b_0^{(t)}\right)$$
$$= \phi_0^{(t)}\left(W_0^{(t)}H_0^{(t-1)}Udiag\left(g_c(\lambda_1), \dots, g_c(\lambda_N)\right)U^T + b_0^{(t)}\right), t = 1, \dots, J, \tag{15}$$

$$H_1^{(t)} = \phi_1^{(t)}\left(W_1^{(t)}H_1^{(t-1)} * g_c(L) + b_1^{(t)}\right)$$
$$= \phi_1^{(t)}\left(W_1^{(t)}H_1^{(t-1)}Udiag\left(g_c(\lambda_1), \dots, g_c(\lambda_N)\right)U^T + b_1^{(t)}\right), t = 1, \dots, J, \tag{16}$$

where $L = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = U\Lambda U^T = Udiag(\lambda_1, \dots, \lambda_N)U^T$ is the normalized Laplacian matrix, and $\{\lambda_i\}_{i=1}^{N}$ and U are the eigenvalues and eigenvectors of matrix L. D denotes the

degree matrix. $\phi_0^{(t)}$ and $\phi_1^{(t)}$ are the activation functions of the t-th GCN encoder layer for X_0 and X_1, respectively. J is the number of encoder layers. * denotes convolutional operator. We use the convolutional operator of GCN as the convolution operation of this model (i.e., $g_c(\lambda_i) = \lambda_i$). For convenience, we denote the gene expression matrix X_0 and X_1 as $H_0^{(0)}$ and $H_1^{(0)}$, respectively.

- *Graph deconvolutional decoder*: Although graph convolution can effectively learn representation with local spatial information, the resulting smoothness affects the expression data reconstruction and weakens the global information of the learnable representation from the gene expression data. To solve this problem, we propose graph deconvolutional network (GDN) as the decoder to attenuate the effect of graph convolution, enabling efficient representation to be learned from the expression. As an inverse to graph convolution, the k-th layer ($k = J+1, \dots, L$, L is the number of GC-VAE layers) of the graph deconvolution decoder for X_0 and X_1, denoted as $H_0^{(k)}$ and $H_1^{(k)}$, is defined as:

$$H_0^{(k)} = \psi_0^{(k)}\left(W_0^{(k)} H_0^{(k-1)} U diag\left(g_d(\lambda_1), \dots, g_d(\lambda_N)\right) U^T + b_0^{(k)}\right), k = J+1, \dots, L, \quad (17)$$

$$H_1^{(k)} = \psi_1^{(k)}\left(W_1^{(k)} H_1^{(k-1)} U diag\left(g_d(\lambda_1), \dots, g_d(\lambda_N)\right) U^T + b_1^{(k)}\right), k = J+1, \dots, L, \quad (18)$$

where $\psi_0^{(k)}$ and $\psi_1^{(k)}$ are the activation functions of the k-th GDN decoder layer for X_0 and X_1, respectively. $g_d(\lambda_i)$ is the deconvolutional operator of GDN.

In general, the deconvolutional operator g_d is the inverse function of g_c, e.g., $g_d(\lambda_i) = 1/\lambda_i$. Due to the randomness of the latent representation in GC-VAE, the general deconvolutional operator are not suitable for this model (Supplementary Note S1). Therefore, we adopt the following new deconvolution operators for the k-th layer decoder $H_0^{(k)}$ and $H_1^{(k)}$:

$$g_d^{(k)}(\lambda_i) = \frac{g_c^{(k)}(\lambda_i)}{g_c^{(k)}(\lambda_i)^2 + \frac{{\sigma_0^{(k)}}^2 + {\sigma_1^{(k)}}^2}{E\left[\left(H_0^{(k)}\right)_i^2\right] + E\left[\left(H_1^{(k)}\right)_i^2\right]}}, k = J+1, \dots, L, \quad (19)$$

where ${\sigma_0^{(k)}}^2$ and ${\sigma_1^{(k)}}^2$ are the variance of the stochastic disturbance in $\left(H_0^{(k)}\right)_i$ and $\left(H_1^{(k)}\right)_i$, and $\mathrm{E}[*]$ denotes mathematical expectation, respectively. The average values (i.e., $\frac{1}{N}\sum_{i=1}^{N} E\left[\left(H_0^{(k)}\right)_i^2\right]$ and $\frac{1}{N}\sum_{i=1}^{N} E\left[\left(H_1^{(k)}\right)_i^2\right]$) are used to approximate the value of $E\left[\left(H_0^{(k)}\right)_i^2\right]$ and $E\left[\left(H_1^{(k)}\right)_i^2\right]$, respectively. Due to the following statistical formula

$$\sum_{i=1}^{N} E\left[\left(H_0^{(k)}\right)_i^2\right] = \sum_{i=1}^{N} E\left[\left(H_0^{(k)}\right)_i\right]^2 + VAR\left[\left(H_0^{(k)}\right)_i\right], \quad (20)$$

$$\sum_{i=1}^{N} E\left[\left(H_1^{(k)}\right)_i^2\right] = \sum_{i=1}^{N} E\left[\left(H_1^{(k)}\right)_i\right]^2 + VAR\left[\left(H_1^{(k)}\right)_i\right]. \quad (21)$$

The values of $E\left[\left(H_0^{(k)}\right)_i^2\right]$ and $E\left[\left(H_1^{(k)}\right)_i^2\right]$ are estimated as follows:

$$E\left[\left(H_0^{(k)}\right)_i^2\right] = \frac{1}{ND'}\left(\left\|H_0^{(k)}\right\|_F^2 + \left\|H_0^{(k)} - \frac{1}{N} H_0^{(k)} \mathbf{1}\right\|_F^2\right), \quad (22)$$

$$E\left[\left(H_1^{(k)}\right)_i^2\right] = \frac{1}{ND'}\left(\left\|H_1^{(k)}\right\|_F^2 + \left\|H_1^{(k)} - \frac{1}{N} H_1^{(k)} \mathbf{1}\right\|_F^2\right), \quad (23)$$

where $\mathbf{1} \in \mathrm{R}^{\mathrm{N}\times N}$ is the all ones matrix and D' is the feature size of the k-th layer of decoder. The variance ${\sigma_0^{(k)}}^2$, ${\sigma_1^{(k)}}^2$ are estimated by considering their neighborhoods as

$$ {\sigma_0^{(k)}}^2 = \frac{1}{ND'} \left\| H_0^{(k)} - D^{-1} A H_0^{(k)} \right\|_F^2, \tag{24} $$

$$ {\sigma_1^{(k)}}^2 = \frac{1}{ND'} \left\| H_1^{(k)} - D^{-1} A H_1^{(k)} \right\|_F^2. \tag{25} $$

- *Deep contrastive variational autoencoder*: To balance spatial local information and global information from the original and augmented expression data, we present a deep contrastive strategy as the soft constraint during representation learning. The contrastive loss can be expressed as:

$$ \mathcal{L}_{con} = \sum_{l=1}^{L} \left\| H_0^{(l)} - H_1^{(l)} \right\|_F^2. \tag{26} $$

Combined with the loss (i.e., $\mathcal{L}_{ELBO}^0$ and $\mathcal{L}_{ELBO}^1$) of variational autoencoder (VAE) for the original data X_0 and the augmented expression data X_1, the overall loss function of SpaCAE is denoted as follows:

$$ \mathcal{L}_{ELBO}^0 = -E_{q\left(H_0^{(J)}|X_0\right)}\left(logp\left(X_0|H_0^{(J)}\right)\right) + KL\left(q\left(H_0^{(J)}|X_0\right)\middle\|p\left(H_0^{(J)}\right)\right), \tag{27} $$

$$ \mathcal{L}_{ELBO}^1 = -E_{q\left(H_1^{(J)}|X_1\right)}\left(logp\left(X_1|H_1^{(J)}\right)\right) + KL\left(q\left(H_1^{(J)}|X_1\right)\middle\|p\left(H_1^{(J)}\right)\right), \tag{28} $$

$$ \mathcal{L} = \mathcal{L}_{ELBO}^0 + \mathcal{L}_{ELBO}^1 + \lambda \mathcal{L}_{con}, \tag{29} $$

where the tunable parameter λ can be manually set and defaults to 1.

1.3 Comparison with baseline methods

To showcase the effectiveness of SpaCAE in spatial clustering, we compared SpaCAE with the state-of-the-art methods, including the SpaGCN, BayesSpace, GraphST and STAGATE. All methods were updated to the latest stable version and the detailed implementation for each method is listed as follows:

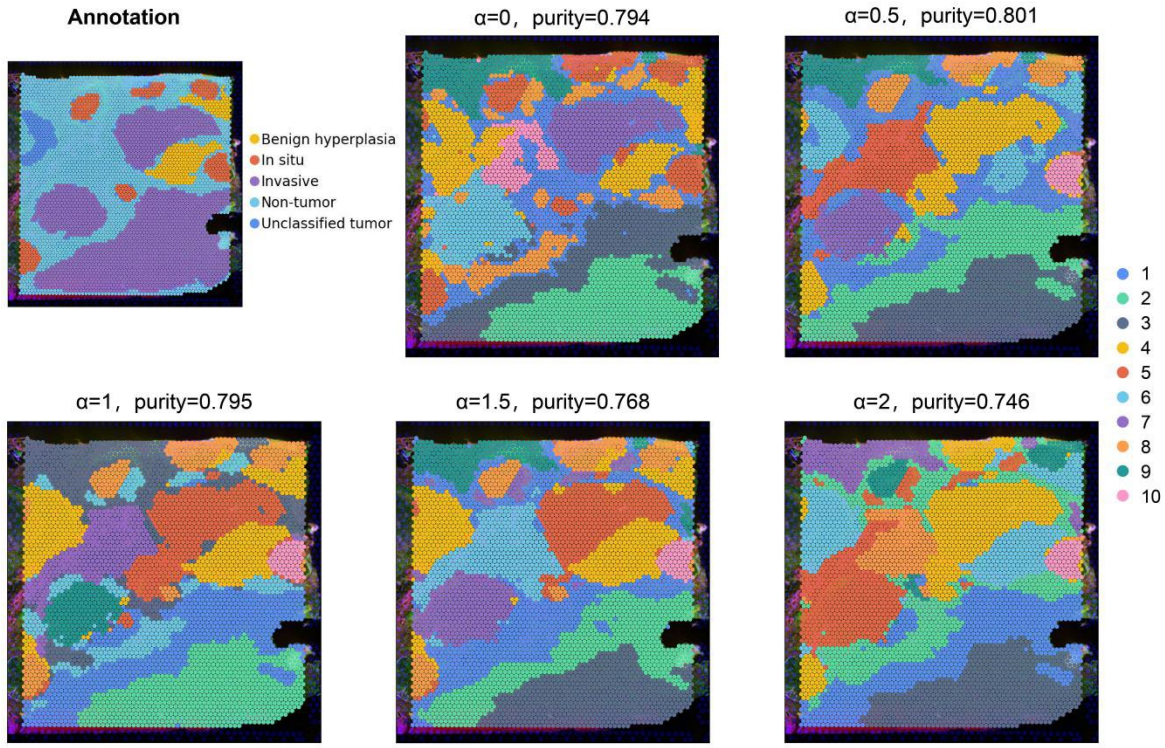
- BayesSpace: We followed the workflow specified in the examples (BayesSpace analysis of DLPFC dataset) of BayesSpace documentation website (https://edward130603.github.io/BayesSpace/articles/maynard_DLPFC.html). Firstly, we created a SingleCellExperiment object using readVisium() function, and performed preprocessing using spatialPreprocess() function. Then, we performed spatial domain identification using spatialCluster() function, whose parameter q is set to the number of annotated labels for each dataset.
- SpaGCN: We followed the workflow specified in the tutorial of the SpaGCN GitHub repository (https://github.com/jianhuupenn/SpaGCN/blob/master/tutorial/tutorial.md). Firstly, we created an AnnData object using raw counts, calculated adjacent matrix, and performed preprocessing on expression data using spg.calculate_adj_matrix(), spg.prefilter_genes(), spg.prefilter_specialgenes(), sc.pp.normalize_per_cell() and sc.pp.log1p() respectively. Then, we ran spg.search_l() and spg.search_res() to set hyper-parameters. Finally, we trained SpaGCN model and used it to identify spatial domains, which was implemented in

clf=spg.SpaGCN(), clf.train() and clf.predict(). We ran these functions with default parameters, and the parameter target_num was set to the number of annotated labels for each dataset.

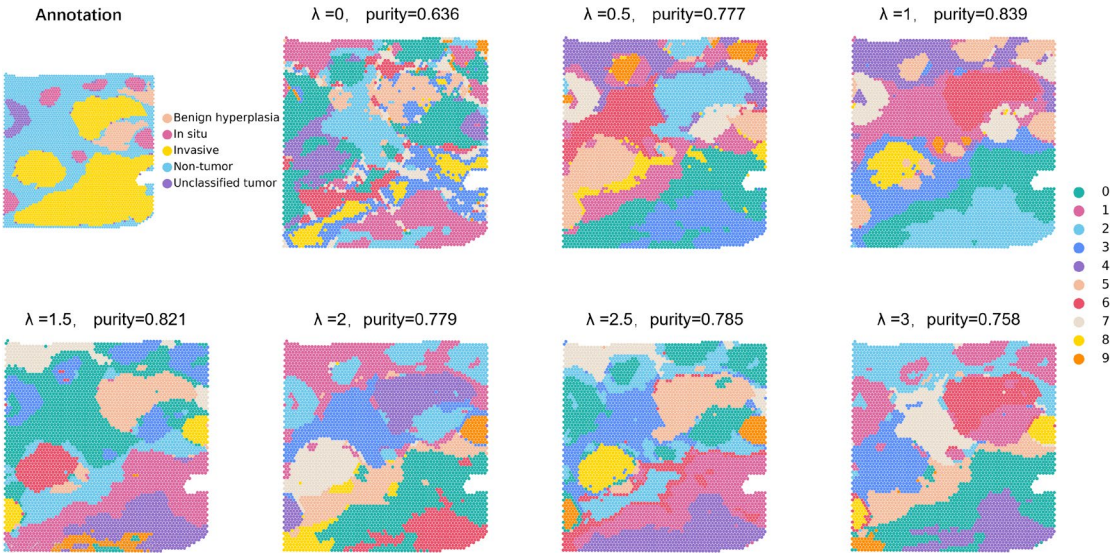
- STAGATE: We followed the workflow specified in the tutorials (Tutorial 1: 10x Visium (DLPFC dataset)) of STAGATE documentation website (https://stagate.readthedocs.io). Firstly, we created an AnnData object using sc.read_visium() function SCANPY package. Then, this AnnData object was normalized and log-transformed using sc.pp.normalize_total() and sc.pp.log1p() SCANPY package. After preprocessing, we sequentially constructed the spatial network and ran STAGATE using STAGATE.Cal_Spatial_Net() and STAGATE.train_STAGATE() respectively. Finally, we computed neighbor graph on the latent representation, identified spatial domains and performed UMAP visualization, which were implemented in sc.pp.neighbors(), sc.pp.leiden() and sc.tl.umap() from SCANPY package respectively. We ran these functions with default parameters, and the resolution was adjusted to match the number of annotated labels for each dataset.
- GraphST: We followed the workflow specified in the tutorial (Tutorial 1: 10X Visium) of GraphST documentation website (https://deepst-tutorials.readthedocs.io). Firstly, we created an AnnData object using sc.read_visium() function SCANPY package. Then, we defined and trained the GraphST model using model=GraphST.GraphST() and model.train() respectively. Finally, we identified spatial domains using clustering() function from GraphST.utils module. We ran these functions with default parameters, and the number of clusters was adjusted to match the number of annotated labels for each dataset.

Section S2. Supplementary Figures and Tables

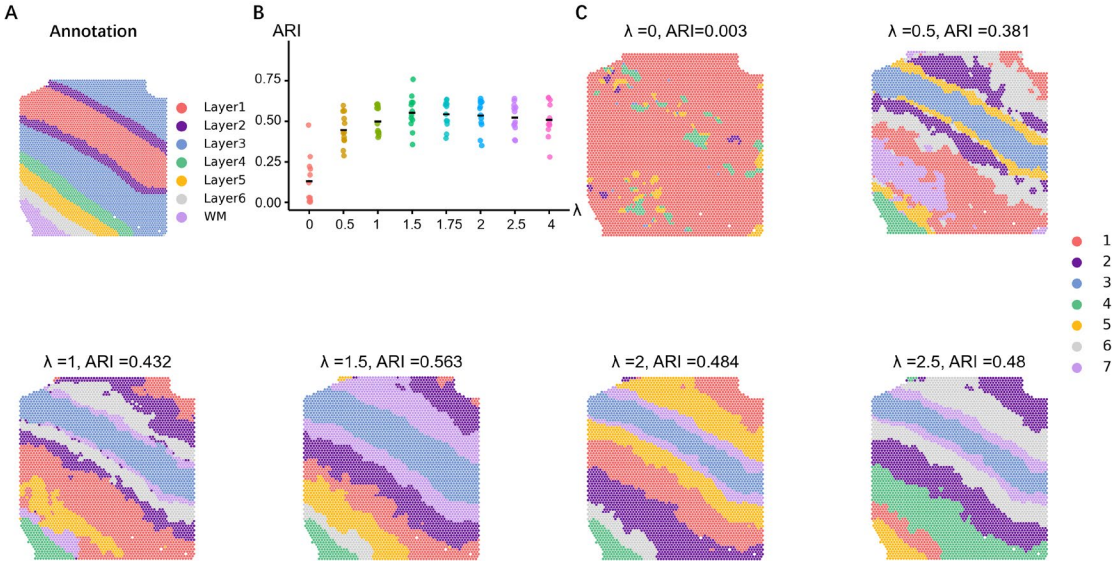
Supplementary Figure 1. The spatial domain identification on invasive ductal carcinoma (IDC) slice to investigate the influence of hyper-parameter α. Cluster purity (i.e., purity) is used to evaluate the clustering performance. The purity increases as the value α increases from 0 to 0.5 and achieve the best performance when α=0.5 on SpaCAE. Nevertheless, the purity decreases as the α goes beyond the optimal value. In summary, the purity indicates a trend of increasing first and then decreasing as hyper-parameter α increases. Specifically, from the in-situ staining images, it can be observed that the clustering results becomes smoother with the increase of α. Obviously, the hyper-parameter α plays a significant role in combining the node features and those of its neighbors with the graph.



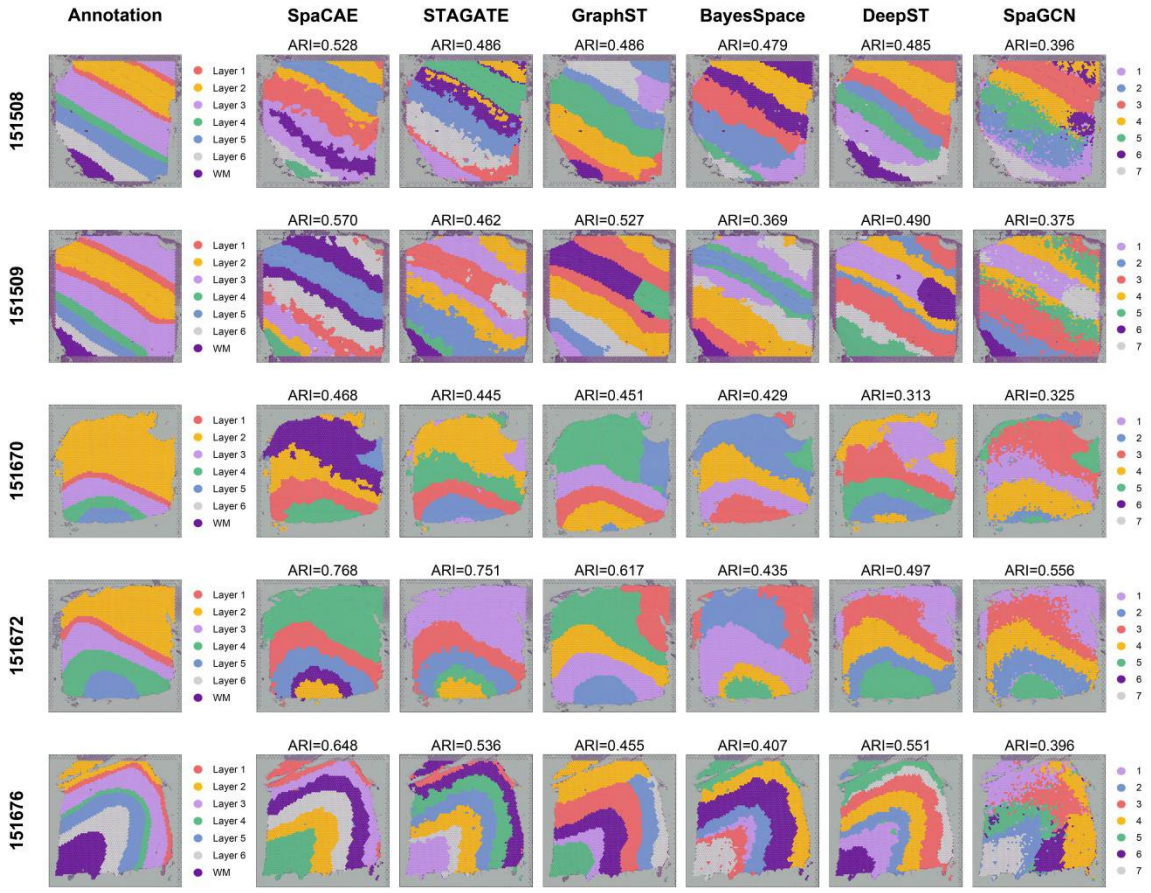
Supplementary Figure 2. The domain identification performance on Invasive Ductal Carcinoma (IDC) slice to investigate the influence of hyper-parameter λ with α fixed to 0.5. Cluster purity (i.e., purity) is used to evaluate the clustering performance. It is clear that the purity increases as the value λ increases from 0 to 1 and SpaCAE performs well with λ less than 2.



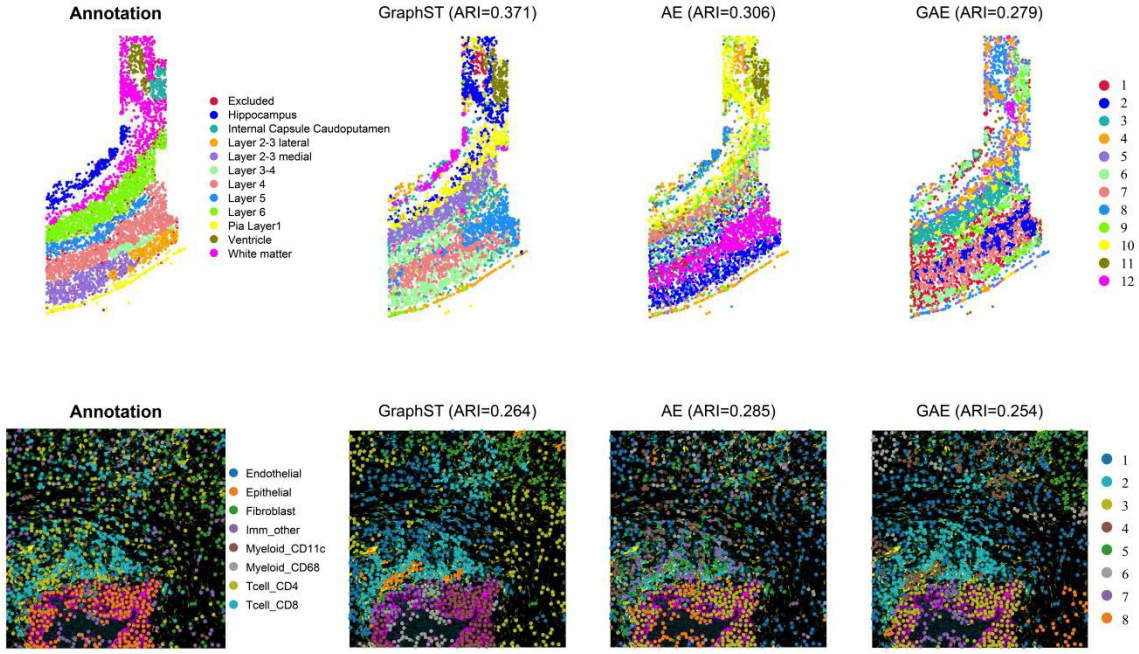
Supplementary Figure 3. The spatial domain identification on the DLPFC dataset to investigate the influence of hyper-parameter λ with α set to 0.3. (A) The ground truth of slide 151509 (n=4,789 spots) slice in DLPFC datasets. (B) Boxplots of the performance of SpaCAE on 12 slices with λ varies from 0 to 4. SpaCAE' s performance demonstrates a trend of initial improvement followed by a decline, achieving the best performance at λ=1.5 (ARI=0.554±0.106). (C) Identification of spatial domains by SpaCAE with different λ.



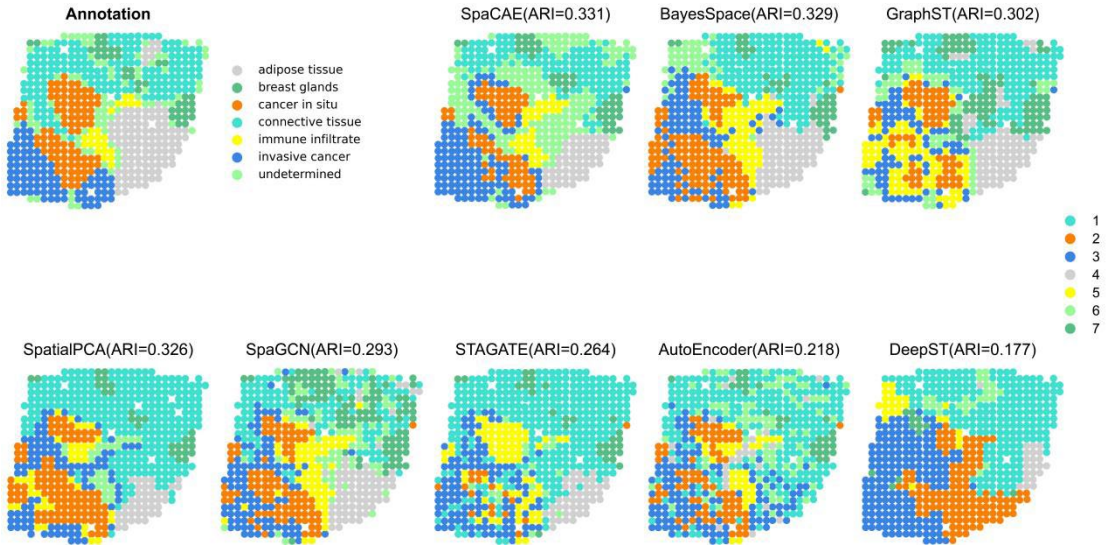
Supplementary Figure 4. Comparison of spatial domains by clustering assignments via SpaCAE, STAGATE, GraphST, BayesSpace, DeepST, SpaGCN, and manual annotation in the representative slices of the DLPFC dataset.



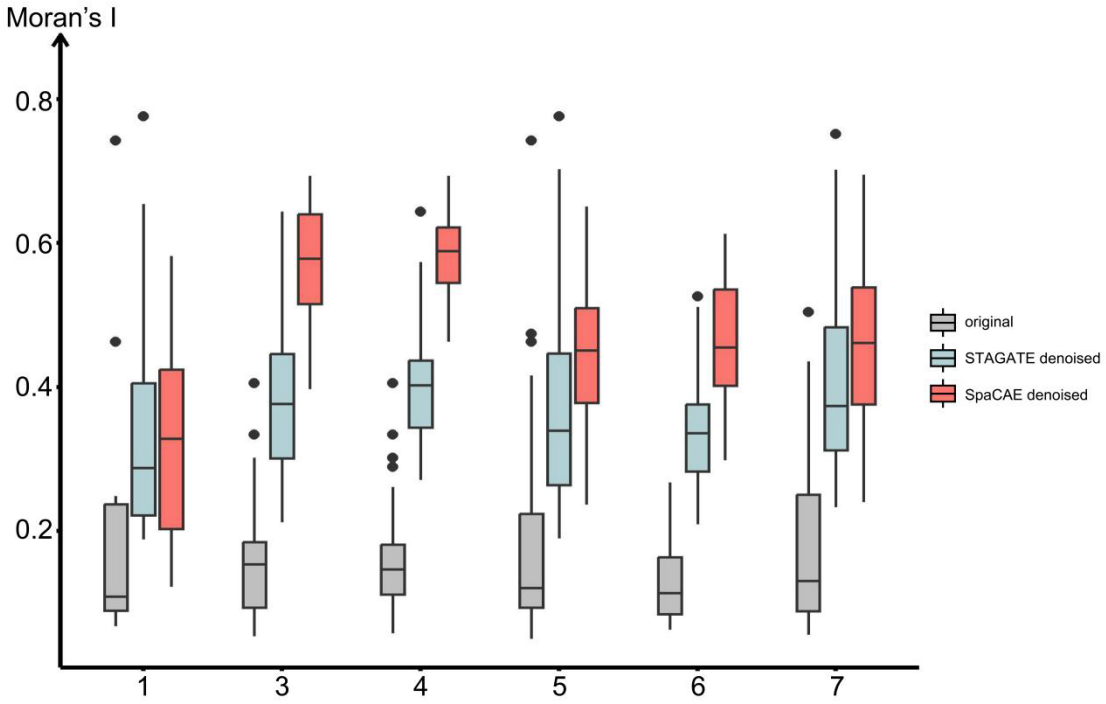
Supplementary Figure 5. Comparison of spatial domains by clustering assignments via GraphST, 3-layer AutoEncoder(with mclust) and GAE(with leiden) on SpaCAE-denoised spatial Mouse somatosensory cortex and imaging-based molecular MIBITOF data respectively.



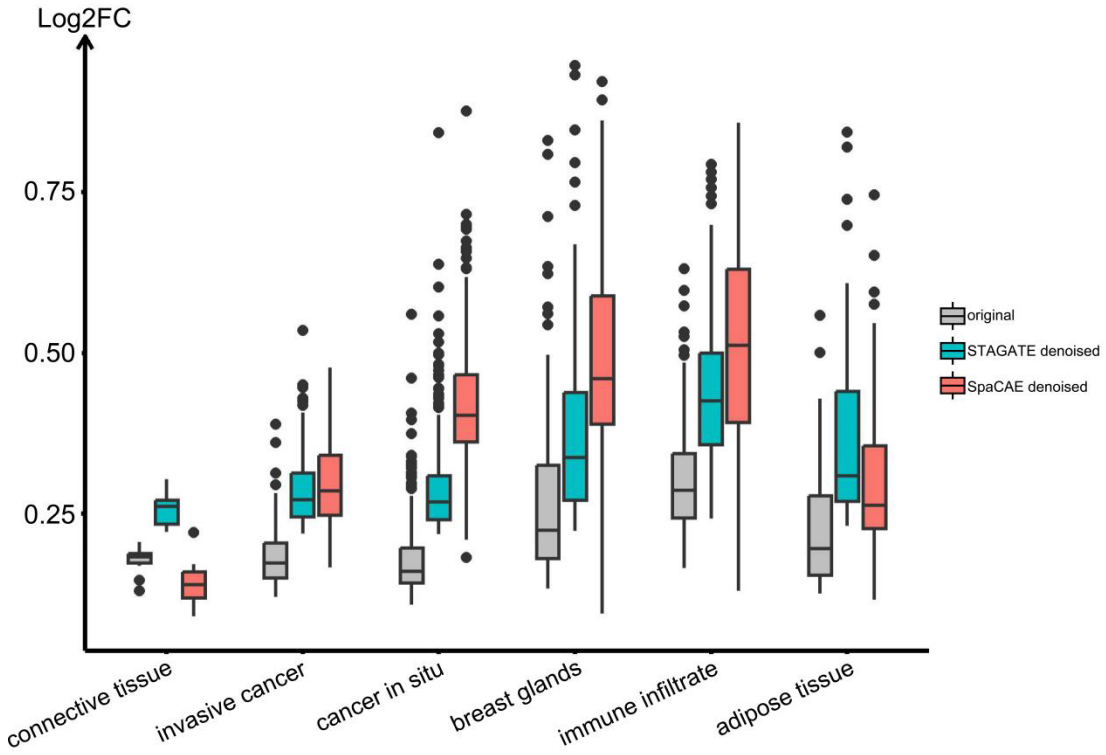
Supplementary Figure 6. The spatial domain identification on human HER2 breast cancer ST data. The identified spatial domains by SpaCAE and competing methods are distinguished by distinctive colors without correspondence. Adjusted Rand index (i.e., ARI) is used to evaluate the clustering performance.



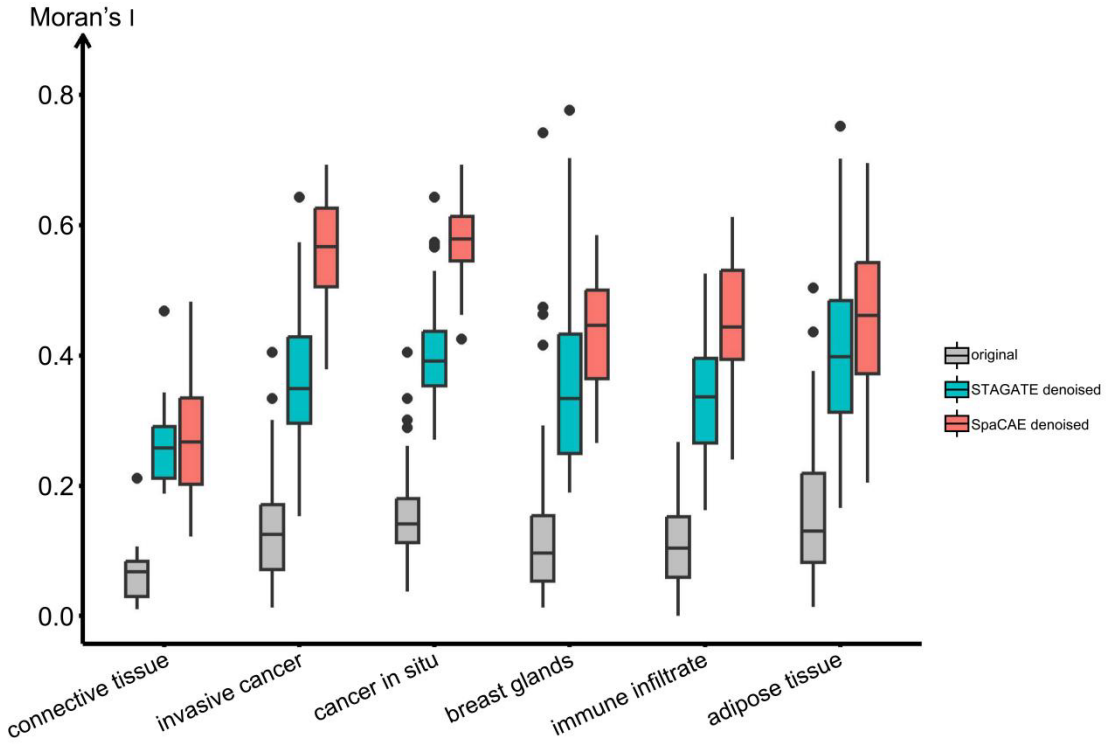
Supplementary Figure 7. The change of spatial expression patterns before and after denoising for the differentially expressed genes identified in each domain. Moran's I measures the spatial autocorrelation of the gene expression for each spatial domain.



Supplementary Figure 8. The change of spatial expression patterns before and after denoising (with SpaCAE and STAGATE) for the differentially expressed genes identified in the annotated tissue structure. Log2FC (i.e., log fold change) is used to evaluate the activities specificity of differential genes' expression patterns between original and denoised data.



Supplementary Figure 9. The change of spatial expression patterns before and after denoising (with SpaCAE and STAGATE) for the differentially expressed genes identified by the annotated tissue structure. Moran's I is used to evaluate the spatial coherence of each annotated tissue structure.



Supplementary Table 1. Detailed settings of parameters α and λ in the SRT data from different platforms.

Dataset	α	λ
10x Visium brain slice	0.1~1	1~2
osmFISH	0.8~1.5	1~3
MIBI-TOF	0.5~1	1~3
Slide-seq V2	2~5	1~3
10x Visium cancer slice	0.3~1	1~3