stDyer enables spatial domain clustering with dynamic graph embedding

Ke Xu[1], Yu Xu[1], Zirui Wang[1], Lu Zhang[1,2*]

[1*]Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong.

[2]Institute for Research and Continuing Education, Hong Kong Baptist University, No.2 Yuexing 3rd Rd., Shenzhen, 518000, Guandong, China.

*Corresponding author(s). E-mail(s): ericluzhang@hkbu.edu.hk;
Contributing authors: cskexu@comp.hkbu.edu.hk; csyuxu@comp.hkbu.edu.hk; cszrwang@comp.hkbu.edu.hk;

Abstract

Spatial transcriptomics data provide insights into gene expression patterns within tissue contexts, where identifying spatial domains with similar gene expression is crucial. Traditional clustering methods for spatial domain clustering often neglect spatial continuity, resulting in disjointed domains. Although recent computational approaches have integrated spatial information, they face limitations in recognizing domain boundaries, scalability, and the need for independent clustering steps. Here, we introduce stDyer, a novel end-to-end deep learning framework for spatial domain clustering in ST data. stDyer utilizes a Gaussian Mixture Variational AutoEncoder combined with graph attention networks to learn deep representations and perform clustering simultaneously. Unique to stDyer is its use of dynamic graphs that adaptively connect units based on Gaussian Mixture assignments in the latent space to improve spatial domain clustering and produce smoother domain boundaries. Additionally, stDyer's mini-batch neighbor sampling facilitates scalability to large-scale datasets and multi-GPU training capabilities. By benchmarking against state-of-the-art tools across multiple ST technologies, stDyer not only demonstrates superior performance in spatial domain clustering, but also shows its effectiveness in multi-slice analysis and handling large-scale datasets. We hope stDyer could contribute to advancing the development of spatial domain clustering and enable the analysis of large-scale dataset analysis.

Keywords: Spatial transcriptomics, Spatial domain clustering, Dynamic graphs, Deep learning

Introduction

Identifying spatial domains is a vital part of analyzing ST data. Spatial domains are distinct areas within a tissue, where the involved units (spots or cells, depending on the ST technologies) exhibit similar patterns of gene expression. These domains are crucial for understanding the spatial organization of gene activity [1, 2]. Traditional clustering or community detection methods, such as K-means and the Leiden algorithm, rely solely on gene expression and overlook spatial information to identify spatial domains. This often results in spatial domains that are not continuous in space. To overcome this limitation, many algorithms have been developed to consider both gene expression and the spatial coordinates of the units. They can be categorized into probabilistic approaches and deep learning models.

For probabilistic approaches, HMRF [3] groups the units with similar gene expression profiles and takes into account their spatial coordinates using Hidden Markov Random Field. The expectation-maximization algorithm and Gaussian density function are employed to estimate model parameters. However, estimating such a large number of parameters in the variable precision matrix across clusters may lead to unstable results. BayesSpace [4] employs a t-distributed error model to generate more robust spatial domains and improves HMRF by utilizing Markov Chain Monte Carlo for parameter inference, allowing for more efficient exploration in the parameter space. BASS [5] adopts a Bayesian hierarchical framework to detect spatial domains and their featured cell types on multiple samples simultaneously.

Deep learning models employ graph convolutional networks (GCNs) [6] on a spatial graph constructed using spatial coordinates of units, such as the K-nearest neighbor (KNN) graph. They group unit embeddings using unsupervised clustering methods to recognize spatial domains within ST data. SpaGCN [7] builds a graph by integrating the spatial coordinates and histological features of the units and applies GCNs to aggregate information from the neighboring units on this graph. STAGATE [8] introduces a graph attention networks (GATs) to a spatial graph to capture intricate spatial dependencies on gene expression and incorporates a cell type-aware module from pre-clustered gene expressions. GraphST [9] enhances the graph embeddings of units by adopting a graph self-supervised contrastive learning model that considers both gene expression profiles and spatial information.

Although the existing methods have been successfully applied to identify spatial domains for various tissues and species, many studies have explored that their performance is still unsatisfactory compared to human-curated benchmarks [1, 10, 11]. All these methods rely on a fixed spatial graph and assume that a unit is very likely to be derived from the same spatial domain with its neighboring units. This assumption may not always be valid, especially for the ones positioned at spatial domain's boundaries, where their neighboring units belong to multiple domains. The misclassification of those boundary units may also be extended to influence the units within the domain. Additionally, most existing deep learning models for spatial domain clustering [8, 12–14] are not designed to be end-to-end, resulting in suboptimal performance as they necessitate an independent clustering step. The application of existing tools to multi-slice and large-scale datasets poses another challenge with respect to scalability [15]. It has been observed that probabilistic approaches often require extensive

processing time for large-scale datasets due to the absence of GPU acceleration, while deep learning models face limitations in graphics memory (Figure 7c) because they need to load the entire spatial graph in the memory. There are currently no methods tailored for spatial domain clustering that can effectively address all of these three issues simultaneously.

In our study, we introduce stDyer, an end-to-end and scalable deep spatial domain clustering approach on ST data. stDyer employs a Gaussian Mixture Variational AutoEncoder [16–18] (GMVAE) with GAT [19] and graph embedding in the latent space. Instead of using an independent clustering step, stDyer enables deep representation learning and clustering from Gaussian Mixture Models (GMMs) simultaneously. stDyer also introduces dynamic graphs to involve more edges to a KNN spatial graph. These connections link a unit to others that belong to the same Gaussian Mixture as it does. Dynamic graphs can increase the likelihood that units at the domain boundaries establish connections with others belonging to the same spatial domain. Moreover, stDyer introduces mini-batch neighbor sampling to enable its application to large-scale datasets. To our knowledge, stDyer is the first method that could enable multi-GPU training for spatial domain clustering. We compared stDyer with seven state-of-the-art tools on four different ST technologies, including 10x Visium, osmFISH, STARmap and Stereo-seq. stDyer demonstrated excellent performance on all these technologies and produced smooth domain boundaries. In the ablation studies, we demonstrated that dynamic graphs could substantially improve spatial domain clustering when compared to the utilization of a KNN spatial graph (stDyer(KNN)). stDyer could also be extended to analyze ST data from multiple slices jointly and scale up to large-scale datasets.

Results

Workflow of stDyer

stDyer employs GMVAE with GAT and graph embedding on dynamic graphs to improve spatial domain clustering using ST data (Figure 1). A KNN spatial graph is constructed using the spatial coordinates of units within ST data. This KNN spatial graph, together with gene expression profiles, is fed into a GMVAE, of which the encoder layer is designed as GATs (Supplementary Figure S5a) to consider the graph structure (Supplementary Figure S5b and **Methods**). Besides GATs, stDyer also considers graph embedding in the latent space by encouraging the gene expression of a unit to be reconstructed by its neighbors in the graph. GMVAE assumes the unit embeddings follow GMMs in the latent space, which could be used to generate temporary spatial domain labels for all units at each epoch. During the first training epoch, the KNN spatial graph is adjusted by linking unit i with additional neighboring units (12 for 10x Visium technology and 4 for other technologies), where all these units (including i) should share the same temporary spatial label (**Methods**). This is the initial dynamic graph we generated. From the second training epoch, dynamic graphs are continuously updated based on temporary spatial labels from GMMs. stDyer includes a post-processing module to remove outliers and smooth spatial domain boundaries by jointly considering the predicted spatial labels of all units (**Methods**). stDyer also

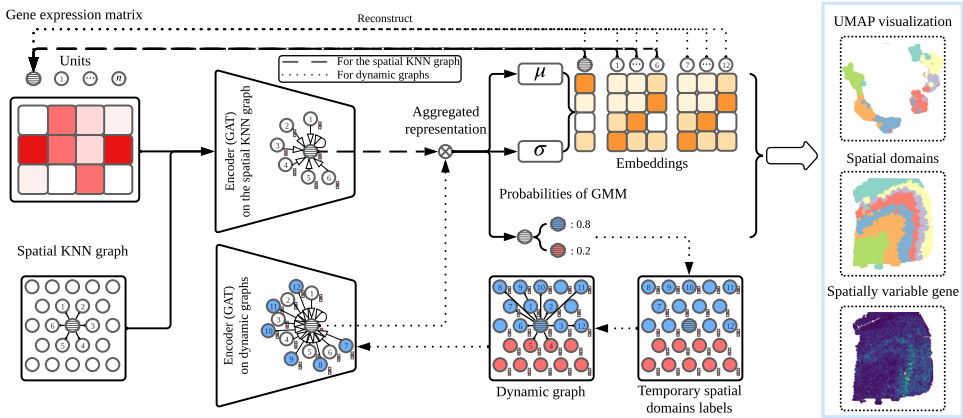

Fig. 1: The workflow of stDyer. stDyer accepts gene expression profiles and a KNN spatial graph (K=6, units 1 to 6 for the shallow unit) from ST data as inputs. The deep embeddings and clustering of units are generated by GMVAE with GATs and graph embedding. The parameters for GMMs (μ and σ) and unit clustering labels are estimated by maximizing the log-likelihood of the marginal distribution across all units (**Methods**). In the latent space, stDyer encourages the embedding of a unit to be reconstructed by its neighboring units in a KNN (dynamic) graph (e.g., 6 neighbors for a KNN spatial graph and 12 neighbors for a dynamic graph in this example), ensuring the concordance of local structure in graphs. A temporary spatial domain label (red and blue) for each unit is generated by GMMs after each epoch. At the first training epoch, the KNN spatial graph is updated by connecting each target unit (e.g., shallow unit) to 6 additional units (e.g., units 7 to 12), where all these 7 units should share the same temporary spatial domain label from GMM. This dynamic graph and temporary spatial domain labels will be continuously updated after each epoch. stDyer also enables the identification of spatially variable genes using integrated gradient analysis.

applies integrated gradient (IG) analysis (**Methods**) to identify spatially variable genes (SVGs) with spatially correlated expression.

stDyer improves spatial domain identification on the DLPFC dataset from 10x Visium technology

We evaluated the performance of stDyer using the human dorsolateral prefrontal cortex (DLPFC) dataset [1] generated by 10x Visium technology. This dataset includes 12 slices, with units ranging from 3,460 to 4,789. Each slice has 5 or 7 annotated spatial domains. We benchmarked stDyer with 8 state-of-the-art methods for spatial domain clustering: HMRF, SpaGCN, BayesSpace, SEDR, DeepST, BASS, STAGATE, and GraphST based on adjusted rand index (ARI; **Methods**). We evaluated these methods on all 12 slices (Figure 2f) and observed stDyer achieved an average ARI of 0.612, which was significantly higher than the second-best method GraphST (average ARI=0.538, p-value=0.014, Wilcoxon rank sum test). We also performed an ablation study to

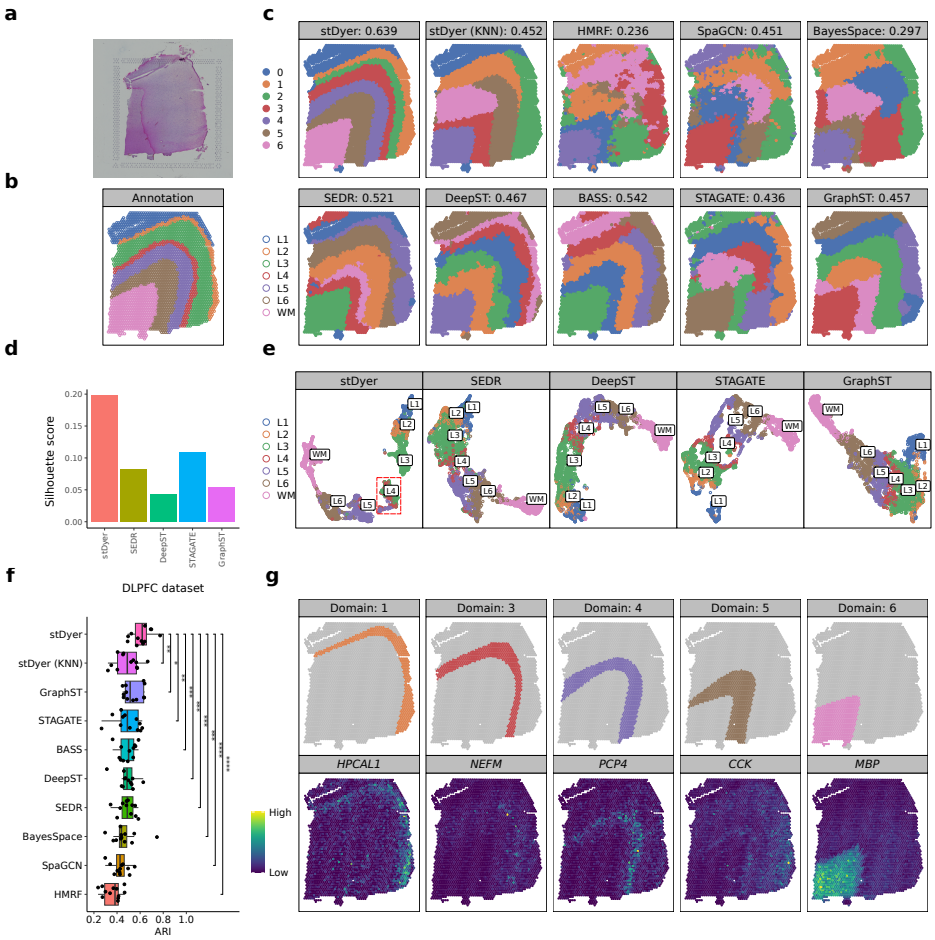

Fig. 2: The performance of stDyer on 12 slices from the DLPFC dataset generated by 10x Visium technology. (**a**) The histology image of slice 151674 of the DLPFC dataset. (**b**) The visualization of spatial domain annotation of slice 151674. (**c**) Spatial domain clustering and ARI scores of different methods. (**d**) The Silhouette scores of the five methods that can generate unit embeddings in the latent space. (**e**) UMAP plots of unit embeddings generated by the five methods in **d**. (**f**) The boxplot of ARIs from all 12 slices. The p-values were computed by the Wilcoxon rank sum test (****: p-value<0.0001; ***: p-value<0.001; **: p-value<0.01; *: p-value<0.05). (**g**) Visualization of SVG expression values (bottom) and their associated spatial domains (top). ARI: adjusted rand index. stDyer (KNN): stDyer with a KNN graph.

examine the efficacy of using dynamic graphs in stDyer. Our results indicated that stDyer (average ARI=0.612, p-value=0.002, Figure 2f) substantially outperformed stDyer (KNN) (average ARI=0.486).

By visualizing slice 151674 as an example, only stDyer (Figure 2c) could identify six laminar layers and white matter (WM) in the annotation (Figure 2b). Furthermore, the spatial domain embeddings from stDyer exhibit continuity and adhere to

the adjacency pattern in the annotation from L1 (domain 0) to WM (domain 6) (Figure 2e). There are five methods that could generate unit embeddings (stDyer, SEDR, DeepST, STAGATE and GraphST). Among these five methods, stDyer produced the highest Silhouette score (Figure 2d) and it was the only one that could separate out L4 domain (highlighted by the red rectangle in Figure 2e). The other methods struggled to distinguish between L2 to L6, e.g., GraphST failed to separate L4 and L5 domains (Figure 2c). Similar patterns were observed on the other slices (Supplementary Figure S1).

We selected 50 SVGs with the highest IG values for each domain (**Methods**, Supplementary Table S1 and Figure S2) and found some of them were reported as marker genes of cortex layers. For example, a previous study identified HPCAL1, PCP4, and MBP as marker genes of L2, L5, and WM [1] (Figure 2g), respectively. The same study also revealed *CCK* could distinguish L2, L3, and L6 from the other layers. We found that *NEFM* could be a marker gene of L4.

stDyer supports multi-slice spatial domain clustering

We divided 12 slices from the DLPFC datasets into three slice sections according to their spatial distance between in the third dimension, thus each section includes four adjacent slices (Figure 3d, Table S4) without obvious batch effect (Figure 3e with slice ids) [1]. The KNN spatial graph was constructed in the three-dimensional space (**Methods**), which could include neighboring units from both the same and adjacent slices. By considering slices 151507, 151508, 151509 and 151510 jointly, we observed the performance of multi-slice stDyer on 151510 (stDyer (M), ARI=0.572) significantly surpassed its performance by only considering single slice (stDyer (S), ARI=0.496, Figure 3c). We also compared stDyer with three methods (GraphST, STAGATE and SEDR) that also supported multi-slice clustering and generated unit embeddings. The multi-slice stDyer achieved an average ARI of 0.650 for all three sections, substantially outperforming the second-best method, GraphST, which had an average ARI of 0.540 (p-value=0.019, Figure 3a). We also observed that multi-slice stDyer provided more accurate unit embeddings of slender domains including L2, L4, L5 and L6 (Figure 3e). Both Silhouette scores (Figure 3b) and UMAP visualization (Figure 3e) demonstrated that the unit embeddings from multi-slice stDyer were much more consistent with the spatial domain annotation.

stDyer recognizes both tumor and interface domains on a zebrafish tumor dataset from 10x Visium technology

We further applied stDyer to a zebrafish melanoma dataset from 10x Visium technology [11] to assess its ability to identify highly heterogeneous domains. This dataset comprises 2,179 units with 16,107 genes, including spatial domains from tumor, interface, and normal tissues. The normal tissue domains are further categorized into the brain, muscle, secondary muscle, skin and others. stDyer with dynamic graphs (ARI=0.616) was the only method that identified both tumor and interface domains (Figure 4a); it could also refine the boundary of the interface domain and recognize the secondary muscle domain compared to stDyer with a KNN spatial graph

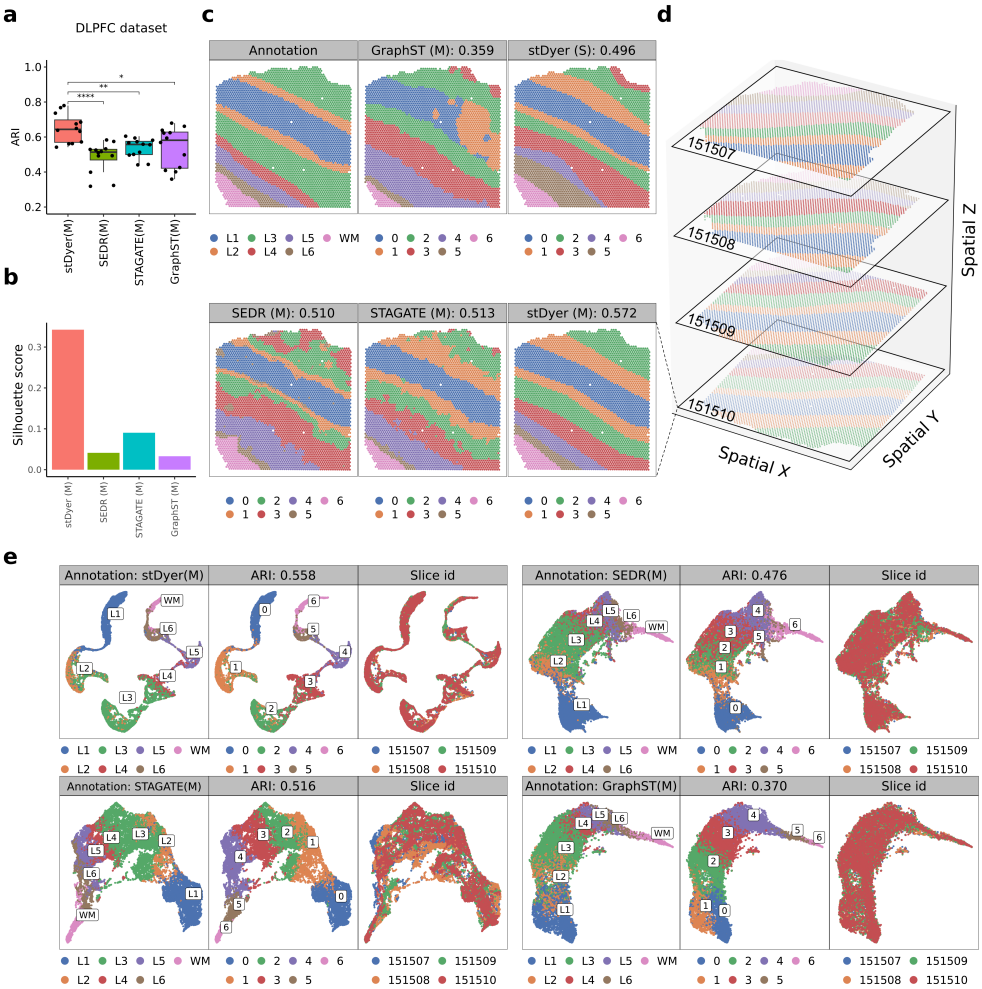

Fig. 3: The performance of stDyer on the DLPFC dataset by jointly considering multiple slices. (**a**) Comparison of four methods that could perform spatial domain clustering jointly. The p-values were computed by the Wilcoxon rank sum test (****: p-value<0.0001; **: p-value<0.01; *: p-value<0.05). (**b**) The Silhouette scores of the four methods in **a** and stDyer on the single slice (stDyer(S)). (**c**) Visualization of spatial domains and ARI scores for the four methods described in **a** on slice 151510. (**d**) The spatial organization of four slices in a section. (**e**) UMAP visualization by labeling units with annotated spatial domains, predicted spatial domains and slice ids.

(stDyer (KNN), ARI=0.590). SpaGCN (ARI=0.445), BASS (ARI=0.418), SEDR (ARI=0.286) and DeepST (ARI=0.452) could correctly identify the interface domain, but they divided tumor domains into more than one part. BayesSpace (ARI=0.367), STAGATE (ARI=0.460) and GraphST (ARI=0.360) were unable to distinguish the interface domain from the surrounding units.

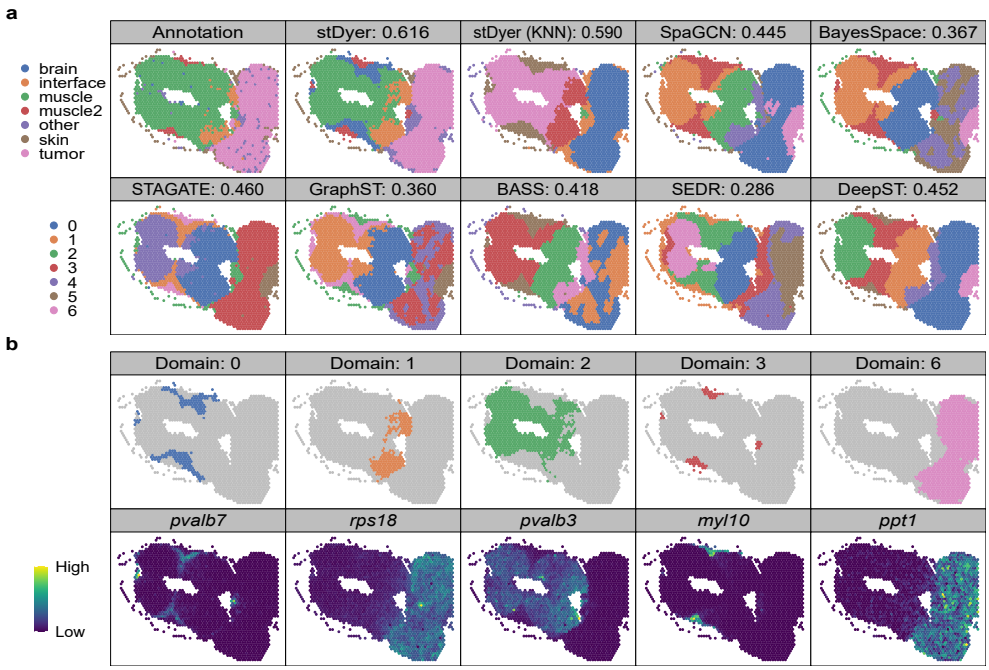

Fig. 4: The performance of stDyer on a zebrafish tumor dataset from 10x Visium technology. (**a**) Visualization and ARIs of different methods for spatial domain clustering. (**b**) Visualization of SVG expression values (bottom) and their associated spatial domains (top). muscle2: secondary muscle.

By considering IG information, we observed that the gene expression of the selected SVGs exhibited clear correlations with the corresponding spatial domains (Figure 4b, Supplementary Table S3 and Figure S4). We found *ppt1* uniquely showed high expression profiles within the tumor domain (Figure 4b). Its homeotic gene *PPT1* has been found to be associated with tumor promotion and prognosis [20]. We noticed that the expression levels of *rps18* progressively decreased as the units transitioned from the tumor domain through the interface domain and into the muscle domain. This pattern indicated that *rps18* might play a role in tumor progression.

stDyer generates spatial domains with smooth boundaries on the mouse cortex dataset from osmFISH

We analyzed a mouse cortex dataset comprising 4,839 units and 33 genes from osmFISH [10] to evaluate the performance of stDyer on *in situ* hybridization technology. The units can be classified into 11 spatial domains, including hippocampus (HPC), internal capsule and caudoputamen (ICC), neocortical layer 1 (L1), neocortical layer 2-3 lateral (L2-3l), neocortical layer 2-3 medial (L2-3m), neocortical layer 3-4 (L3-4), neocortical layer 4 (L4), neocortical layer (L5), neocortical layer (L6), ventricle (VT), and white matter (WM). stDyer achieved the best performance with ARI of 0.733 (Figure 5a), which was higher than stDyer (KNN) (ARI=0.632) and all seven

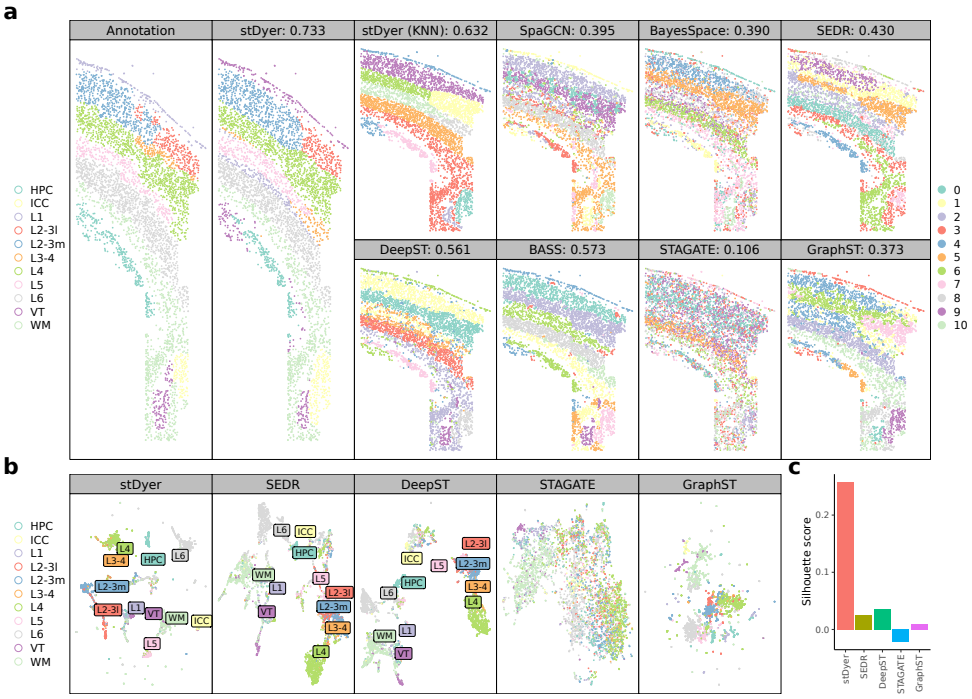

Fig. 5: The performance of stDyer on a mouse cortex dataset from osmFISH. (**a**) Visualization and ARIs of different methods for spatial domain clustering. stDyer (KNN): stDyer with a KNN spatial graph. (**b**) UMAP visualization by labeling units with annotated spatial domains. (**c**) The Silhouette scores for the five methods that provide unit embeddings.

other state-of-the-art tools (the second-best tool BASS with ARI=0.573). stDyer (ARI=0.733), DeepST (ARI=0.561), BASS (ARI=0.573) and GraphST (ARI=0.373) could all generate notably smooth spatial domain boundaries (Figure 5a), whereas the remaining three methods preferred to label adjacent units with different domains. Both DeepST and BASS incorrectly merged L2-3l and L2-3m domains in their results. stDyer generated highly separable unit embeddings from different domains (Figure 5b) and obtained the highest Silhouette score (Figure 5c). It was noted that stDyer generated unit embeddings of L2-3l domain as a line shape that could be easily distinguished from L2-3m.

stDyer detects clear laminar structures on the mouse cortex dataset from STARmap

stDyer was used to analyze a mouse cortex dataset [21] comprising 1,207 units and 1,020 genes from STARmap to evaluate the performance of stDyer on *in situ* sequencing technologies. The units in this dataset were annotated to be from seven spatial domains, including corpus callosum (CC), hippocampus (HPC), neocortical layer 1 (L1), neocortical layer 2/3 (L2/L3), neocortical layer 4 (L4), neocortical layer 5 (L5),

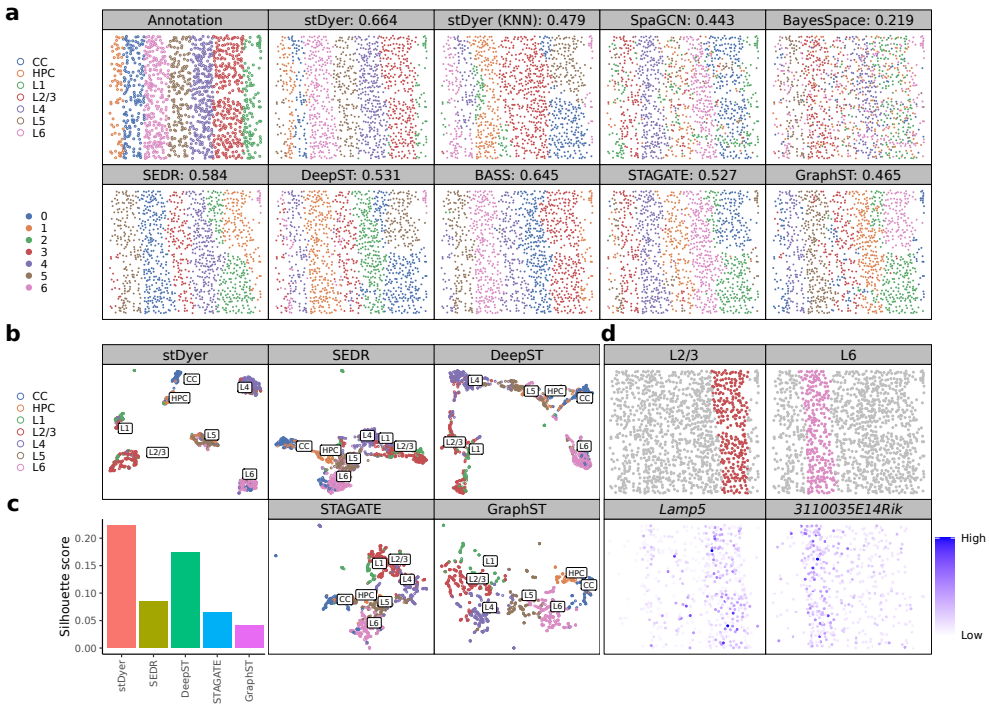

Fig. 6: The performance of stDyer on a mouse cortex dataset from STARmap. (**a**) stDyer detected clearer laminar structures and boundaries compared to the other tools. (**b**) UMAP visualization by labeling units with annotated spatial domains. (**c**) The Silhouette scores for the five methods that generate unit embeddings. (**d**) Visualization of SVG expression values (bottom) and their associated spatial domains (top).

and neocortical layer 6 (L6). As shown in Figure 6a, stDyer showed superior performance (ARI=0.664) in predicting the laminar structure of the mouse cortex. The other tools demonstrated issues in predicting different layers. There were only a few cells annotated in the second and sixth domains of BASS (ARI=0.645) and SEDR (ARI=0.584), which could be due to incorrect labeling of units at the domain boundaries. DeepST (ARI=0.531) and STAGATE (ARI=0.527) divided L2/L3 into two vertical sections. SpaGCN (ARI=0.443) failed to predict the hippocampus domain, while GraphST (ARI=0.465) was unable to predict L1 and divided L2/L3 erroneously. BayesSpace (ARI=0.219) exhibited poor performance on this dataset, yielding only four spatial domains with clear boundaries. We observed the unit embeddings from stDyer showed excellent separation between different spatial domains (Figure 6b) and achieved the highest Silhouette score (Figure 6c) compared to the other four methods. We observed that the IG analysis (Supplementary Table S2 and Figure S3) also revealed SVGs associated with two annotated layers (Figure 6d).

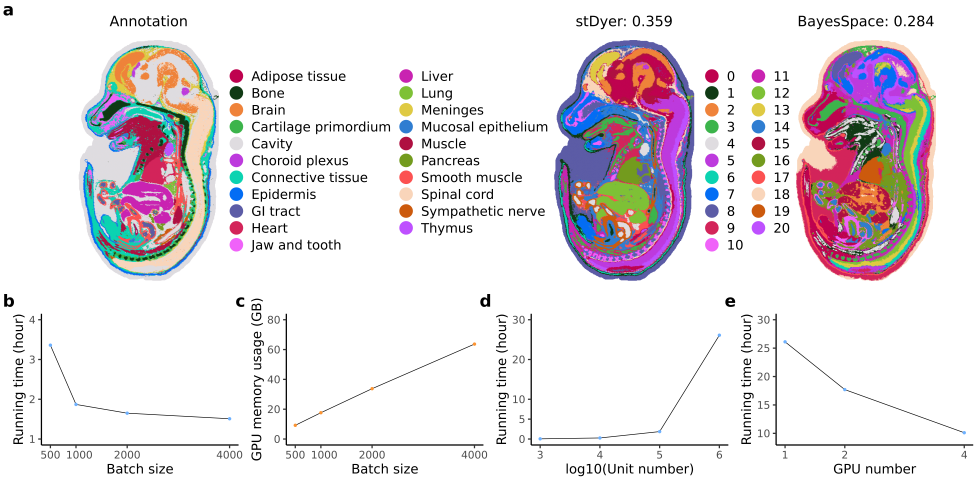

Fig. 7: The performance of stDyer on a large-scale mouse embryo dataset from Stereo-Seq. (**a**) Visualization and ARIs of stDyer and BayesSpace on E16.5_E2S7 slice in MOSTA dataset. (**b**) The impact of batch size on running time for stDyer to analyze 100,000 units. (**c**) The impact of batch size on GPU memory usage for stDyer to analyze 100,000 units. (**d**) The impact of the unit number on running time for stDyer. (**e**) The impact of GPU number on running time for stDyer to analyze 1,000,000 units.

stDyer can be efficiently scaled up to accommodate a large-scale mouse embryo dataset from Stereo-Seq

We attempted to apply stDyer to a mouse embryo dataset from Stereo-seq [2], including 155,047 units with 3,000 highly variable genes. stDyer (ARI=0.359, Figure 7a) finished spatial domain clustering on this dataset within one hour with six NVIDIA A100 sets, each equipped with 80GB of graphics memory. SpaGCN failed to finish clustering on this Stereo-seq dataset within a week due to its lack of GPU acceleration. BayesSpace finished the whole procedure in around 25 hours on the top 15 principal components of the gene expression matrix with 256 CPU cores, but its performance (ARI=0.284) was much worse than stDyer. STAGATE, SEDR, DeepST and GraphST exceeded the maximum graphics memory limit of 80GB, as they were required to keep the entire spatial graph in memory. BASS raised a stack overflow error while processing such a large number of units.

Because stDyer adopts a mini-batch technique for training, the entire spatial/dynamic graph does not necessarily need to be loaded into GPU at once; instead, they will be divided into small batches with limited units to enable stDyer to be applied to large-scale datasets. We investigated the impact of batch sizes and the unit number on the running time and GPU memory usage on the same Stereo-seq data (a set of NVIDIA A100 Figure 7b, c and d). We found that setting the batch size to 1,000 can significantly reduce the running time when the total unit number was 100,000 (Figure 7b). It only required 17.64 GB GPU memory (Figure 7c), which could even be satisfied by consumer GPUs like RTX 3090. By varying the total unit numbers from 1,000 to 1,000,000 (batch size=1,000, Figure 7d), we observed stDyer only required

1.87 and 26.11 hours to process 100,000 and 1,000,000 units, respectively. We also involved four sets of NVIDIA A100 and found the processing time could reduce to 10.11 hours for 1,000,000 units (Figure 7e).

Discussion

The ST technologies have revolutionized our understanding of the spatial dynamics of cells and gene expression, providing an efficient way to investigate the intricate relationship between gene activity and its location within tissue development or tumor microenvironment. Based on ST data, spatial domains could be identified to investigate spatial heterogeneity of gene expression patterns and their functional implications. The performance of existing tools for spatial domain clustering is still unsatisfactory and it is challenging to accurately delineate the boundaries of spatial domains. In this study, we introduced stDyer as an end-to-end and scalable graph-based deep learning model for spatial domain clustering using dynamic graphs. Instead of using a fixed spatial graph, the edges in dynamic graphs would be updated based on units' temporary spatial domain labels from GMVAE. Thus, there is a high chance that units sharing the same spatial domain labels could be connected in dynamic graphs. We evaluated the performance of stDyer on different ST technologies and tissues and proved it could substantially improve spatial domain clustering. The ablation studies also highlighted the superiority of the dynamic graphs over a KNN spatial graph.

Some ST technologies, such as 10x Visium and Stereo-seq, are capable of producing histological images of the same tissue slice, which is expected to be beneficial for spatial domain clustering. We tried to concatenate image patch embeddings with gene expression profiles of the 151674 slice from the DLPFC dataset as input for our model. We cut the whole image into small image patches for each unit and extracted the image feature embeddings using the pre-trained ResNet50 [22]. We found the ARI decreased to 0.237 (ARI without image: 0.639) and the predicted spatial domains were influenced much by noises in the histological image (Supplementary Figure S9). This suggested that image features could not guarantee to improve spatial domain clustering. To enhance clustering outcomes, the use of high-quality, biologically informative images or advanced image processing models will be required.

Nowadays, advanced ST technologies, such as Stereo-Seq, enable the measurement of a huge number of units for individual slices. Many graph-based approaches, such as SpaGCN and GraphST, struggle to handle large spatial graphs that contain upwards of 100,000 units due to their reliance on a basic implementation of graph convolutional networks. These methods attempt to load the entire graph into the GPU graphics memory, which is inherently limited and unable to expand at the same rate as the growing number of units being measured. To address this issue, stDyer utilizes a mini-batch technique to efficiently analyze graphs with millions of units. Additionally, stDyer is also able to support multiple slices spatial domain clustering, where the slices could be aligned vertically (e.g., DLPFC) or horizontally adjacent (e.g., concatenated as a single large slice).

stDyer is an open-source tool with broad applicability for all existing ST technologies. Although our experiments are limited to datasets produced by 10x Visium,

osmFISH, STARmap, and Stereo-seq, stDyer is designed to handle datasets from any ST technologies with gene expression profiles and spatial information. We have improved stDyer to enable the processing of multiple slices and incorporated both a mini-batch approach and multi-GPU support to efficiently handle the analysis of large-scale datasets. In the future, we would like to extend stDyer by incorporating histological images.

Methods

Data preprocessing

An ST dataset comprises two primary components: one matrix representing the gene expression profiles of m genes in n units ($\boldsymbol{X} \in \mathbb{R}^{n \times m}$) and the other matrix containing two-dimensional spatial coordinates for n units ($\boldsymbol{C} \in \mathbb{R}^{n \times 2}$). We performed data preprocessing on the gene expression matrix using functions provided by Scanpy [23]: 1. removed ERCC spike-ins; 2. removed units [scanpy.pp.filter_cells(min_counts=1)] and genes [scanpy.pp.filter_genes(min_counts=1)] with zero RNA counts; 3. selected top 3,000 highly variable genes [scanpy.pp.highly_variable_genes]; 4. library normalization [scanpy.pp.normalize_total]; 5. logarithmic transformation [scanpy.pp.log1p]; 6. gene-wise z-score standardization [scanpy.pp.scale].

Construct dynamic graph

We established the initial spatial graph by employing a KNN approach (K = 4 for smFISH, STARmap and Stereo-seq; K = 6 for 10x Visium due to its distinctive honeycomb structure) on the matrix $\boldsymbol{C}$ to link neighboring units based on Euclidean distance. During each training epoch of GMVAE, a temporary spatial domain label were assigned to each unit based on GMM (Figure S6). Subsequently, additional neighbors (12 for 10x Visium and 4 for smFISH, STARmap and Stereo-seq) for each unit were incorporated into the KNN spatial graph to connect units sharing identical spatial domain labels. For a given unit, the additional neighbors were chosen at random from other units that share the same temporary spatial domain label, unless these units are already included among the closest 18 (for 10x Visium technology) or 8 (for other ST technologies) spatial neighbors. In such a case, these closest units would be selected as unit additional neighbors. In a dynamic graph, each unit has a total of 18 (10x Visium) or 8 neighbors (other ST technologies).

Preprocessing of ST data from multiple slices

We divided 12 slices from the DLFPC dataset into three sections, each comprising four spatially contiguous slices, using the alignment method PASTE2 [24]. This method refines the existing two-dimensional spatial coordinates of units in the same slice to align them within a unified coordinate system. Given that each slice contains only two-dimensional spatial information, we introduced an additional artificial spatial coordinate of the third dimension to merge the four slices in the same section, thus enabling the construction of a three-dimensional spatial graph that spans across four slices. Assuming perfect overlap of the slices, this strategy allows each unit to

identify 6 spatial neighbors from its own slice, as well as from the slices immediately above and below it, providing a total of 18 nearest neighbors for the three-dimensional spatial graph construction.

The network structure of stDyer

To construct dynamic graphs, stDyer designs an end-to-end spatial domain clustering framework based on GMVAE with graph embedding (Supplementary Figure S5b). The framework includes an inference module (encoder) and a generative module (decoder). For the encoder, stDyer utilizes GATs (Supplementary Figure S5a) to aggregate gene expression profiles of neighbors of a given unit from a spatial/dynamic graph. Given a predefined number of clusters (k^{cluster}), the expression profiles of the target unit i ($\boldsymbol{X}_i$) and its neighbors ($\{\boldsymbol{X}_i\}^{\text{neigh}} \overset{\text{def}}{=} \{\boldsymbol{X}_r | r = 1, 2, \cdots, k_i^{\text{neigh}}\}$), we formulate the inference component as follows:

$$\boldsymbol{G}_i = \text{GAT}(\boldsymbol{X}_i, \{\boldsymbol{X}_i\}^{\text{neigh}}) \tag{1}$$

$$\boldsymbol{Logits}_i = \text{FC}(\boldsymbol{G}_i) \tag{2}$$

$$\hat{\boldsymbol{Y}}_i = \text{Softmax}(\boldsymbol{Logits}_i) \tag{3}$$

$$\boldsymbol{onehot}(j) = [0_1, \cdots, 0_{j-1}, 1_j, 0_{j+1}, \cdots, 0_{k^{\text{cluster}}}] \tag{4}$$

$$\begin{aligned} \boldsymbol{H}_{i,j} = \text{GAT}(& \\ & \text{concat}(\boldsymbol{X}_i, \boldsymbol{onehot}(j)), \\ & \text{concat}(\boldsymbol{X}_1, \boldsymbol{onehot}(j)), \\ & \cdots, \\ & \text{concat}(\boldsymbol{X}_{k_i^{\text{neigh}}}, \boldsymbol{onehot}(j))), j = 1, 2, \cdots, k^{\text{cluster}} \end{aligned} \tag{5}$$

$$\boldsymbol{\mu}_{i,j}^{\text{post}} = \text{FC}(\boldsymbol{H}_{i,j}), j = 1, 2, \cdots, k^{\text{cluster}} \tag{6}$$

$$\log \boldsymbol{\sigma}_{i,j}^{\text{post}^2} = \text{FC}(\boldsymbol{H}_{i,j}), j = 1, 2, \cdots, k^{\text{cluster}} \tag{7}$$

$$\boldsymbol{Z}_{i,j} = \boldsymbol{\mu}_{i,j}^{\text{post}} + \boldsymbol{\sigma}_{i,j}^{\text{post}} \cdot \epsilon, \epsilon \sim \mathcal{N}(0, 1) \tag{8}$$

The aggregated gene expression from the neighboring units of i is represented by $\boldsymbol{G}_i$, which is passed to a fully connected (FC) layer to generate $\boldsymbol{Logits}_i$ to predict the temporary spatial domain label $\hat{\boldsymbol{Y}}_i$. The spatial domain labels of units are from the modes of GMM they belong to. $\boldsymbol{H}_{i,j}$ aggregates gene expression from neighboring units of unit i by assuming it belongs to a spatial domain j. $\boldsymbol{H}_{i,j}$ is also used to derive parameters of the posterior distribution of the jth mode of GMM ($\boldsymbol{\mu}_{i,j}^{\text{post}}$ and $\boldsymbol{\sigma}_{i,j}^{\text{post}}$) for $\hat{\boldsymbol{X}}_{i,j}$. $\hat{\boldsymbol{X}}_{i,j}$ refers to the predicted $\boldsymbol{X}_i$ from the jth mode of GMM. $\boldsymbol{Z}_{i,j}$ is the embedding of $\hat{\boldsymbol{X}}_{i,j}$ sampled from $\mathcal{N}(\boldsymbol{\mu}_{i,j}^{\text{post}}, \boldsymbol{\sigma}_{i,j}^{\text{post}^2})$.

The decoder generates the parameters of the prior distribution of each component in GMM using $\boldsymbol{onehot}(j)$. It is also required to reconstruct $\boldsymbol{X}_i$ from $\boldsymbol{Z}_{i,j}$. We formulate the generative module as follows:

$$\boldsymbol{\mu}_j^{\text{prior}} = \text{FC}(\boldsymbol{onehot}(j)), j = 1, 2, \cdots, k^{\text{cluster}} \tag{9}$$

$$\log {\boldsymbol{\sigma}_j^{\text{prior}}}^2 = \text{FC}(\boldsymbol{onehot}(j)), j = 1, 2, \cdots, k^{\text{cluster}} \tag{10}$$

$$\boldsymbol{F}_{i,j} = \text{FC}(\boldsymbol{Z}_{i,j}), j = 1, 2, \cdots, k^{\text{cluster}} \tag{11}$$

$$\boldsymbol{\mu}_{i,j}^{\boldsymbol{x}} \stackrel{\text{def}}{=} \hat{\boldsymbol{X}}_{i,j} = \text{FC}(\boldsymbol{F}_{i,j}), j = 1, 2, \cdots, k^{\text{cluster}} \tag{12}$$

$$\log {\boldsymbol{\sigma}_{i,j}^{\boldsymbol{x}}}^2 = \text{FC}(\boldsymbol{F}_{i,j}), j = 1, 2, \cdots, k^{\text{cluster}} \tag{13}$$

$\boldsymbol{\mu}_j^{\text{prior}}$ and $\boldsymbol{\sigma}_j^{\text{prior}}$ denote the mean and standard deviation of the prior distribution of the jth Gaussian mixture. $\boldsymbol{\mu}_{i,j}^{\boldsymbol{x}}$ and $\boldsymbol{\sigma}_{i,j}^{\boldsymbol{x}}$ represent the mean and standard deviation of the distribution of $\hat{\boldsymbol{X}}_{i,j}$.

Probabilistic models of modified GMVAE in stDyer

We modified the GMVAE in stDyer by replacing its encoder from FC layers to GAT and utilized θ and ϕ to represent the parameters involved during generative (decoder) and inference (encoder) processes. The generative process could be represented as follows:

$$p(y_i = j) = \frac{1}{k^{\text{cluster}}}, j = 1, 2, \cdots, k^{\text{cluster}} \tag{14}$$

$$p_\theta(\boldsymbol{z}_i | y_i = j) = \mathcal{N}(\boldsymbol{\mu}_j^{\text{prior}}, {\boldsymbol{\sigma}_j^{\text{prior}}}^2), j = 1, 2, \cdots, k^{\text{cluster}} \tag{15}$$

$$p_\theta(\boldsymbol{x}_i | \boldsymbol{z}_i) = \mathcal{N}(\boldsymbol{\mu}_{i,j}^{\boldsymbol{x}}, {\boldsymbol{\sigma}_{i,j}^{\boldsymbol{x}}}^2), j = 1, 2, \cdots, k^{\text{cluster}} \tag{16}$$

$$p_\theta(y_i = j, \boldsymbol{z}_i, \boldsymbol{x}_i) = p(y_i = j) p_\theta(\boldsymbol{z}_i | y_i = j) p_\theta(\boldsymbol{x}_i | y_i = j, \boldsymbol{z}_i), j = 1, 2, \cdots, k^{\text{cluster}} \tag{17}$$

$\boldsymbol{x}_i$, y_i and $\boldsymbol{z}_i$ are three random variables that denote the gene expression profile, spatial domain label and latent embedding from the jth modes of GMM of the target unit i. The inference process is formulated as follows:

$$q_\phi(y_i = j | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}}) = \hat{\boldsymbol{Y}}_i(j), j = 1, 2, \cdots, k^{\text{cluster}} \tag{18}$$

$$q_\phi(\boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}}, y_i = j) = \mathcal{N}(\boldsymbol{\mu}_{i,j}^{\text{post}}, {\boldsymbol{\sigma}_{i,j}^{\text{post}}}^2), j = 1, 2, \cdots, k^{\text{cluster}} \tag{19}$$

$$q_\phi(y_i = j, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}}) = q_\phi(y_i = j | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}}) \cdot q_\phi(\boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}}, y_i = j), \quad j = 1, 2, \cdots, k^{\text{cluster}} \tag{20}$$

where $\hat{\boldsymbol{Y}}_i(j)$ denotes the predicted probability that the target unit i belongs to the jth mode of GMM, $\{\boldsymbol{x}_i\}^{\text{neigh}}$ represents the gene expression profiles of the neighbors of unit i. The spatial domain assignment vector $\hat{\boldsymbol{Y}}_i(j)$ is inferred by GAT in Equation (3).

The objective function of stDyer

The objective function of stDyer is to maximize the log-likelihood of the marginal distribution across all involved n units:

$$\max_{\theta,\phi} \sum_{i=1}^{n} \log p_\theta(\boldsymbol{x}_i) \tag{21}$$

stDyer incorporates a smoothness constraint to encourage neighboring units to share the same spatial domain labels and have contiguous embedding.

$$\mathrm{JS}_{\theta,\phi}^{\mathrm{smooth}} = \sum_{r=1}^{k_i^{\mathrm{neigh}}} \mathrm{JS}_{\theta,\phi}^{\mathrm{smooth}(r)} \tag{22}$$

$$\begin{aligned} \mathrm{JS}_{\theta,\phi}^{\mathrm{smooth}(r)} &\overset{\mathrm{def}}{=} \mathrm{JS}\big(q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\mathrm{neigh}}) \parallel q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\mathrm{neigh}})\big), \\ &\qquad r = 1, 2, \cdots, k_i^{\mathrm{neigh}} \end{aligned} \tag{23}$$

The objective function could be rewritten as:

$$\max_{\theta,\phi} \sum_{i=1}^{n} \log p_\theta(\boldsymbol{x}_i) - \mathrm{JS}_{\theta,\phi}^{\mathrm{smooth}} \tag{24}$$

where JS denotes Jensen-Shannon divergence and $\boldsymbol{x}_r$ refers to the gene expression profile of neighboring unit r of unit i. The log-likelihood $\log p_\theta(\boldsymbol{x}_i)$ for the target unit i can be decomposed as:

$$\begin{aligned} \log p_\theta(\boldsymbol{x}_i) = &\ \mathrm{KL}\big(q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\mathrm{neigh}}) \parallel p_\theta(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i)\big) \\ &+ E_{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\mathrm{neigh}})} \ln \frac{p_\theta(y_i, \boldsymbol{z}_i, \boldsymbol{x}_i)}{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\mathrm{neigh}})} \end{aligned} \tag{25}$$

stDyer aims to reconstruct $\boldsymbol{x}_i$ using its neighboring units $\boldsymbol{x}_r$ to promote similarity in their spatial domain labels and latent embeddings. Thus, Equation (25) could be rewritten as:

$$\begin{aligned} \log p_\theta(\boldsymbol{x}_i) = &\ \mathrm{KL}\big(q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\mathrm{neigh}}) \parallel p_\theta(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i)\big) \\ &+ E_{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\mathrm{neigh}})} \ln \frac{p_\theta(y_i, \boldsymbol{z}_i, \boldsymbol{x}_i)}{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\mathrm{neigh}})}, \\ &\ r = 1, 2, \cdots, k_i^{\mathrm{neigh}} \end{aligned} \tag{26}$$

By taking the average of both sides of the two equations, Equation (25) and the mean of Equation (26) over $r = 1, 2, \cdots, k_i^{\text{neigh}}$, we derived:

$$\begin{aligned}
\log p_\theta(\boldsymbol{x}_i) = \frac{1}{2k_i^{\text{neigh}}} \sum_{r=1}^{k_i^{\text{neigh}}} \Bigg[& \mathrm{KL}\big(q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}}) \parallel p_\theta(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i)\big) \\
& + \mathrm{KL}\big(q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}}) \parallel p_\theta(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i)\big) \\
& + E_{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{p_\theta(y_i, \boldsymbol{z}_i, \boldsymbol{x}_i)}{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}})} \\
& + E_{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{p_\theta(y_i, \boldsymbol{z}_i, \boldsymbol{x}_i)}{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})} \Bigg] \\
= \frac{1}{k_i^{\text{neigh}}} \sum_{r=1}^{k_i^{\text{neigh}}} & G_{\theta,\phi}^r - \frac{1}{2}(L_{\text{self}}^i + L_{\text{neigh}}^i)
\end{aligned} \tag{27}$$

$$L_{\text{self}}^i \stackrel{\text{def}}{=} E_{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}})}{p_\theta(y_i, \boldsymbol{z}_i, \boldsymbol{x}_i)} \tag{28}$$

$$L_{\text{neigh}}^i \stackrel{\text{def}}{=} E_{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})}{p_\theta(y_i, \boldsymbol{z}_i, \boldsymbol{x}_i)} \tag{29}$$

$$\begin{aligned}
G_{\theta,\phi}^r \stackrel{\text{def}}{=} \frac{1}{2} \Bigg[& E_{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{q_\phi(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i, \{\boldsymbol{x}_i\}^{\text{neigh}})}{p_\theta(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i)} \\
& + E_{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{q_\phi(y_r, \boldsymbol{z}_r | \boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})}{p_\theta(y_i, \boldsymbol{z}_i | \boldsymbol{x}_i)} \Bigg]
\end{aligned} \tag{30}$$

We also proved $\mathrm{JS}_{\theta,\phi}^{\text{smooth}(r)}$ was the lower bound of $G_{\theta,\phi}^r$ (Equation (32)):

$$G_{\theta,\phi} = \frac{1}{k_i^{\text{neigh}}} \sum_{r=1}^{k_i^{\text{neigh}}} G_{\theta,\phi}^r \tag{31}$$

$$
\begin{aligned}
G^r_{\theta,\phi} &= \frac{1}{2}\Bigg(E_{q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})}{p_\theta(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i)} \\
&\quad + E_{q_\phi(y_r,\boldsymbol{z}_r|\boldsymbol{x}_r,\{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{q_\phi(y_r,\boldsymbol{z}_r|\boldsymbol{x}_r,\{\boldsymbol{x}_r\}^{\text{neigh}})}{p_\theta(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i)} \Bigg) \\
&= \frac{1}{2}\Bigg(E_{q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})}{M} \\
&\quad + E_{q_\phi(y_r,\boldsymbol{z}_r|\boldsymbol{x}_r,\{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{q_\phi(y_r,\boldsymbol{z}_r|\boldsymbol{x}_r,\{\boldsymbol{x}_r\}^{\text{neigh}})}{M} \Bigg) + E_M \ln \frac{M}{p_\theta(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i)} \\
&= \text{JS}^{\text{smooth}(r)}_{\theta,\phi} + \text{KL}(M \parallel p_\theta(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i)) \\
&\geq \text{JS}^{\text{smooth}(r)}_{\theta,\phi}
\end{aligned} \tag{32}
$$

where $M = \frac{1}{2}[q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}}) + q_\phi(y_r,\boldsymbol{z}_r|\boldsymbol{x}_r,\{\boldsymbol{x}_r\}^{\text{neigh}})]$.

Thus, the objective function Equation (24) could be reformatted with Equation (27) as:

$$
\max_{\theta,\phi} \sum_{i=1}^{n} \log p_\theta(\boldsymbol{x}_i) - \text{JS}^{\text{smooth}}_{\theta,\phi} = \max_{\theta,\phi} \sum_{i=1}^{n} \log p_\theta(\boldsymbol{x}_i) - G_{\theta,\phi} = \min_{\theta,\phi} \sum_{i=1}^{n} \frac{1}{2}(L^i_{\text{self}} + L^i_{\text{neigh}}) \stackrel{\text{def}}{=} L_{\text{GMVAE}} \tag{33}
$$

As defined in Equation (28) and Equation (29), L^i_{self} and L^i_{neigh} can be decomposed as:

$$
L^i_{\text{self}} = L^{\text{self}}_{\text{rec}} + L^{\text{self}}_{\text{gauss}} + L^{\text{self}}_{\text{cat}} \tag{34}
$$

$$
L^i_{\text{neigh}} = L^{\text{neigh}}_{\text{rec}} + L^{\text{neigh}}_{\text{gauss}} + L^{\text{neigh}}_{\text{cat}} = \frac{1}{k_i^{\text{neigh}}} \sum_{r=1}^{k_i^{\text{neigh}}} L^r_{\text{rec}} + L^r_{\text{gauss}} + L^r_{\text{cat}} \tag{35}
$$

$$
L^{\text{self}}_{\text{rec}} \stackrel{\text{def}}{=} E_{q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{1}{p_\theta(\boldsymbol{x}_i|\boldsymbol{z}_i)} \tag{36}
$$

$$
L^{\text{self}}_{\text{gauss}} \stackrel{\text{def}}{=} E_{q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{q_\phi(\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}},y_i)}{p_\theta(\boldsymbol{z}_i|y_i)} \tag{37}
$$

$$
L^{\text{self}}_{\text{cat}} \stackrel{\text{def}}{=} E_{q_\phi(y_i,\boldsymbol{z}_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})} \ln \frac{q_\phi(y_i|\boldsymbol{x}_i,\{\boldsymbol{x}_i\}^{\text{neigh}})}{p_\theta(y_i)} \tag{38}
$$

$$
L^{\text{neigh}}_{\text{rec}} \stackrel{\text{def}}{=} \frac{1}{k_i^{\text{neigh}}} \sum_{r=1}^{k_i^{\text{neigh}}} L^r_{\text{rec}} \tag{39}
$$

$$L_{\text{gauss}}^{\text{neigh}} \stackrel{\text{def}}{=} \frac{1}{k_i^{\text{neigh}}} \sum_{r=1}^{k_i^{\text{neigh}}} L_{\text{gauss}}^{r} \tag{40}$$

$$L_{\text{cat}}^{\text{neigh}} \stackrel{\text{def}}{=} \frac{1}{k_i^{\text{neigh}}} \sum_{r=1}^{k_i^{\text{neigh}}} L_{\text{cat}}^{r} \tag{41}$$

$$L_{\text{rec}}^{r} \stackrel{\text{def}}{=} E_{q_\phi(y_r, \boldsymbol{z}_r|\boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{1}{p_\theta(\boldsymbol{x}_i|\boldsymbol{z}_i)} \tag{42}$$

$$L_{\text{gauss}}^{r} \stackrel{\text{def}}{=} E_{q_\phi(y_r, \boldsymbol{z}_r|\boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{q_\phi(\boldsymbol{z}_r|\boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}}, y_r)}{p_\theta(\boldsymbol{z}_i|y_i)} \tag{43}$$

$$L_{\text{cat}}^{r} \stackrel{\text{def}}{=} E_{q_\phi(y_r, \boldsymbol{z}_r|\boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})} \ln \frac{q_\phi(y_r|\boldsymbol{x}_r, \{\boldsymbol{x}_r\}^{\text{neigh}})}{p_\theta(y_i)} \tag{44}$$

Here, $L_{\text{rec}}^{\text{self}}$ represents the loss when reconstructing the target unit from its own embedding, while L_{rec}^{r} represents the loss when reconstructing the target unit from the embedding of its rth neighboring unit. Both $L_{\text{gauss}}^{\text{self}}$ and L_{gauss}^{r} regularize the parameters of the posterior distribution to ensure consistency with those of the prior distribution in each mixture for the target unit i and its rth neighboring unit, respectively. The $\dim(\boldsymbol{Z}_{i,j})$ denotes the number of dimensions of $\boldsymbol{Z}_{i,j}$. Finally, $L_{\text{cat}}^{\text{self}}$ and L_{cat}^{r} encourage the alignment between the prior distribution and the posterior distribution for the predicted GMM assignment of the target unit and its rth neighbor unit.

Using previously defined generative and inference processes, $L_{\text{rec}}^{\text{self}}$, $L_{\text{gauss}}^{\text{self}}$, $L_{\text{cat}}^{\text{self}}$, $L_{\text{rec}}^{\text{r}}$, $L_{\text{gauss}}^{\text{r}}$ and $L_{\text{cat}}^{\text{r}}$ can be further expanded as:

$$L_{\text{rec}}^{\text{self}} = \sum_{j=1}^{k^{\text{cluster}}} \frac{1}{2} \hat{\boldsymbol{Y}}_i(j) * \left[\left(\frac{\hat{\boldsymbol{X}}_{i,j} - \boldsymbol{X}_i}{\boldsymbol{\sigma}_{\boldsymbol{x}_{i,j}}} \right)^2 + \log 2\pi \boldsymbol{\sigma}_{\boldsymbol{x}_{i,j}}^2 \right] \tag{45}$$

$$L_{\text{rec}}^{r} = \sum_{j=1}^{k^{\text{cluster}}} \frac{1}{2} \hat{\boldsymbol{Y}}_r(j) * \left[\left(\frac{\hat{\boldsymbol{X}}_{rj} - \boldsymbol{X}_i}{\boldsymbol{\sigma}_{\boldsymbol{x}_{rj}}} \right)^2 + \log 2\pi \boldsymbol{\sigma}_{\boldsymbol{x}_{rj}}^2 \right], r = 1, 2, \cdots, k_i^{\text{neigh}} \tag{46}$$

$$\begin{aligned} L_{\text{gauss}}^{\text{self}} = \sum_{j=1}^{k^{\text{cluster}}} \frac{1}{2} \hat{\boldsymbol{Y}}_i(j) * \Bigg[& \mathrm{e}^{\log \boldsymbol{\sigma}_{i,j}^{\text{post}\,2}} - \mathrm{e}^{\log \boldsymbol{\sigma}_j^{\text{prior}\,2}} + \frac{(\boldsymbol{\mu}_{i,j}^{\text{post}} - \boldsymbol{\mu}_j^{\text{prior}})^2}{\mathrm{e}^{\log \boldsymbol{\sigma}_j^{\text{prior}\,2}}} \\ & + \dim(\boldsymbol{Z}_{i,j}) + \log \boldsymbol{\sigma}_j^{\text{prior}\,2} - \log \boldsymbol{\sigma}_{i,j}^{\text{post}\,2} \Bigg] \end{aligned} \tag{47}$$

$$\begin{aligned} L_{\text{gauss}}^{r} = \sum_{j=1}^{k^{\text{cluster}}} \frac{1}{2} \hat{\boldsymbol{Y}}_r(j) * \Bigg[& \mathrm{e}^{\log \boldsymbol{\sigma}_{rj}^{\text{post}\,2}} - \mathrm{e}^{\log \boldsymbol{\sigma}_j^{\text{prior}\,2}} + \frac{(\boldsymbol{\mu}_{rj}^{\text{post}} - \boldsymbol{\mu}_j^{\text{prior}})^2}{\mathrm{e}^{\log \boldsymbol{\sigma}_j^{\text{prior}\,2}}} \\ & + \dim(\boldsymbol{Z}_{i,j}) + \log \boldsymbol{\sigma}_j^{\text{prior}\,2} - \log \boldsymbol{\sigma}_{rj}^{\text{post}\,2} \Bigg], \\ & r = 1, 2, \cdots, k_i^{\text{neigh}} \end{aligned} \tag{48}$$

$$L_{\text{cat}}^{\text{self}} = \text{KL}(\hat{\boldsymbol{Y}}_i \parallel \frac{1}{k^{\text{cluster}}}) \tag{49}$$

$$L_{\text{cat}}^{r} = \text{KL}(\hat{\boldsymbol{Y}}_r \parallel \frac{1}{k^{\text{cluster}}}), r = 1, 2, \cdots, k_i^{\text{neigh}} \tag{50}$$

Training strategies of stDyer

stDyer adopted a two-stage training process. In the first stage (1-50 epochs), stDyer was trained based on the spatial KNN graph and the Gaussian losses ($L_{\text{gauss}}^{\text{self}}$ and $L_{\text{gauss}}^{\text{neigh}}$) was ignored to enable the model to focus more on the reconstruction losses. In the second stage trained on dynamic graphs, we applied mclust [25] to group unit embeddings after the 50th epoch and generate pseudo spatial domain labels $\tilde{\boldsymbol{Y}}$ for all units. We then initialized predicted spatial domain labels $\hat{\boldsymbol{Y}}$ using $\tilde{\boldsymbol{Y}}$ using cross-entropy loss Equation (S1) for 10 epochs (Supplementary Note S6). From the 61th to the 200th epoch, L_{GMVAE} was employed for model training and spatial domain clustering. We utilized an Adam optimizer to optimize these objective functions with a learning rate of 0.0001 and training epochs of 200. The prior distribution of unit domain labels was initialized as a uniform distribution and updated based on unit temporary spatial domain labels from the last epoch with a learning rate of 0.1.

Spatial domain label refinement

To refine the spatial domains with outlier labels, we used sklearn.neural_network.MLPClassifier to construct the autoencoder with mean squared error (MSE) as the loss function. For model training, the input and output of this autoencoder are unit spatial coordinates after min-max normalization and unit spatial domain labels from the modified GMVAE. We up-sampled the units from the smallest domain during model training to avoid this domain disappearing after refinement. This was done by scaling them up to the unit number of the second-smallest domains.

Integrated gradient analysis

We incorporated the integrated gradients analysis [26]) in stDyer to identify SVGs. The IG vector was used to represent the effect of gene g in predicting spatial domain label of unit i.

$$\boldsymbol{IG}_{ig} = \boldsymbol{X}_{i\cdot} \odot \left(\frac{1}{2} \times \frac{\partial(\frac{1}{m}\hat{\boldsymbol{Y}}_{ig})}{\partial \boldsymbol{X}_{i\cdot}} + \sum_{p=2}^{m-1} \frac{\partial(\frac{p}{m}\hat{\boldsymbol{Y}}_{ig})}{\partial \boldsymbol{X}_{i\cdot}} + \frac{1}{2} \times \frac{\partial(\frac{m}{m}\hat{\boldsymbol{Y}}_{ig})}{\partial \boldsymbol{X}_{i\cdot}} \right) \times \frac{1}{m} \tag{51}$$

where m (50 by default) is the number of steps needed in the summation procedure in Equation (51) to estimate the definite integral. $\boldsymbol{X}$ and $\hat{\boldsymbol{Y}}$ are gene expression profiles of units and their predicted spatial domain labels, respectively. We averaged IG vectors of gene g across all units for each spatial domain to estimate its impact in prediction. Thus, SVGs for each domain were selected if their IGs within the top 50 largest IGs.

Evaluation metric

We used Adjusted Rand Index (ARI) to evaluate the performance of spatial domain clustering [27]. ARI is a score computed between ground truth domain labels and predicted domain labels, ranging from -1 to 1. A higher ARI score indicates better clustering performance. ARI is defined as the following given the ground truth spatial domain $\boldsymbol{A} = (a_1, a_2, \cdots, a_t)$ and the predicted spatial domain $\boldsymbol{B} = (b_1, b_2, \cdots, b_p)$:

$$ARI = \frac{\sum_{ij} \binom{c_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] \Big/ \binom{n}{2}} \tag{52}$$

where $c_{ij} = |a_i \cap b_j|$, and n is the number of units. We computed the p-value of ARIs on the multi-slice DLPFC dataset using the Wilcoxon rank sum test with stat_compare_means function in ggpubr package [28]. The settings of hyper-parameters of stDyer and the other involved methods were shown in Supplementary Note S6 and Note S7.

Funding

This research was partially supported by Young Collaborative Research Grant (C2004-23Y), HMRF (11221026) and HKBU Start-up Grant Tier 2 (RC-SGT2/19-20/SCI/007).

Data availability

The human dorsolateral prefrontal cortex dataset [1] is available in https://research.libd.org/spatialLIBD/ in h5 format. The osmFISH dataset [10] can be downloaded at http://linnarssonlab.org/osmFISH/availability/ in loom format. The STARmap dataset [21] is accessible at https://figshare.com/articles/dataset/STARmap_datasets/22565209. The zebrafish tumor dataset [11] is available at the Gene Expression Omnibus (GEO) under accession number "GSE159709". The Stereo-seq dataset [2] is downloaded from https://ftp.cngb.org/pub/SciRAID/stomics/STDS0000058/stomics/E16.5_E2S7.MOSTA.h5ad.

Code availability

The source code of stDyer is publicly available at Github: https://github.com/ericcombiolab/stDyer.

Competing interests

The authors declare no competing interests.

Authors' contributions

L.Z conceived and supervised the project. K.X. and L.Z designed the method, conducted the experiments and wrote the manuscript. L.Z., Y.X. and Z.W. reviewed the manuscript. All authors approved the manuscript.

Supplementary information. There is a single supplementary file containing all supplementary materials.

References

[1] Maynard, K.R., Collado-Torres, L., Weber, L.M., Uytingco, C., Barry, B.K., Williams, S.R., Catallini, J.L., Tran, M.N., Besich, Z., Tippani, M., Chew, J., Yin, Y., Kleinman, J.E., Hyde, T.M., Rao, N., Hicks, S.C., Martinowich, K., Jaffe, A.E.: Transcriptome-scale spatial gene expression in the human dorsolateral prefrontal cortex. Nature Neuroscience **24**(3), 425–436 (2021) https://doi.org/10.1038/s41593-020-00787-0

[2] Chen, A., Liao, S., Cheng, M., Ma, K., Wu, L., Lai, Y., Qiu, X., Yang, J., Xu, J., Hao, S., Wang, X., Lu, H., Chen, X., Liu, X., Huang, X., Li, Z., Hong, Y., Jiang, Y., Peng, J., Liu, S., Shen, M., Liu, C., Li, Q., Yuan, Y., Wei, X., Zheng, H., Feng, W., Wang, Z., Liu, Y., Wang, Z., Yang, Y., Xiang, H., Han, L., Qin, B., Guo, P., Lai, G., Muñoz-Cánoves, P., Maxwell, P.H., Thiery, J.P., Wu, Q.-F., Zhao, F., Chen, B., Li, M., Dai, X., Wang, S., Kuang, H., Hui, J., Wang, L., Fei, J.-F., Wang, O., Wei, X., Lu, H., Wang, B., Liu, S., Gu, Y., Ni, M., Zhang, W., Mu, F., Yin, Y., Yang, H., Lisby, M., Cornall, R.J., Mulder, J., Uhlén, M., Esteban, M.A., Li, Y., Liu, L., Xu, X., Wang, J.: Spatiotemporal transcriptomic atlas of mouse organogenesis using dna nanoball-patterned arrays. Cell **0**(0) (2022) https://doi.org/10.1016/j.cell.2022.04.003

[3] Zhu, Q., Shah, S., Dries, R., Cai, L., Yuan, G.-C.: Identification of spatially associated subpopulations by combining scrnaseq and sequential fluorescence in situ hybridization data. Nature Biotechnology **36**(12), 1183–1190 (2018) https://doi.org/10.1038/nbt.4260

[4] Zhao, E., Stone, M.R., Ren, X., Guenthoer, J., Smythe, K.S., Pulliam, T., Williams, S.R., Uytingco, C.R., Taylor, S.E.B., Nghiem, P., Bielas, J.H., Gottardo, R.: Spatial transcriptomics at subspot resolution with bayesspace. Nature Biotechnology, 1–10 (2021) https://doi.org/10.1038/s41587-021-00935-2

[5] Li, Z., Zhou, X.: Bass: multi-scale and multi-sample analysis enables accurate cell type clustering and spatial domain detection in spatial transcriptomic studies. Genome Biology **23**(1), 168 (2022) https://doi.org/10.1186/s13059-022-02734-7

[6] Thomas N. Kipf, Max Welling: Semi-supervised classification with graph convolutional networks. International Conference on Learning Representations (2016)

[7] Hu, J., Li, X., Coleman, K., Schroeder, A., Ma, N., Irwin, D.J., Lee, E.B., Shinohara, R.T., Li, M.: Spagcn: Integrating gene expression, spatial location and histology to identify spatial domains and spatially variable genes by graph convolutional network. Nature Methods, 1–10 (2021) https://doi.org/10.1038/s41592-021-01255-8

[8] Dong, K., Zhang, S.: Deciphering spatial domains from spatially resolved transcriptomics with an adaptive graph attention auto-encoder. Nature Communications **13**(1), 1739 (2022) https://doi.org/10.1038/s41467-022-29439-6

[9] Long, Y., Ang, K.S., Li, M., Chong, K.L.K., Sethi, R., Zhong, C., Xu, H., Ong, Z., Sachaphibulkij, K., Chen, A., Zeng, L., Fu, H., Wu, M., Lim, L.H.K., Liu, L., Chen, J.: Spatially informed clustering, integration, and deconvolution of spatial transcriptomics with graphst. Nature Communications **14**(1), 1155 (2023) https://doi.org/10.1038/s41467-023-36796-3

[10] Codeluppi, S., Borm, L.E., Zeisel, A., La Manno, G., van Lunteren, J.A., Svensson, C.I., Linnarsson, S.: Spatial organization of the somatosensory cortex revealed by osmfish. Nature Methods **15**(11), 932–935 (2018) https://doi.org/10.1038/s41592-018-0175-z

[11] Hunter, M.V., Moncada, R., Weiss, J.M., Yanai, I., White, R.M.: Spatially resolved transcriptomics reveals the architecture of the tumor-microenvironment interface. Nature Communications **12**(1), 6278 (2021) https://doi.org/10.1038/s41467-021-26614-z

[12] Cang, Z., Ning, X., Nie, A., Xu, M., Zhang, J.: Scan-it: Domain segmentation of spatial transcriptomics images by graph neural network. BMVC : proceedings of the British Machine Vision Conference. British Machine Vision Conference **32** (2021)

[13] Ren, H., Walker, B.L., Cang, Z., Nie, Q.: Identifying multicellular spatiotemporal organization of cells with spaceflow. Nature Communications **13**(1), 4076 (2022) https://doi.org/10.1038/s41467-022-31739-w

[14] Xu, C., Jin, X., Wei, S., Wang, P., Luo, M., Xu, Z., Yang, W., Cai, Y., Xiao, L., Lin, X., Liu, H., Cheng, R., Pang, F., Chen, R., Su, X., Hu, Y., Wang, G., Jiang, Q.: Deepst: identifying spatial domains in spatial transcriptomics by deep learning. Nucleic Acids Research **50**(22), 131 (2022) https://doi.org/10.1093/nar/gkac901

[15] Yuan, Z., Zhao, F., Lin, S., Zhao, Y., Yao, J., Cui, Y., Zhang, X.-Y., Zhao, Y.: Benchmarking spatial clustering methods with spatially resolved transcriptomics data. Nature Methods, 1–11 (2024) https://doi.org/10.1038/s41592-024-02215-8

[16] Dilokthanakul, N., Mediano, P.A.M., Garnelo, M., Lee, M.C.H., Salimbeni, H., Arulkumaran, K., Shanahan, M.: Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders. https://arxiv.org/pdf/1611.02648

[17] Jiang, Z., Zheng, Y., Tan, H., Tang, B., Zhou, H.: Variational deep embedding: An unsupervised and generative approach to clustering. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pp. 1965–1972 (2017). https://doi.org/10.24963/ijcai.2017/273

[18] Yang, L., Cheung, N.-M., Li, J., Fang, J.: Deep clustering by gaussian mixture variational autoencoders with graph embedding. In: 2019 International Conference on Computer Vision, pp. 6439–6448. IEEE, Piscataway, NJ

(2019). https://doi.org/10.1109/ICCV.2019.00654 . https://openaccess.thecvf.com/content_ICCV_2019/papers/Yang_Deep_Clustering_by_Gaussian_Mixture_Variational_Autoencoders_With_Graph_Embedding_ICCV_2019_paper.pdf Accessed 2021/6/23

[19] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, Yoshua Bengio: Graph attention networks. International Conference on Learning Representations (2022)

[20] Rebecca, V.W., Nicastri, M.C., Fennelly, C., Chude, C.I., Barber-Rotenberg, J.S., Ronghe, A., McAfee, Q., McLaughlin, N.P., Zhang, G., Goldman, A.R., Ojha, R., Piao, S., Noguera-Ortega, E., Martorella, A., Alicea, G.M., Lee, J.J., Schuchter, L.M., Xu, X., Herlyn, M., Marmorstein, R., Gimotty, P.A., Speicher, D.W., Winkler, J.D., Amaravadi, R.K.: Ppt1 promotes tumor growth and is the molecular target of chloroquine derivatives in cancer. Cancer Discovery **9**(2), 220–229 (2019) https://doi.org/10.1158/2159-8290.CD-18-0706

[21] Wang, X., Allen, W.E., Wright, M.A., Sylwestrak, E.L., Samusik, N., Vesuna, S., Evans, K., Liu, C., Ramakrishnan, C., Liu, J., Nolan, G.P., Bava, F.-A., Deisseroth, K.: Three-dimensional intact-tissue sequencing of single-cell transcriptional states. Science **361**(6400) (2018) https://doi.org/10.1126/science.aat5691

[22] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition, pp. 770–778 (2016). https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

[23] Wolf, F.A., Angerer, P., Theis, F.J.: Scanpy: large-scale single-cell gene expression data analysis. Genome Biology **19**(1), 15 (2018) https://doi.org/10.1186/s13059-017-1382-0

[24] Liu, X., Zeira, R., Raphael, B.J.: Partial alignment of multislice spatially resolved transcriptomics data. Genome Research **33**(7), 1124–1132 (2023) https://doi.org/10.1101/gr.277670.123

[25] Scrucca, L., Fop, M., Murphy, T.B., Raftery, A.E.: mclust 5: Clustering, classification and density estimation using gaussian finite mixture models. The R journal **8**(1), 289–317 (2016)

[26] Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 3319–3328. PMLR, New York, New York, USA (2017). https://proceedings.mlr.press/v70/sundararajan17a.html

[27] Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification **2**(1), 193–218 (1985) https://doi.org/10.1007/BF01908075

[28] Kassambara, A.: Ggpubr: 'ggplot2' Based Publication Ready Plots. (2023). R package version 0.6.0. https://rpkgs.datanovia.com/ggpubr/

[29] Dries, R., Zhu, Q., Dong, R., Eng, C.-H.L., Li, H., Liu, K., Fu, Y., Zhao, T., Sarkar, A., Bao, F., George, R.E., Pierson, N., Cai, L., Yuan, G.-C.: Giotto: a toolbox for integrative analysis and visualization of spatial expression data. Genome Biology **22**(1), 78 (2021) https://doi.org/10.1186/s13059-021-02286-2