Reproducing Results from 'CAST'

Yifan Zhang

School of Mathematical Sciences
Zhejiang University

Overview of 'CAST'

Goal

To integrate and compare spatial omics data across different technologies and modalities.

Application

- Allows spatially resolved differential analysis ($\Delta Analysis$) to pinpoint and visualize disease-associated molecular pathways and cell–cell interactions
- Allows single-cell relative translational efficiency profiling to reveal variations in translational control across cell types and regions
- Many others

Structure of CAST

CAST is composed of three modules: CAST Mark, CAST Stack and CAST Projection:

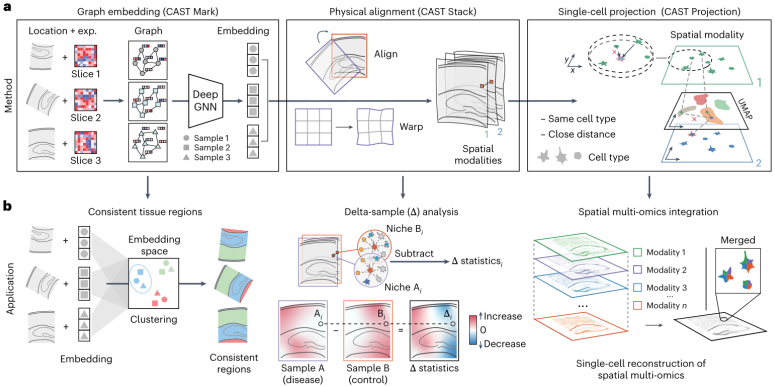


Figure 1: Structure of CAST

Details and Results Reproduction

- CAST Mark
- CAST Stack
- CAST Projection

CAST Mark Algorithm

Goal Generate common embeddings for cells across different samples.

Input Data

Suppose there are M cells and N features (eg. genes) in a sample. The inputs for this algorithm are:

- Spacial coordinates of the cells: $\Psi \in R^{M \times 2}$
- Feature expression matrix: $X \in R^{M \times N}$

Data Preprocessing

(i) Normalize the sum of the raw read counts (eg. gene expression) of each cell to 10^4

(ii) Perform log_2 transformation

(iii) Scale the data without zero-centering

CAST Mark Algorithm

Step 1: Perform Delaunay triangulation for each sample

[Def](Triangulation)
Given a set of points in the plane $P = \{P_1, P_2, \cdots, P_n\}$, if there exists a set of triangles $T = \{t_1, t_2, \cdots, t_m\}$ s.t:

(a) The union of all triangle vertices is P,

(b) For any two triangles t_i and t_j, the edges of t_i and t_j either coincide or do not intersect,

(c) The union of all triangles forms the convex hull of P,

then T is called a triangulation of P.

[Def](Delaunay triangulation)
Let T be a triangulation of P. If for every $t_i \in T$, the circumcircle of t_i does not contain any other points from P, then T is called the Delaunay triangulation of P.

[Prop]
For the two-dimensional case, the Delaunay triangulation exists and is unique.

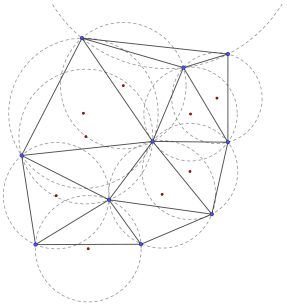


Figure 2: Delaunay Triangulation

Step 1: Perform Delaunay triangulation for each sample

Input

The spatial coordinates of the sample $\Psi \in R^{M\times 2}$

Output

The Delaunay triangulation $G = (V, E)$

(The adjacency matrix of the Delaunay triangulation of the sample $A \in R^{M\times M}$)

Step 2: Generate two augmented views of G

Input The Delaunay triangulation G

Output Two augmented views of G: $G_1 = (\tilde{X}_1, \tilde{A}_1), G_2 = (\tilde{X}_2, \tilde{A}_2)$

Method Apply random node feature masks and random edge masks to the initial graph G.

Goal We want the embedding to tolerate the intrinsic and sample-level stochasticity of gene expression and spatial locations of cells at microscopic scales.

Step 3: Apply CAST mark GNN

- (Optional) Single-layer perception encoder (aims to reduce feature dimension)
- L GCNII layers:

for each layer $l = 0, 1, ..., L-1$, the propogation formula is:

$$H^{(l+1)} = \sigma(((1-\alpha_l)(D^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}})H^{(l)} + \alpha_l H^{(0)})((1-\beta_l)I_N + \beta_l W^{(l)})) \quad (1)$$

Goal

Address the over-smoothing problem caused by traditional GNN architectures.

Using the CAST mark GNN ϵ_θ, we can create node embeddings for the two augmented views: $H_1 = \epsilon_\theta(\tilde{X}_1, \tilde{A}_1), H_2 = \epsilon_\theta(\tilde{X}_2, \tilde{A}_2)$

then we normalize H_1 and H_2:

$$\tilde{H} = \frac{H - \mu(H)}{\sigma(H)M} \tag{2}$$

The objective function is:

$$\mathcal{L} = \|\tilde{H_1} - \tilde{H_2}\|_F^2 + \lambda(\|\tilde{H_1}^T\tilde{H_1} - I\|_F^2 + \|\tilde{H_2}^T\tilde{H_2} - I\|_F^2) \tag{3}$$

Here, $\|\cdot\|_F$ denotes the Frobenius norm.

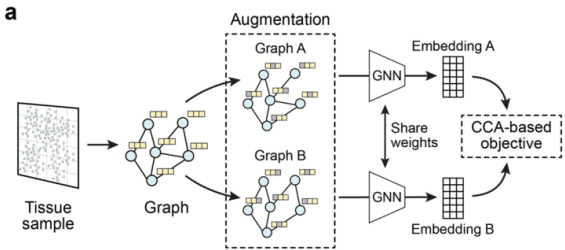


Figure 3: CAST Mark Algorithm

After the training process, the final graph embedding of the original graph G is $H_{init} = \epsilon_{theta}(X, A)$.

CAST Mark Results Reproduction

Apply CAST Mark to the 2,766-gene STARmap PLUS dataset composed of eight coronal brain slices near the hippocampus region (slices S1–S8) from multiple mice with different conditions, ages and strains.

First, we perform Delaunay triangulation on these eight samples(Figure 4):

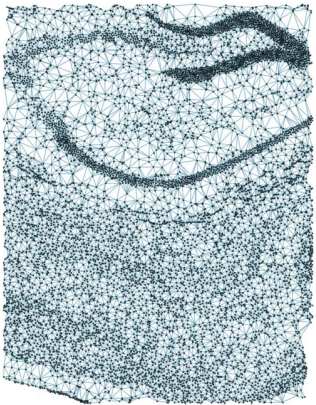
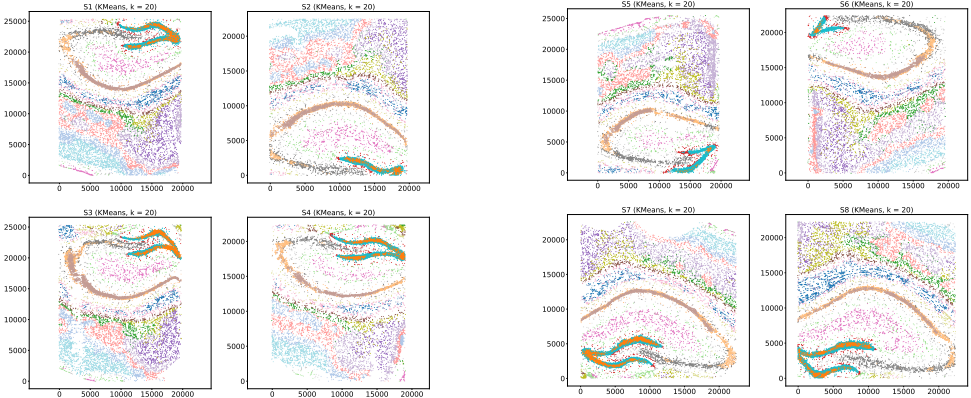


Figure 4: Delaunay Triangulation of Sample S_1

- Apply the CAST mark GNN to the eight samples and train for 20 epochs (in the original paper, the model was trained for 400 epochs)
- Generate a 512-dimensional common embedding
- Perform K-means clustering (k = 20)



CAST Stack Algorithm

Goal Align and integrate different samples into a unified spatial coordinate system.

Input Data Suppose we want to align a query sample with a reference sample. The inputs for this algorithm are:

- The spatial coordinates for both the query sample and the reference sample
- The graph embedding for both the query sample and the reference sample

CAST Stack performs alignment using a rigid alignment phase followed by a non-rigid alignment phase.

Step 1: Rigid alignment

Allows:

- Translation, Rotation, Scaling, Reflection

Disallows:

- Shear mapping

The affine transformation can be written as $\Psi = T_{affine}(\Psi^0) = A\Psi^0 + \mathbf{b}$, where:

$$A = \begin{bmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi \end{bmatrix} \cdot \begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix} = \begin{bmatrix} acos\phi & -dsin\phi \\ asin\phi & dcos\phi \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{4}$$

Basically, there are five parameters to optimize: a, d, ϕ, b_1,and b_2, forming a vector:

$$\boldsymbol{\theta} = \begin{bmatrix} a, & d, & \phi, & b_1, & b_2 \end{bmatrix}^T \tag{5}$$

Thus, the affine transformation can be noted as $T_{affine}(\Psi^0; \boldsymbol{\theta})$. We want to optimize $\boldsymbol{\theta}$ using the gradient descent method.

Suppose there are M cells in the query sample. We define the loss function as the sum of the adjusted Pearson distance J_i between each query cell i and its nearest reference cell:

$$J = \sum_{i=1}^{M} J_i \tag{6}$$

Here, $J_i = max(r) - r_i, i = 1, \cdots, M$, and r_i is the Pearson correlation value between each query cell i's graph embedding and its nearest reference cell's graph embedding.

We use the gradient-descent method for optimization:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_{\boldsymbol{t}} - \alpha \nabla_{\boldsymbol{\theta}} J \tag{7}$$

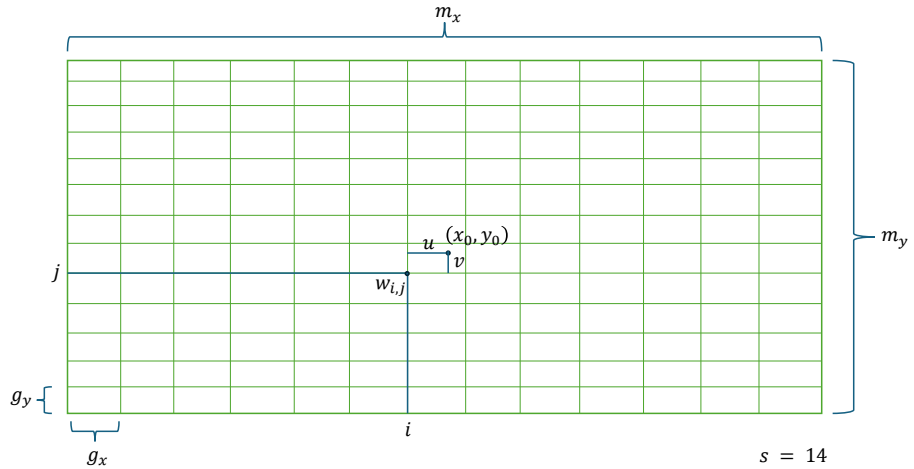
Where α is a weighting parameter of the GD. More specifically, we can apply the chain rule:

$$\frac{\partial J}{\partial \theta_k} = \sum_{i=1}^{M} \left(\frac{\partial J}{\partial \boldsymbol{\Psi}_{\boldsymbol{i}}}\right)^T \frac{\partial \boldsymbol{\Psi}_{\boldsymbol{i}}}{\partial \theta_k} \tag{8}$$

Here, $\nabla_{\boldsymbol{\theta}} J = \left[\frac{\partial J}{\partial \theta_1}, \quad \cdots, \quad \frac{\partial J}{\partial \theta_5}\right]^T$, $\frac{\partial J}{\partial \boldsymbol{\Psi}_{\boldsymbol{i}}} = \left[\frac{\partial J}{\partial x_i}, \quad \frac{\partial J}{\partial y_i}\right]^T$, $\frac{\partial J}{\partial \theta_k} = \left[\frac{\partial x_i}{\partial \theta_k}, \quad \frac{\partial y_i}{\partial \theta_k}\right]^T$.

Non-rigid Alignment

The non-rigid alignment algorithm is based on the B-spline method. First, we generate a mesh grid for the spatial slice:



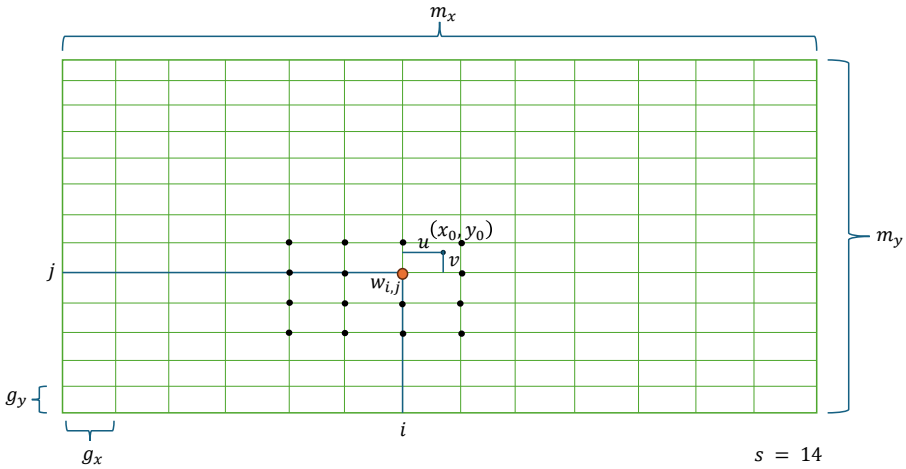
Here, s (the number of control points in each dimension) is given, resulting in $s \times s$ control points $w_{i,j}$. m_x and m_y represent the maximum coordinates of the slice in each dimension, and $g_x = \frac{m_x}{s-1}, g_y = \frac{m_y}{s-1}$ indicate the initial mesh spacings.

The B-spline transformation is written as:

$$\Psi = T_{B-Spline}(\Psi^0) = \sum_{l=-2}^{1} \sum_{m=-2}^{1} B_l(u) B_m(v) w_{i+l,j+m} \tag{9}$$

Where $i = \lfloor x^0/g_x \rfloor, j = \lfloor y^0/g_y \rfloor, u = x^0 - i \times g_x$, and $v = y^0 - j \times g_y$.

The B-spline transformation can be viewed as the weighted average of the 16 control points around the sample point, based on the B-spline basis functions:



Here, the cubic spline basis functions are:

$$\begin{cases} B_{-2}(u) = \dfrac{(1-u)^3}{6}, \\ B_{-1}(u) = \dfrac{u^3}{2} - u^2 + \dfrac{2}{3}, \\ B_0(u) = \dfrac{1}{6}(-3u^3 + 3u^2 + 3u + 1), \\ B_1(u) = \dfrac{u^3}{6} \end{cases} \tag{10}$$

The transformation is non-linear, and since it only depends on the nearest 16 control points, it is regional. Moreover, we have $\sum_{i=-2}^{1} B_i(u) \equiv 1$, thus $\sum_{l=-2}^{1} \sum_{m=-2}^{1} B_l(u) B_m(v) \equiv 1$. This ensures the invariance of the deformation field under rotation.

Since the B-spline transformation depends only on the control points, we train it using the gradient-descent method. The loss function is still $J = \sum_{i=1}^{M} J_i$, and we have:

$$w_{t+1} = w_t - \alpha \nabla_w J \tag{11}$$

Similarly, we can still apply the chain rule:

$$\frac{\partial J}{\partial w_{i,j}} = \sum_{h=1}^{M} \left(\frac{\partial J}{\partial \mathbf{\Psi}_h}\right)^T \frac{\partial \mathbf{\Psi}_h}{\partial w_{i,j}} \tag{12}$$

Here, $\nabla_w J = \begin{bmatrix} \frac{\partial J}{\partial w_{0,0}} & \cdots & \frac{\partial J}{\partial w_{0,s-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial J}{\partial w_{s-1,0}} & \cdots & \frac{\partial J}{\partial w_{s-1,s-1}} \end{bmatrix}$.

CAST Stack Results Reproduction

For results reproduction, we still use the 2,766-gene STARmap PLUS dataset composed of eight coronal brain slices near the hippocampus region from multiple mice with different conditions, ages and strains. Our goal is to align S4 to S1.

We can see that, before alignment, the spatial positions of the cells in the two samples do not overlap.

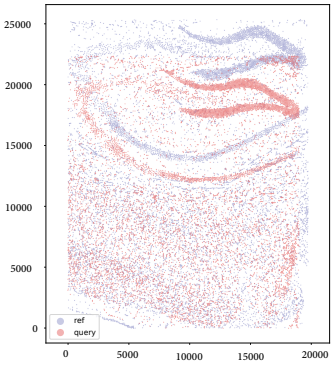


Figure 8: Before Alignment

Rigid Alignment

For the rigid alignment phase, I trained the gradient-descent method for 500 epochs, and obtained the result (Figure 11):
We can see that, after rigid alignment, the spatial positions of the cells in the two samples have a high degree of overlap already.

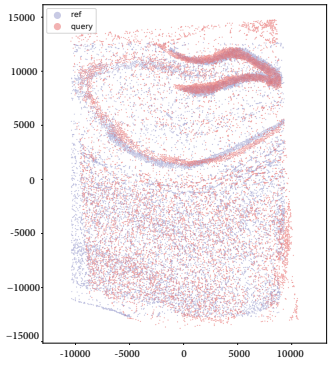


Figure 9: After Rigid Alignment

We can also visualize the changes in the parameters a, d, ϕ, b_1, and b_2, as well as the change in total loss, during the 500 training epochs (Figure 9).

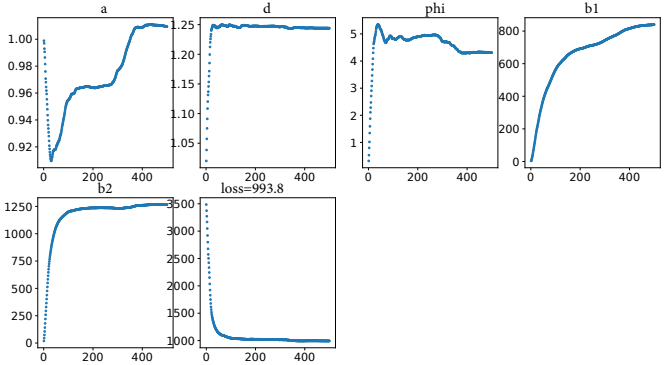


Figure 10: Changes in the Parameters and Total Loss

Non-rigid Alignment

For the non-rigid alignment phase, I trained the gradient-descent method for 400 epochs, and obtained the result (Figure 10):
We can see that, after non-rigid alignment, the spatial positions of the cells in the two samples overlap well.

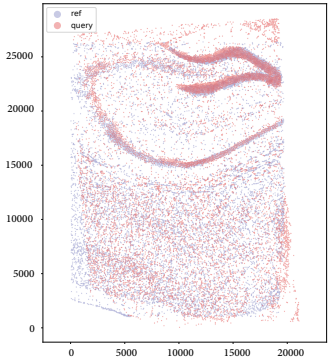


Figure 11: After Non-rigid Alignment

To better visualize the changes in the non-rigid alignment phase (which isn't very obvious), we can plot the cell locations after the alignment with the cell locations before the alignment together in one figure (Figure 12), and minor changes can be discerned.

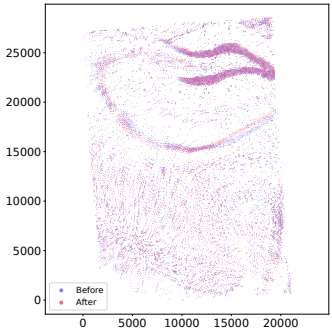


Figure 12: Changes in the Non-rigid Alignment Phase

More Capabilities

- Align samples across multiple different spatial omics technologies
- Align samples with different gene panels
- Align a small, truncated query sample to a large reference sample (Figure 13)
- Search one query sample against large reference atlas datasets
- Perform $\Delta Analysis$ to pinpoint disease-associated molecular pathways

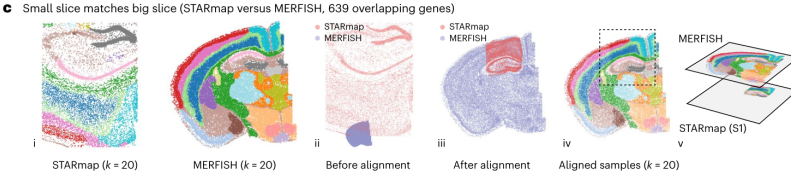


Figure 13: Align samples with different sizes

CAST Projection Algorithm

Goal Integrate samples with different spatial omic modalities.

Input data Suppose we want to project a sample from one modality (query sample) onto a sample of another modality (reference sample). The inputs for this algorithm are:

- The spatial coordinates for both the query sample and the reference sample
- The cell types for all the cells included
- The features (eg. gene expression) for all the cells included

Note that the query sample and the reference sample should already be in the same spatial coordinate system, since we have performed the CAST Stack algorithm.
If we don't have the cell type label, the k-means clustering results produced by CAST Mark can serve as cell types.

Step 1 Project the features of the cells in both the query sample and the reference sample into a common low-dimensional space using a sequential combination of Combat and Harmony integration.

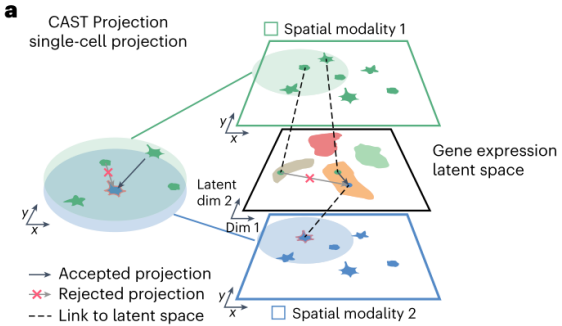


Figure 14: CAST Projection Algorithm

Cosine distance is used to measure the similarity of cell features in the integrated embedding:

$$d_{cos} = 1 - \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} \tag{13}$$

Clearly, cosine distance is concerned solely with the direction of vectors, ignoring their lengths.

However, only relying on cosine distance in the latent space to project cells from different samples does not yield ideal results; we also need to incorporate information from physical coordinates.

For a given reference cell, CAST first identifies the candidate query cells within a radius r of the reference cell.

CAST calculates the cell-type-specific cell average distance $d_{average}$ based on the Delaunay triangulation graph.

- Default: $r = 2d_{average}$
- In AD samples: $r = 1.5d_{average}$
- In RIBOmap-STARmap: $r = 3d_{average}$

Among the candidate query cells, CAST identifies the cell with the closest cosine distance to project.

CAST Projection Results Reproduction

Apply CAST Projection to a brain sample whose transcriptomes and translatomes were profiled respectively with STARmap and RIBOmap technologies at single-cell resolution (Figure 15).

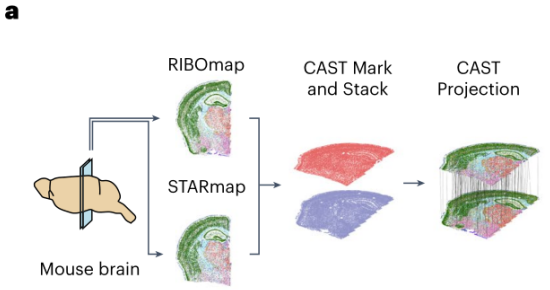


Figure 15: CAST Projection Demo

The confusion matrix, sensitivity and precision of CAST Projection are shown in Figure 16:

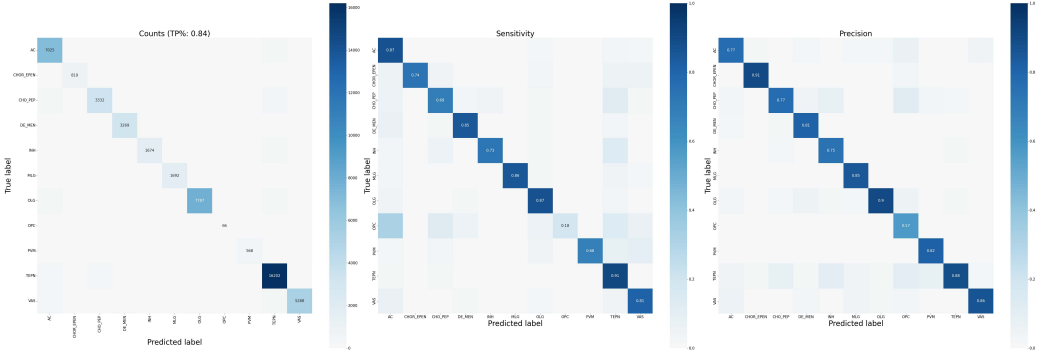


Figure 16: Performance of CAST Projection

Finally, we can visualize the result:

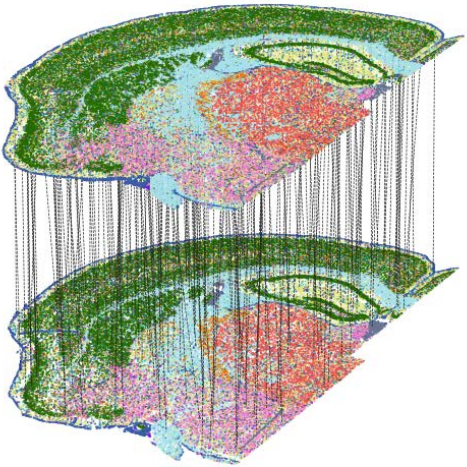


Figure 17: CAST Projection Result

More Capabilities

- Analyze spatially resolved single-cell translation efficiency:

After CAST Projection, each cell contained both RIBOmap and STARmap measurements. RIBOmap reflects translation levels, while STARmap reflects transcription levels.
For a given gene i and a cell j, we can define single-cell relative translation efficiency as:

$$scRTE = zscore(log_2(\frac{RIBO_{i,j}}{STAR_{i,j}})) \tag{14}$$

We can visualize the result (Figure 18):

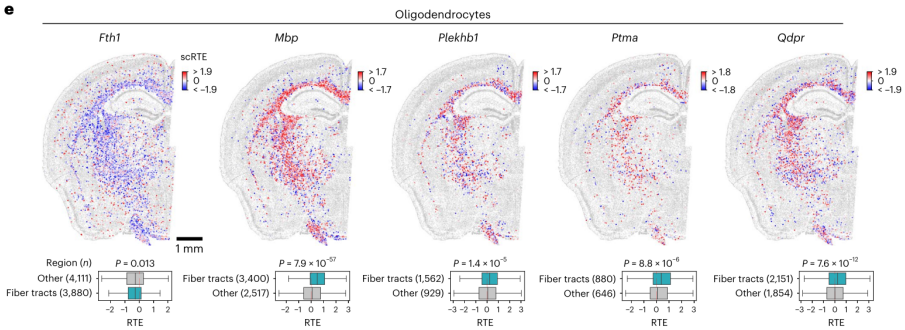


Figure 18: scRTE for Different Genes in Oligodendrocytes

Conclusion

What we have learned about CAST:

- A comparative analysis tool that can search, match and visualize both similarities and differences of molecular features in space across multiple samples
- Based on deep graph neural network, along with many other mathematical methods
- Has numerous applications

What we have done:

- Understood the framework of CAST
- Reproduced basic results generated by CAST Mark, CAST Stack, as well as CAST Projection