Genetic Tool to Cell Type Bayesian Mapper (GT2CT Bayesian Mapper)

Nikhil Karthik and Chaitali Khan

May 30, 2025

Genetic Tool to Cell Type Bayesian Mapper (GT2CT Bayesian Mapper) is a python based toolkit for finding the likely fractions of brain cell-types (class, subclass, supercluster or cluster) present in the GFP/RFP positive cells tagged by genetic tools.

In the note below, we will use "VISp" to specify a region of brain, but it is just a placeholder for any choice of brain region parcellation. We will use the term "subclass" to specify cell-type, but it is a placeholder for cell-type at class, subclass, supercluster or cluster. The tool kit is written in a generalized way to perform analysis in any brain region and cell-type classification levels. But, we have only tested the algorithm in the VISp at subclass level at the time of writing.

1 Mathematical background

1.1 Notation

Let $\mathbf{x} = (x, y, z)$ be the three-dimensional CCF coordinates in the VISp area. The brain slices are labeled by constant z values. Let S be the set of subclasses present in VISp area, and let N be the size of this set of subclasses. That is, for VISp, $S = \{$L6 CT CTX Glut, Pvalb Gaba,...$\}$ with 28 other subclasses in the set. Note that this information on cell-type is inferred from the MERFISH cell data.

Next, we have the GFP/RFP positive cells tagged by a gene tool. Let K be the target specificity of the gene tool. Either one knows how many K cell types can be tagged by the genetic tool, or one knows a prior distribution over K for gene tools used. For now, we assume that the value of K is known, and later we will use prior knowledge that K is small. Given the value of K, there are $\binom{N}{K}$ subclass combinations possible. Let $\mathcal{K}$ is the set of these $\binom{N}{K}$ subclasses K-plets, and let its members be k. These K-plets k are, for example, such as $k_1 =$ (Oligo, Astro, Sst Gaba), $k_2 =$(L5 ET CTX Glut, Sncg Gaba, Microglia NN), etc., for $K = 3$.

1.2 Probability distributions

Let $P(s|\mathbf{x})$ be the probability to find a cell of type $s \in S$ at position $\mathbf{x}$. The net probability to find any of the subclasses at a location $\mathbf{x}$ should be one; that is

$$\sum_{s \in S} P(s|\mathbf{x}) = 1. \tag{1}$$

Let $P'(k|x)$ be the probability to find a subclass K-plet $k \in \mathcal{K}$ at CCF location x. This is related to the distribution of individual subclass $P(s|\mathbf{x})$ as

$$P'(k|\mathbf{x}) \propto \sum_{s \in k} P(s|\mathbf{x}), \tag{2}$$

with the proportionality constant that correctly normalizes $P'(k|\mathbf{x})$. We also want the spatial distribution of a K-plet k given by $q(\mathbf{x}|k)$. This is given by the Bayes theorem to be

$$q(\mathbf{x}|k) = \frac{P'(k|\mathbf{x})}{\sum_{\mathbf{x}'} P'(k|\mathbf{x}')}, \tag{3}$$

assuming a uniform prior distribution over position. For example, let us say that we know that a genetic tool specifically targets k =(L5 ET CTX Glut, Sncg Gaba, Microglia NN). Then, the distribution $q(\mathbf{x}|k)$ gives the spatial distribution to find one of the members of k to be at various x.

Let $Q_{\rm GFP}(\mathbf{x})$ be the spatial distribution of GFP/RFP cells. If one had a really high resolution image of GFP/RFP cells, and one knew with 100% confidence which were GFP/RFP cells versus the background, then we can still describe the cells through a distribution $Q_{\rm GFP}(\mathbf{x})$ that are non-zero only at exact locations of the GFP cells. In general, this is not the case, and we take the approach to assign only a continuous spatial distribution $Q_{\rm GFP}(\mathbf{x})$ that there is a GFP/RFP cell at various $\mathbf{x}$. We will define our procedure to assign $Q_{\rm GFP}(\mathbf{x})$ based on intensity distribution later in this note.

1.3 Mathematical statement of the algorithm

The basis of our algorithm is the following. Given $Q_{\rm GFP}(\mathbf{x})$ of the GFP/RFP cells via the STPT images of the brain, given a spatially smooth distribution $P(s|x)$ of the celltypes inferred from MERFISH data, and given that the gene tool can tag only certain cell types from K-plets k all over VISp, we find the single best K-plet k whose $q(\mathbf{x}|k)$ is the most similar to $Q_{\rm GFP}(\mathbf{x})$.

Such a measure of similarity between two probability distributions p and q is given by the cross-entropy $\mathcal{H}(p,q)$ which is a minimum only if $p=q$. For our case, we want to find k such that

$$\mathcal{H}(k) = -\sum_{\mathbf{x}\in \rm VISp} Q_{\rm GFP}(\mathbf{x}) \log\left[q(\mathbf{x}|k)\right], \tag{4}$$

is minimized among all $k \in \mathcal{K}$. Let $k=k^*$ be the optimal K-plet. Then, given the assumption/model that the gene tool has specificity of K cell types, the expected fractions f_s of subclasses $s \in k^*$ present all over VISp is

$$f_s = \sum_{\mathbf{x}\in \rm VISp} Q_{\rm GFP}(\mathbf{x}) \frac{P(s|\mathbf{x})}{\sum_{s'\in k^*} P(s'|\mathbf{x})}, \quad \text{for} \quad s \in k^*. \tag{5}$$

For $s \notin k^*$, $f_s = 0$. The assumptions of specificity K and the optimal k^* from $\mathcal{K}$ are implicit, and it is to be understood that f_s is the above equation is actually $f_s\left(k^*(K)\right)$.

1.4 Model averaging

The above step concludes the algorithm. But, one could go further to alleviate the model dependence on the choice of gene tool specificity K by a weighted average of $f_s(k^*)$ for different K. Studies are needed on how best to perform the weighted average, but for this challenge, we do a simple model averaging as

$$f_s = \frac{1}{K_{\rm max}} \sum_{K=1}^{K_{\rm max}} f_s\left(k^*(K)\right), \tag{6}$$

for $K_{\rm max}=4$.

2 Implementation of Algorithm

2.1 Estimating $P(s|\mathbf{x})$ using multilayer perceptron

2.1.1 Details of MLP

The distribution $P(s|\mathbf{x})$ is a multi-celltype position-based classification problem. We wanted to design a protocol which estimated $P(s|\mathbf{x})$ that can be used in downstream analysis. We modeled $P(s|\mathbf{x})$ using a simple multilayer perceptron (MLP) with the input feature being a scaled version $(\tilde{x},\tilde{y},\tilde{z})$ of three coordinate components (x,y,z) – more on scaling function below. The output of the MLP is a 28-way log-probabilities $\log(P(s|\mathbf{x}))$ for the 28 VISp subclasses. Between the input and out layers, we used four hidden layers with 64 hidden units each with ReLU activation functions. This simple setup was found sufficient, and also essential given that the tool should be usable on a CPU.

2.1.2 Increasing training data with VISp neighborhood

For training the weights w of the MLP, we used all the MERFISH data (coordinates and their subclasses) in the VISp area. We brought the MERFISH cells on left section into the right using reflection symmetry of CCF coordinates. This brought the net trainable data in VISp to $\sim$ 62000. Next, we increased the trainable data using cells in VISp's neighborhood. For this, we found the centroid $(x_{\rm centroid}, y_{\rm centroid}, z_{\rm centroid})$ of the VISp volume, and quantified the net lateral and longitudinal extend of the VISp area as

$$\begin{aligned} \sigma_{xy} &= \langle\sqrt{(x-x_{\rm centroid})^2+(y-y_{\rm centroid})^2}\rangle_{\rm VISp}, \\ \sigma_z &= \langle\sqrt{(z-x_{\rm centroid})^2}\rangle_{\rm VISp}, \end{aligned} \tag{7}$$

with $\langle\ldots\rangle_{\rm VISp}$ is an average over VISp space. Then, we included MERFISH cells in the neighborhood of VISp that were within a radius of $1.5\sigma_{xy}$ from the VISp's centroid z-axis. This increased the trainable MERFISH cell data to 211052. This way, we allowed the data slightly outside of VISp to constrain the $P(s|x)$ in VISp. In order to regularize the numbers being fed to the MLP, the we input the scaled CCF coordinates

$$\tilde{x} = \frac{x-x_{\rm centroid}}{\sigma_{xy}}; \tilde{x} = \frac{y-y_{\rm centroid}}{\sigma_{xy}}; \tilde{x} = \frac{z-z_{\rm centroid}}{\sigma_z}. \tag{8}$$

2.1.3 Training details

We trained the MLP over 20 epochs with 90%-10% train-test split with a batch size of 128. For the output of MLP to be interpretable as log-probabilities, we used cross-entropy loss. The model got trained within 5-6 cycles, and the cross-entropy loss plateaued around 1.68 on test MERFISH data, and gave a top-3 classification success rate of 76%.

2.2 Converting STPT data into GFP/RFP probability distributions

2.2.1 Choosing ideal CCF slices in STPT data

We worked with the STPT data registered to 25 micron CCF. First, we chose the optimal z-slices of the 25 micron CCF that are closest to the MERFISH cells that are only available in about every 200 microns. We wanted to reduce the effect of subdued constraints on the neural network in CCF regions where MERFISH data are poor or absent. We included only STPT CCF slices where the average distance to the neighboring MERFISH cell is less than 0.03 micron within 1-standard deviation.

2.2.2 Thresholding STPT image

Next, we removed the diffuse background noise from the STPT data in the chosen color channel. We first set z-slice dependent intensity thresholds $I_{\rm thresh}(z)$ such that they encompass 98-percentile of the voxels that contained a nonzero signal in a z-slice. From an examination of OME Zarr image, this choice approximately reflected the manually chosen thresholds so that one could just see the individual GFP/RFP cells in many of the cases. To avoid bias and to avoid sampling from z-slices that were dimmer compared to other areas of VISp, we placed a uniform global threshold $\bar{I}_{\rm thresh} = {\rm Median}(I_{\rm thresh}(z))$. Furthermore, to accommodate left-right asymmetry in some STPT samples, we allowed left-right dependence of this global $\bar{I}_{\rm thresh}$. We subtracted this left-right specific global thresholds from all the voxels, and set any subtracted intensity less than 0 to be 0. Since we did not wish to have few outlier bright GFP/RFP cells to be weighted more in our analysis, we placed an upper cutoff of 2000 on the threshold subtracted voxel intensities.

2.2.3 Constructing GFP spatial probability distribution

After the above thresholding procedure, let these new subtracted intensities be $I_{\rm subtracted}(\mathbf{x})$ at various CCF coordinates/voxels x. We constructed a probability distribution $Q_{\rm GFP}(\mathbf{x})$ to find a GFP cell at $\mathbf{x}$ as

$$Q_{\rm GFP}(\mathbf{x}) \equiv \frac{I_{\rm subtracted}(\mathbf{x})}{\sum_{\mathbf{x}'\in{\rm VISp}} I_{\rm subtracted}(\mathbf{x}')}. \tag{9}$$

We use this estimated GFP spatial distribution in Eq. (4) and Eq. (5). As a practical matter, we sampled $N_{\rm samp} = 80000$ positions $\mathbf{x}_{\rm samp}$ using distribution $Q_{\rm GFP}(\mathbf{x})$ to evaluate weighted sums such as Eq. (4) and Eq. (5) as a simple Monte Carlo sampled averages.

2.3 Computing fractions of cell types

2.4 Choice of values of typical gene tool target specificity K

In the last step, we are fitting the best set of K subclasses that we assume the gene tool targets, and hence are likely to explain the GFP distribution over all location in VISp. What values of K should one use? Here, we need to make use of prior information that gene tool is likely to act specifically on few subclasses. From Figure-4 of Ben-Simon et al (2025), the value of K are typically around 3 to 4 cell types with substantial fractions for many of the gene tools. Thus we scanned over values of K between 1 to 4. The choice of K, in the end, is a trade-off between bias versus noise.

2.4.1 Computing cross-entropy between subclass K-plet distributions and GFP distributions

This final step implements Eq. (4) and Eq. (5). The distribution $q(x|k)$ determined from Eq. (3) using the trained MLP implementation of $P(s|x)$. We evaluate Eq. (4) using sampled GFP coordinates $\mathbf{x}_{\rm samp}$ drawn according to $Q_{\rm GFP}$ (denoted $\mathbf{x}_{\rm samp} \sim Q_{\rm GFP}$) as

$$\mathcal{H}(k) = -\frac{1}{N_{\rm samp}} \sum_{\mathbf{x}_{\rm samp} \sim Q_{\rm GFP}} \log\left[q(\mathbf{x}_{\rm samp}|k)\right]. \tag{10}$$

As we noted, there are $\binom{N}{K}$ choices for K-plets k that we assume the gene pool can potentially target. The number of such K-plets become large for $K \geq 4$ even for $N = 28$, and hence not feasible to find the value of $\mathcal{H}(k)$ for all of them. Therefore, we follow a heuristic procedure to include only the top-15 subclasses ordered by their cross-entropies (the above equation with $K = 1$) as an effective upper cutoff on N, and allow only K-plets formed out of these top-15 subclasses. We sort the values of $\mathcal{H}(k)$, and find the best possible K-plet k^* which minimizes $\mathcal{H}$.

2.4.2 Computing cell fractions f_s

We find the fractions of celltypes s that are contained with the best K-plot k^* using Eq. (5) implemented using the sampled points $\mathbf{x}_{\rm samp}$ as

$$f_s = \frac{1}{N_{\rm samp}} \sum_{\mathbf{x}_{\rm samp} \sim Q_{\rm GFP}} \frac{P(s|\mathbf{x}_{\rm samp})}{\sum_{s' \in k^*} P(s'|\mathbf{x}_{\rm samp})}, \quad \text{for} \quad s \in k^*. \tag{11}$$

Our algorithm in the Jupyter notebook outputs these fractions in its penultimate cell for a given choice of K. For this data challenge, we repeated the last two steps for choices of $K = 1, 2, 3$ and 4, and performed a model averaging of f_s using Eq. (6).