An Approach for Adding Morphological Insight to Spot Transcriptomics

For the consideration of Miltenyi Researchers

Tempest Plott 03-05-2025

Agenda

3 Personal Introduction

5 Why morphology?

6 Why machine-learning?

7 Combining insights with a spot transcriptomic platform

13 Process and Results Deeper Dive

10 Conclusions

11 Questions

Questions are welcome at any time!

Hi, I'm Tempest!

I have a decade of experience in biotech. For the last five years, I have worked as a team member to develop a platform that unleashes the potential of high content imaging by empowering biologists to generate and utilize machine learning insights.

I've left the resources for you to follow along with this demo project HERE IN THIS GITHUB LINK

Let’s get started!

The power of cell morphology...

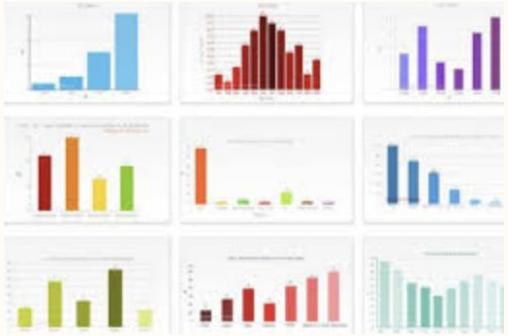
Recognize, Model, and Quantify:

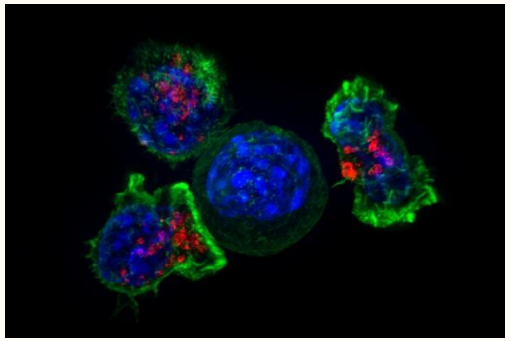
- Cell to cell interactions - Complex behavior - Subtle changes

Harness:

- The intuition of expert biologists - Flexibility

CAR-T Cells Attack A Tumor Cell, or Seeing Is Believing:





[1]

...meets the power of machine learning.

Liberate Scientists:

- The heavy lifting is done by the computer - Confidence is built via feedback loop
- Scientists can focus on the big picture
- Combine the desire for precision with the room to be surprised

[2]

The Demo Project

Set up [3]

We will use a publicly available murine brain tissue slide with spot transcriptomics data in a pretend narrative for this demonstration.

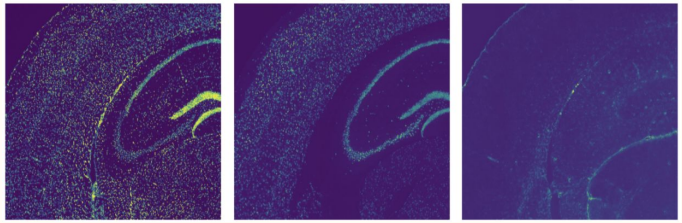
Let's pretend this mouse was part of an Alzheimer's model, and was given a treatment that the researchers suspect reduces plaque in glial cells. It is hypothesized that if this does occur, it will likely be because of upregulation of the gene Ptgds, which among other things, inhibits platelet aggregation.

To investigate the effects of the treatment on the glial cells, the researchers need to examine that population of cells closely and see if the treatment affects all glial cells across the brain equally.

The Data

The 10X Genomics Visium platform is similar to the Miltenyi MACSima platform.

Here is the publicly available histology slide, which we will artificially add a morphological stain to.

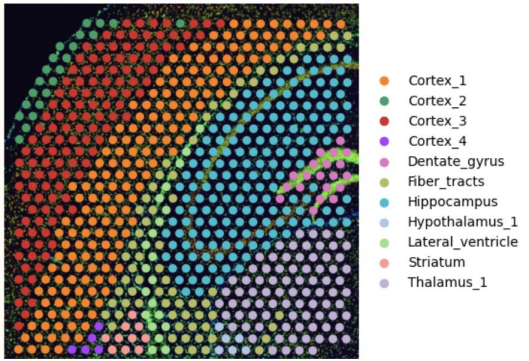


DAPI anti-Neuronal anti-Glial

Histo section of one mouse brain, with WTS spot transcriptomics and 3 stains.

“Phalloidin”

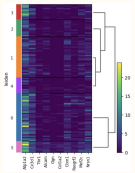
Initial Transcriptomic Output



[4]

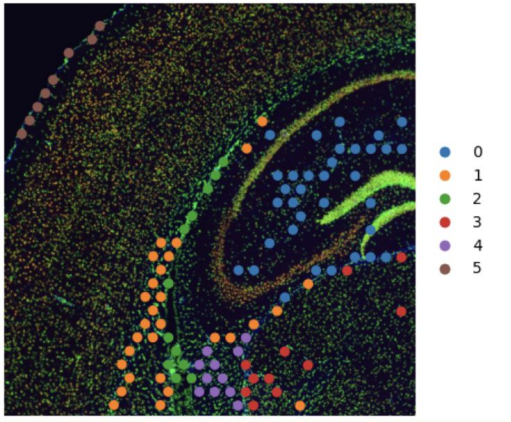
The Issue

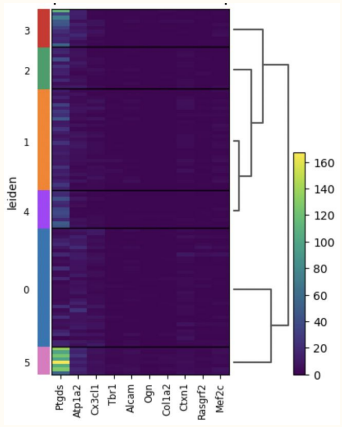
Transcriptomics does not do a fantastic job separating the glial cell sub-populations. We want to know if we can use ML to study the morphology of glial cells and better separate sub-clusters.

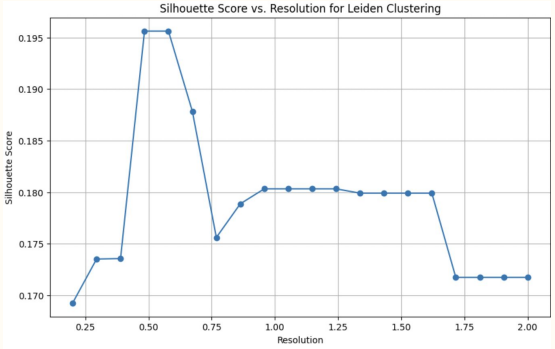


< Visualize with top gene removed

Glial Cell Sub-clustering:

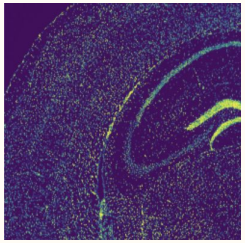


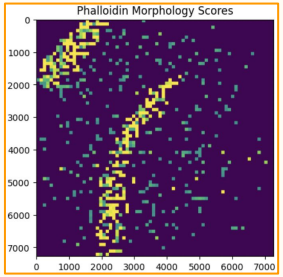


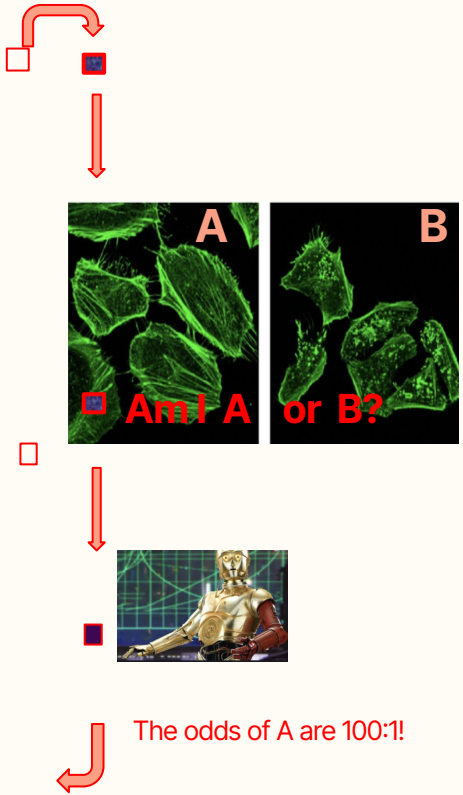


The Solution

Generate predictions of target cytoskeletal morphologies across the image and observe differences among glial cell populations.







Step 1: Break image into tiny pieces (single cell sized)

Step 2: Train a model to recognize target single-cell cytoskeletal morphologies

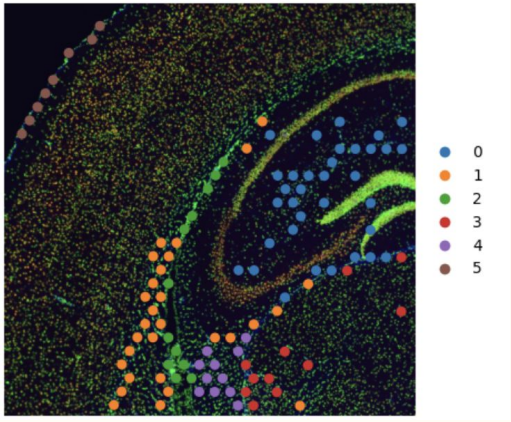
Step 3: The model reports the probability of the presence of the target morphology for each piece

Step 4: We store these prediction values as an image so that they can play well with the rest of the data in the experiment. [5]

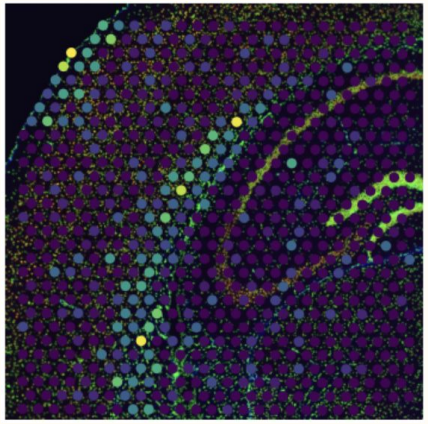
The Result - there was more to the story!

We see that the morphology which correlates with increased target gene expression is present in two areas of the brain, and three of the 5 glial sub-clusters. The treatment is not only affecting the expression profile, and the effect is most pronounced in two regions of the brain.

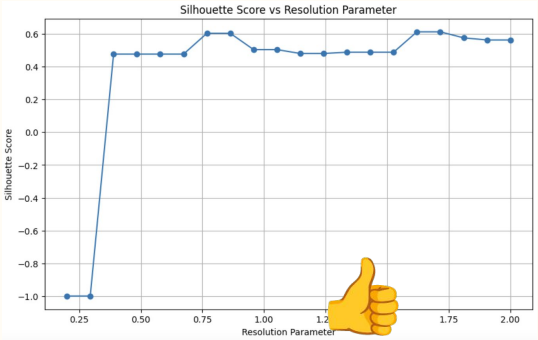
Transcriptomic Clustering:



Morphology finder:



The morphology clustering is much stronger.



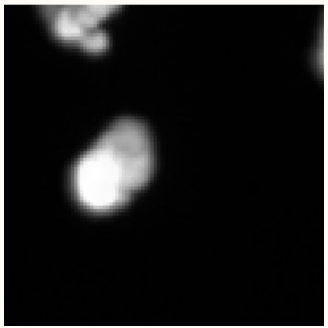
Process and Results Deeper Dive

The Data

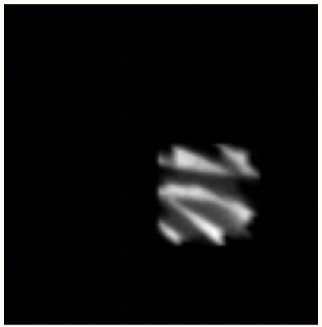
The example “phalloidin” image contains two textures which represent different cytoskeletal morphologies. It also contains some sporadic “imaging artifacts”.

These features are small- the textures are about the size of one nucleus in the histo slide.

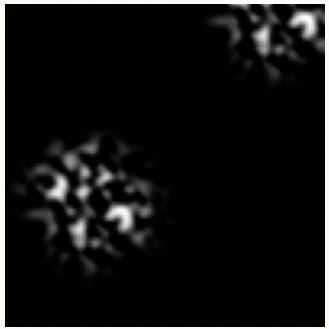
101 × 101 pixel crop of DAPI stained nuclei:



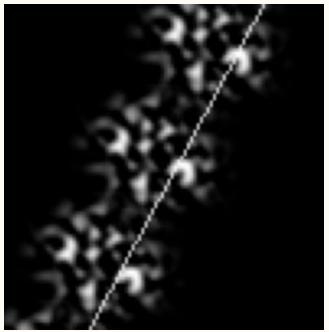
101 × 101 pixel crop of cyto morpho #1:



101 × 101 pixel crop of cyto morpho #2:



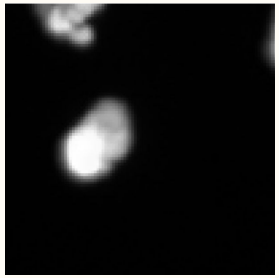
101 × 101 pixel crop of artifact and CM#2:



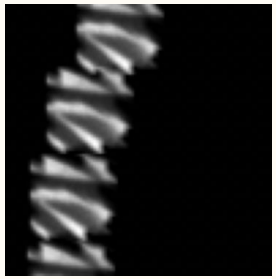
The Data

Comparison to real-world

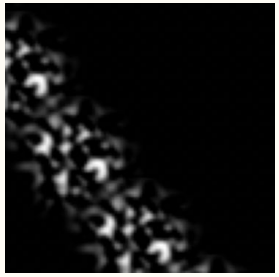
101 × 101 pixel crop of DAPI stained nuclei:



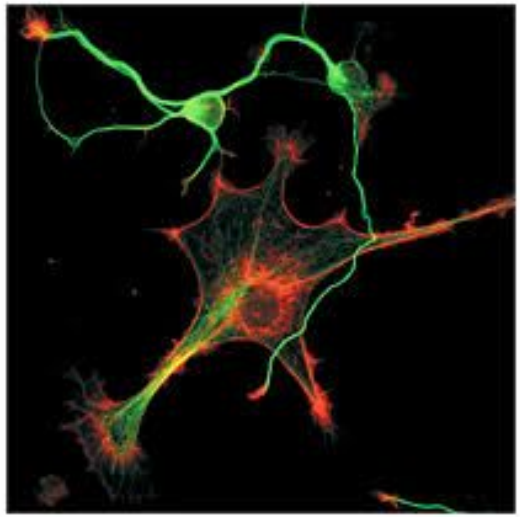
101 × 101 pixel crop of cyto morpho #1:



101 × 101 pixel crop of cyto morpho #2:



Example of two different phalloidin (red) morphologies in glial and neuronal cells. [6]



The training set

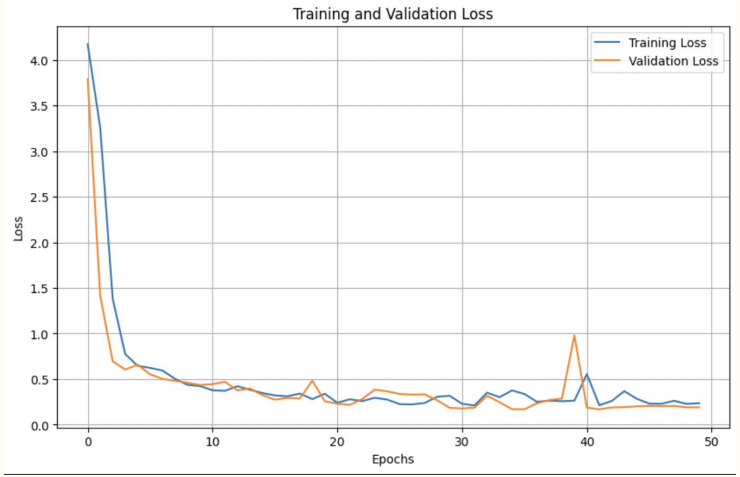
25 examples of the target morphology (“positive”) were collected from the entire image. 20 examples of the negative morphology were similarly collected, with some containing the thin line imaging artifact. 5 examples of empty tiles were also included as belonging to the negative morphology.

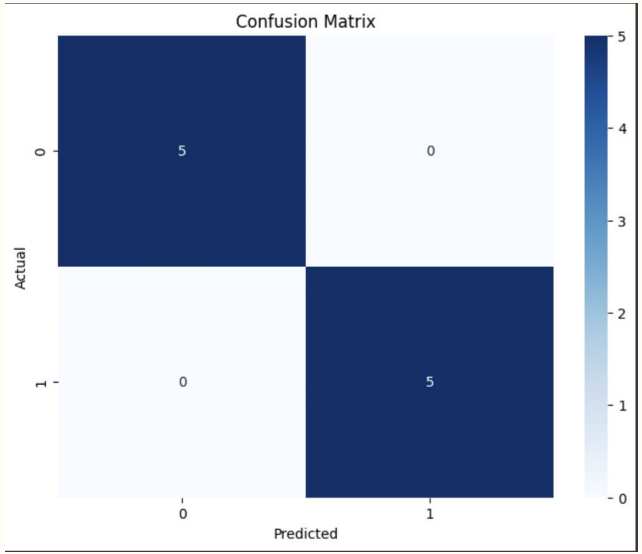
All of these tiles were then subjected to random rotation to ensure that they could not be simply memorized, with the caveat that rotation angles resulting in erroneous empty images were rejected.

The order of training tiles was then shuffled to stop the model from memorizing order information.

The CNN Model

Final model performs well on unseen data, and does not show evidence of overfitting.





The CNN Model

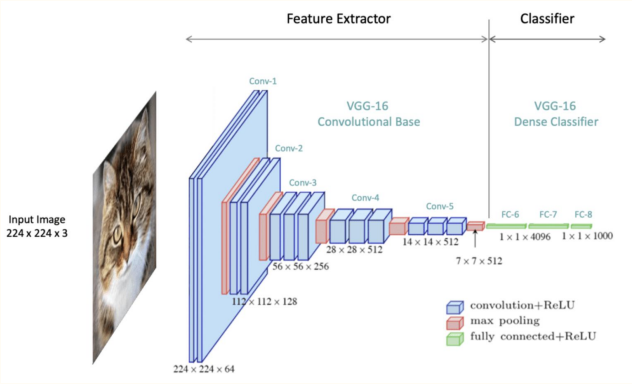
Class weights (equal), 5-fold Cross Validation, Early Stopping, and Data Augmentation per batch were used. 50 epochs were used with patience 20 for val_loss.

```python
def create_morecomplex_cnn_model(input_shape):
    model = models.Sequential()
    model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(128, (3, 3), activation='relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(128, (3, 3), activation='relu'))
    model.add(layers.Flatten())
    model.add(layers.Dense(128, activation='relu'))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(1, activation='sigmoid'))  # Binary classification output

    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    return model
```

What is a Convolutional Neural Network?

CNN architecture works by passing an input image through a series of convolutional layers, each trained to perform mathematical operations which detect abstract features of an image.These layers are followed by pooling layers that downsample the spatial dimensions, reducing complexity but retaining essential information. As the image passes through deeper layers, the network learns more complex and abstract features. Finally, the extracted features are fed into fully-connected layers for classification.



How well does this image match the real cats in your training set of 50 photos of cats and 50 drawings of garfield?

What is a Convolutional Neural Network? (cont.)

The features are abstract in the sense that the user does not directly control them - the model learns to optimize them from comparing its final results to an answer key provided by the human.

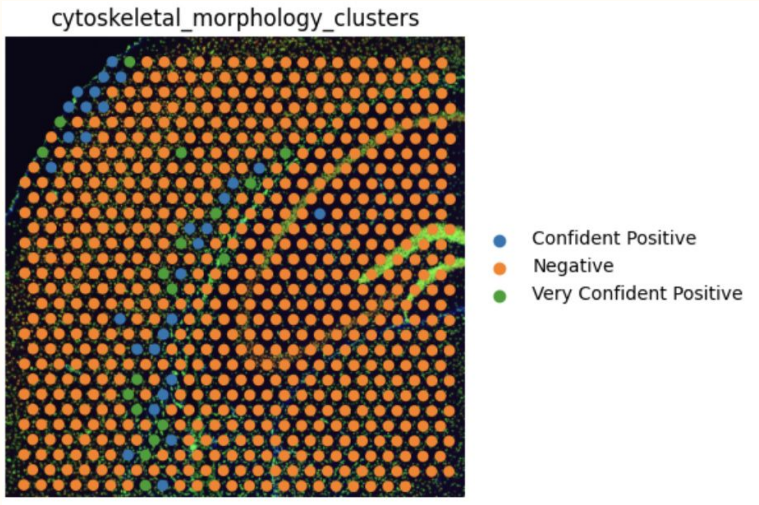
This is perhaps analogous to PCA/UMAP/tSNE. What I mean by this is that the “units” and exact mathematical operations are essentially meaningless. What is important is the final understanding of the overall similarity with the intended result.

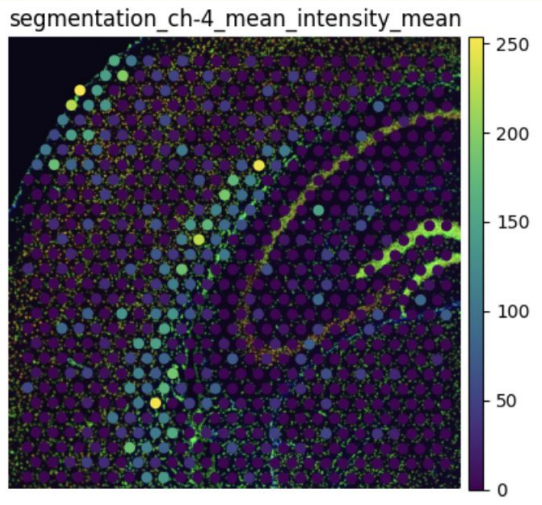
You can’t interpret the meaning of each layer of the CNN, but you can use the final result.

For the PCA analogy, you can imagine performing PCA on all the pixels from an image and then comparing your final plot to a different PCA plot. Your task is then to judge how likely it is that both plots came from the same image. You wouldn’t need to truly understand what is in the photo to judge similarity.

The Results

Clustering again based on morphology score shows an interesting step function in confidence level within the positive prediction class.





Conclusion

Conclusion

This was a small taste of what ML can do to take advantage of information-rich histology slides.

This approach is powerful and fast (takes 7 minutes to process the histo slide and 5 minutes to train the model once examples have been labelled by a scientist.)

This approach is flexible, allowing for multiple morphology stains, different sized tiles to learn to model larger interactions, or perhaps even learn smaller features if your microscopic resolution is strong and the interest is there.

This approach does not require complicated back-end support, needing only simple packages to run.

Conclusions - Caveats and Potential Improvements

This example used demonstration data that had a known ground truth (I created the “phalloidin” image.) An improvement to this demo would include generating a mock phalloidin image from genuine glial and neuronal cells stained with phalloidin, but unfortunately I could not find such a dataset to borrow from and resorted to two textures from the image software Seashore.

Some features in real-life data are difficult to classify. In those cases, further training from data from many experiments, more complex model architectures, and pre-processing the image can help. ML can’t do everything, but there’s a lot we can do to help it along when it struggles. I believe it is worth the investment to develop strong models to help your researchers with whatever is too difficult or time-consuming for a human to quantify.

Questions?