# Generative Model for Imputing Imaging Mass Spectrometry from Serial Two-Photon Tomography

Kevin Yang
BC Cancer
675 W 10th Ave, Vancouver, BC
kyang@bccrc.ca

## Abstract

*The ABSTRACT is to be in fully justified italicized text, at the top of the left-hand column, below the author and affiliation information. Use the word "Abstract" as the title, in 12-point Times, boldface type, centered relative to the column, initially capitalized. The abstract is to be in 10-point, single-spaced type. Leave two blank lines after the Abstract, then begin the main text. Look at previous CVPR abstracts to get a feel for style and length.*

## 1. Background

Cancer is a complex disease driven by genetic mutations. Originating from a single mutated cell, subsequent rounds of proliferation and additional mutations ultimately give rise to the tumour (Nowell, 1976). This evolutionary process fosters tumour heterogeneity, producing genetically distinct populations of cells known as clones. Notably, these clonal populations differ from one another in their genetic makeup and have varied responses to treatment. Therefore, investigating tumour heterogeneity can direct treatment decisions to combat metastasis and recurrence, ultimately improving prognosis.

However, cancer is not exclusively made up of malignant cells, but also of surrounding normal cells (Junttila and de Sauvage, 2013). This mixture of cell types is called the tumour microenvironment (TME). Interactions between malignant and non-malignant cells within the TME can provide insight into the mechanisms that promote cancer growth (Egeblad et al., 2010); this in turn can inform and advance drug development. To fully understand the TME, a comprehensive picture of the morphology and the molecular profile of the tumour must be obtained (Heindl et al., 2015). The introduction of powerful imaging techniques has enabled detailed illustrations of morphology and protein abundance profiles for tumors (Bressan et al., 2021). In particular, serial two-photon tomography (STPT) uncovers tumour morphology, while imaging mass spectrometry reveals protein abundance inside the TME. These two imaging techniques are introduced in the following two subsections.

### 1.1. Serial Two-Photon Tomography (STPT)

STPT is an automated imaging technique that produces high-throughput and high-fidelity imaging by integrating two-photon microscopy and tissue sectioning (Taranda and Turcan, 2021). The general workflow is as follows. First, place the specimen on the XYZ stage under the objective of a two-photon microscope; second, image an optical section as a mosaic of fields of view; third, cut off a slice of tissue using a vibrating blade; repeat.

Imaged portions of the specimen is tens to hundreds of microns below the surface of the tissue (Amato et al., 2016). As a result, the imaged tissue will remain in a pristine state, unaffected by deformations from mechanical sectioning. Moreover, each serial section is imaged before being sliced, allowing for near perfect alignment of imaged tissue sections. Finally, STPT is an automatic procedure which does not require human intervention, greatly mitigating labor costs (Ragan et al., 2012). After imaging is completed, algorithms can reconstruct a 3D image from the 2D image slices (González-Solares et al., 2021). These 3D images uncover salient morphological features which can be used in drug development and TME analysis.

### 1.2. Imaging Mass Spectrometry (IMC)

IMC is an imaging technique that can analyze the protein abundance of a specimen at single-cell resolution. The general workflow is as follows. First, an antibody-stained tissue sample is ablated using a UV laser producing a plume of particles. Next, these particles are sent to a mass cytometer to undergo ionization where cell-specific signals are captured (Devine and Behbehani, 2021).

The ability of IMC to resolve proteomic features of individual cells is crucial, since the TME is comprised of a mixture of cell-types, each varying in their protein abundance (Spitzer and Nolan, 2016). Additionally, IMC conveys spatial relationships between individual cells. This further increases its utility in TME analysis, since the spatial context in which cells operate is informative of their function in cancerous tumors (Baharlou et al., 2019). In view of these capabilities, IMC can reveal biomarkers for monitoring therapy efficacy as well as targets for antibodies in immunotherapy, making it a valuable asset in TME analysis.

### 1.3. Difficulties in obtaining STPT and IMC images

However, it is costly to perform both STPT and IMC on the same solid tumor section. Although it is possible to first image a slice of a solid tumour using STPT and subsequently do IMC on that image slice, there are two caveats in doing this. The first is that it is difficult to align the image sets. Mechanical perturbations that take place when the vibratome slices the solid tumor during STPT and uncertainties in the exact spatial location of signals from IMC confounds the alignment process. The second is that performing both imaging techniques is expensive.

Considering these, we aim to construct a generative model that reconstructs IMC images from STPT images. This model will learn how to reliably relate STPT and IMC images. It will take in STPT images as input and output IMC images as if IMC was done. Using our model, researchers and practitioners will be able to effectively obtain both STPT and IMC images while only incurring costs from STPT. That's two for the price of one!

## 2. Methods

### 2.1. Image Dataset Description

18 physical sections of a solid tumour from breast cancer patient tissue were imaged using both STPT and IMC imaging techniques. Each physical section has 2 STPT images at two imaging depths (30 μm and 38 μm) (Bressan et al., 2021), and 40 IMC 2D grayscale images - each corresponding to a different protein marker. The 4 channels in STPT images correspond to far red, red, green, and blue. See Figure 1 for an example of STPT and IMC images. Refer to the Methods Section of González-Solares et al. (2021) for more details on the image data, image data acquisition process and image preprocessing steps.

### 2.2. Image Preprocessing

While training, we found that our hardware was not able to handle STPT and IMC images because of their size. Therefore, we resorted to crop each image into smaller chunks so that our network could train on these smaller images.

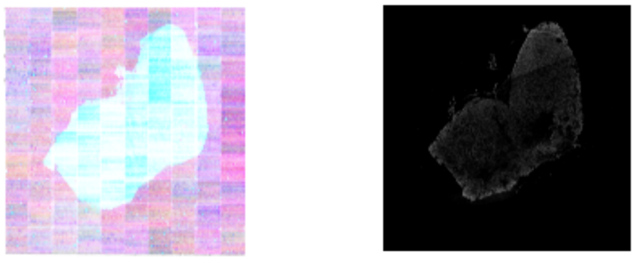
Figure 1. Example images from STPT (left) and IMC (right)

First, we performed normalization and log scaling on all images. Next, for STPT images, we concatenated the two optical sections per physical section, giving an 8-channel tensor; for IMC images, we concatenated all 40 2D grayscale images, giving a 40-channel tensor. STPT images are (20800x20800) and IMC images are (18720x18720), so we resized each STPT image to (18720x18720) using torchvision.transforms.Resize to match IMC images. Finally, we used the function torch.split from the PyTorch library to crop original images into 256x256 chunks. See Figure 2 for an example image of a chunk. See Additional Details for more details about chunks. We treated this collection of 256x256 chunks as our new dataset.

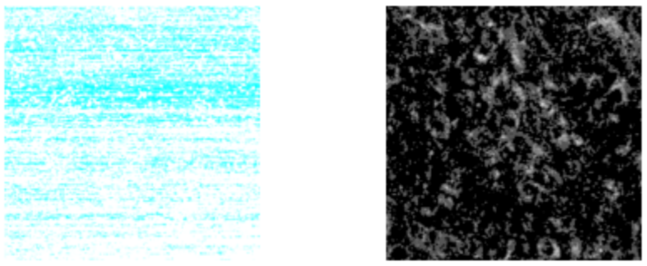
Figure 2. Cropped images from STPT (left) and IMC (right)

### 2.3. U-Net

We used a U-Net architecture (Ronneberger et al., 2015) as a benchmark to our point-cloud inspired architecture described in the next subsection. Our network architecture was nearly the same to that of the original U-Net paper. We only added a batch normalization layer before each 'block' of two 3x3 convolutions, followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation. We used mean-squared error (MSE) the AdamW (Loshchilov and Hutter, 2019) optimizer with learning rate 0.001 and weight decay 0.01 for training. 10% of all images were randomly selected to make up the validation set. We also ensured that the validation set contained images from each physical section. Minibatches of size 64 were used and the model was trained for 5 epochs.

### 2.4. PointSetGen

3D data are becoming ubiquitous because of technological advancements in applications such as robotics, autonomous driving, and virtual reality (Wu et al., 2020). Point clouds, a collection of (x,y,z) points in Euclidean space, are one of many representations of 3D data (Qi et al., 2017). Special considerations must be taken account when working with point clouds. For example, algorithms typically employ a symmetric function to respect the permutation invariance of point clouds (Qi et al., 2017, Wu et al., 2020).

We trained the 'hourglass' neural network architecture from the PointSetGen paper by Fan et al. (2016). The original aim in the paper was to reconstruct a 3D point cloud object from a single 2D image. In our application, we noted that IMC images of individual channels resembled point clouds (See Figure x) and STPT images were multi-channeled 2D images. Hence, the reconstruction task from STPT to IMC images was akin to the one described in Fan et al.

We followed the architecture described in Fan et al. ReLU activation functions and batch normalization were done after convolutional layers. We diverge from Fan et al. at the last layer where we perform a point cloud rendering technique based on the gaussian equation. The output according to the original architecture in Fan et al. is an (Nx3) tensor, where N is the number of points in the final point cloud. However, since IMC images are not truly point clouds, but 2D grayscale images, we converted the (Nx3) tensor outputted by the original architecture to a 2D image using a custom point cloud rendering technique (See Additional Materials).

## 3. Results

To evaluate performance of the U-Net-based and the point cloud-based model, we selected both cropped STPT images for reconstruction. We could not use full STPT images in our model due to hardware constraints. Cropped STPT images were chosen based on how informative they were of the model's performance (e.g., we did not choose cropped images of the black background). We ran forward passes on selected STPT images to obtain reconstructed IMC images. Each forward pass generated 40 2D grayscale images packaged in a 40-channel tensor – each channel corresponding to a different protein marker. For convenience, we only compare reconstruction on 3 randomly chosen channels. We compared each model's reconstructed IMC image by plotting both reconstructed and ground truth images using matplotlib, and by recording mean-squared error rounded to the second decimal place.

### 3.1. U-Net

Here we show some reconstruction results produced by our U-Net architecture. Each set of 3 reconstructions portray images of 3 randomly selected channels corresponding to a randomly chosen physical section and a selected chunk from the original image (see the Additional Materials for how to interpret chunks). See Figures 2-5 for reconstructed and ground truth images. Reconstructed images are on the left column and ground truth images are on the right column. Each row corresponds to a reconstructed / ground truth image pair. MSE values for each reconstruction pair (top, middle, bottom) is recorded in the captions of each figure.

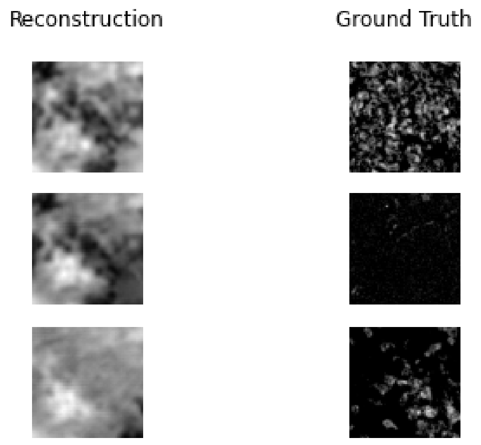

Figure 3. Physical Section 12 chunk 30/41
MSE: 207.42 (top), 276.92 (middle), 264.55 (bottom)

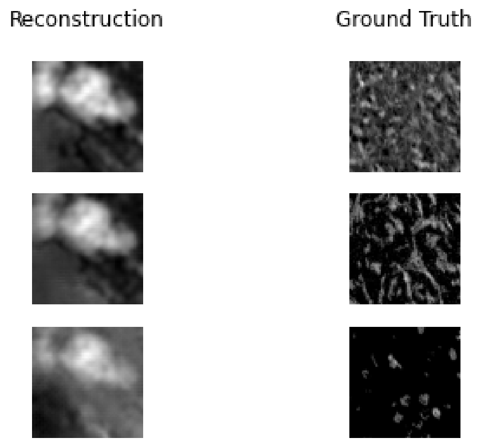

Figure 4. Physical Section 7 chunk 40/20
MSE: 288.13 (top), 269.04 (middle), 274.63 (bottom)

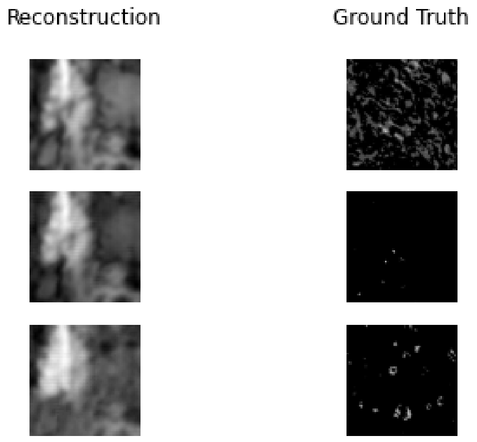

Figure 5. Physical Section 3 chunk 40/20
MSE: 243.47 (top), 277.61 (middle), 272.33 (bottom)

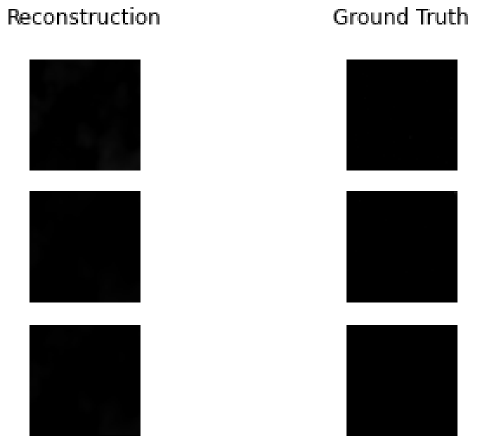

Figure 6. Physical Section 3 chunk 40/35
MSE: 1.8 (top), 0.3 (middle), 0.2 (bottom)

We observed poor reconstructions overall. The reconstructions do not seem to capture fine details present in ground truth images. The reconstructed image for physical section 12 chunk 30/41 in Figure 2 seemed to show slightly better results than that of Figures 3 and 4 (particularly the top reconstruction), but by a negligible amount. The network was able to accurately reconstruct the black background as seen in Figure 5.

We also plotted the loss value for each epoch. The loss is constant throughout training indicating the model is not learning over epochs. This might be due to the fact that U-Net is not able to pick up on grain-like structures, since it was originally used as a cell segmentation algorithm where cells were relatively large. In future studies, it would be good to revisit the U-Net model to try and uncover why this is the case.

### 3.2. PointSetGen

Our hardware resources prevented proper training of our PointSetGen architecture. Since the 'hourglass' version of PointSetGen contains many convolutional layers, it was very memory demanding. We tried various workarounds to evade this problem. First, we lowered the batch size to 1, but still encountered the same memory issues. Next, we converted our model and images to 32-bit floats, which removed the memory issue, but produced nan loss values due to numeric instability. We could also fix the memory issue by using PyTorch's automatic mixed precision package torch.cuda.amp, which performs some operations using torch.float32 datatype and other operations using torch.float16, depending on numerical precision demands.

We thought the numeric instability may come from model parameters going to inf, -inf or nan at some point during training. In an effort to combat numeric instability, we applied the He weight and bias initialization technique (He et al., 2015) when initializing the model parameters, which unfortunately did not resolve the problem. We also tried to lower the learning rate, but to no avail.

## 4. Future Directions

We note a significant limitation in our study. Our computing resources were very limited given the amount of time we had to complete the report, so we were not able to train for many epochs in the case of the U-Net model. Therefore, in later iterations of the project, it would be advantageous to gain access to a cluster with more computing power. In the case of our point cloud model, the numeric instability might suggest to use another point cloud architecture other than PointSetGen that is more numerically stable for our particular application, or view IMC images as something else instead of a point cloud.

Some other considerations that could improve performance include hyperparameter tuning, assessing different types of loss functions such as the binary cross entropy (BCE) loss, and exploring data augmentation methods such as elastic deformation.

## 5. Conclusion

In this study, we attempted to reconstruct IMC images from STPT images, which is useful in bio-medical imaging applications for mitigating costs and reducing inconsistencies during imaging. The task involved an image preprocessing pipeline to crop raw images to smaller (256x256) chunks to comply with GPU memory constraints for exploiting parallelization. Two neural network architectures were proposed to implement the reconstruction. The first architecture was based on the well known U-Net architecture introduced by Ronneberger et al. (2015). We observed poor reconstruction results from the U-Net architecture through qualitative and quantitative analysis. U-Net reconstructed images did not pick up the fine details of ground truth IMC images and had an average reconstruction error using MSE of about 270. Moreover, the loss curve seems to fluctuate around the same value, indicating the model is not learning effectively.

## 6. Additional Materials

### 6.1. Rendering Point Clouds as a 2D image

First, we first constructed two 256x256 mesh grids using torch.meshgrid and concatenated them to a (2x256x256) tensor. Next, we extracted the x and y coordinates from the (Nx3) tensor, to form an (Nx2) tensor. We then used the gaussian equation to compute the intensity of each pixel, resulting in an (Nx256x256) tensor. This operation allowed gradients to flow through via coordinates to make training feasible. Refer to the Additional Materials for more details. We also constructed a separate multilayer-perceptron (MLP) that applies a convolutional layer to the last hidden layer of the neural network and outputs a (40xN) tensor. Finally, we perform element-wise multiplication between the tensor from the gaussian equation operation and the tensor from the MLP, ultimately giving us a (40x256x256) tensor; 40 2D images of size (256x256) for each protein marker.

Generally speaking, let uv be a tensor with shape (HxWx2), xy be a tensor with shape (Nx2) and x be a tensor with shape (CxN). The following piece of code describes the gaussian equation to compute the intensity of each pixel. The result is a (40x256x256) image, provided the MLP outputs a 40-channeled tensor.

```
torch.exp(((uv[None,:,:,0]
           -xy[:,None,None,0])**2
   + (uv[None,:,:,1]
       -xy[:,None,None,1])**2)
   / (xy[:,None,None,2]**2+1)) * MLP(x)
```

### 6.2. Chunks

We form a grid on each STPT and IMC image when cropping original raw images. Each grid component is a (256x256) tensor. Therefore, (18720x18720) images will be sectioned into a (72x72) grid, each element in this (72x72) grid being a (256x256) cropped image (however, the last row and column are of size (32x32) because 18720 is not perfectly divisible by 256). We used 'chunk xx/yy' to locate where a specific cropped image lies in the original raw image. The 'xx' refers to the row index and 'yy' refers to the column index of the (72x72) grid. Therefore, 'chunk 30/41' would refer to the square whose top left corner's coordinates are (7424,10240) and bottom right coordinates are (7679, 10495) on the original raw STPT or IMC image.