IFT 3710 - Team "GANg"
Data Augmentation for Cell Segmentation

Bio Samir Gbian[1], Kamen Damov[1], Simon Langlois[1], Guillaume Genois[1], and Johann Sourou[1]

[1]Departement of Computer Science and Operations Research
[1]University of Montreal

March 9, 2025

GitHub Code Repository

1 Preprocessing

This section details the image transformation pipeline used in a microscopy image segmentation framework based on the MONAI (Medical Open Network for AI) library. The transformations are applied to both the input images and their corresponding segmentation labels to augment the dataset and improve model generalization.
Training Transformations The training pipeline incorporates both spatial and intensity transformations to create a diverse set of training samples:

$$\mathcal{T}_{train} = T_1, T_2, \ldots, T_n \tag{1}$$

where each T_i represents a specific transformation operation.
Loading and Channel Transformations

$$T_1 = \textbf{LoadImaged}(\text{keys} = [\text{"img"}, \text{"label"}]) \tag{2}$$
$$T_2 = \textbf{AddChanneld}(\text{keys} = [\text{"label"}]) \tag{3}$$
$$T_3 = \textbf{AsChannelFirstd}(\text{keys} = [\text{"img"}] \tag{4}$$

These initial transformations handle the loading of images and labels, ensuring the proper channel arrangements. The images are originally in format $(H, W, 3)$ and are transformed to $(3, H, W)$, while labels are expanded from (H, W) to $(1, H, W)$.
Intensity Normalization

$$T_4 = \textbf{ScaleIntensityd}(\text{keys} = [\text{"img"}]) \tag{5}$$

This transformation normalizes the intensity values of the input images, scaling them to a standardized range.
Spatial Transformations

$$T_5 = \textbf{SpatialPadd}(\text{keys} = [\text{"img"}, \text{"label"}]), \tag{6}$$
$$T_6 = \textbf{RandSpatialCropd}(\text{keys} = [\text{"img"}, \text{"label"}]), \tag{7}$$
$$T_7 = \textbf{RandAxisFlipd}(\text{keys} = [\text{"img"}, \text{"label"}]), \tag{8}$$
$$T_8 = \textbf{RandRotate90d}(\text{keys} = [\text{"img"}, \text{"label"}]), \tag{9}$$

These transformations modify the spatial properties of the images and labels through padding, cropping, flipping, and rotation operations, which introduces geometric variability into the training set.
Intensity Augmentations

$$T_9 = \textbf{RandGaussianNoised}(\text{keys} = [\text{"img"}]), \tag{10}$$
$$T_{10} = \textbf{RandAdjustContrastd}(\text{keys} = [\text{"img"}]), \tag{11}$$
$$T_{11} = \textbf{RandGaussianSmoothd}(\text{keys} = [\text{"img"}]), \tag{12}$$
$$T_{12} = \textbf{RandHistogramShiftd}(\text{keys} = [\text{"img"}]), \tag{13}$$

These transformations modify the intensity characteristics of the images through noise addition, contrast adjustment, smoothing, and histogram manipulation, simulating various imaging conditions.
Advanced Spatial Transformations

$$T_{13} = \textbf{RandZoomd}(\text{keys} = [\text{"img"}, \text{"label"}]), \tag{14}$$

This transformation randomly scales the images and labels, creating variations in object size and resolution.
Type Conversion

$$T_{14} = \textbf{EnsureTyped}(\text{keys} = [\text{"img"}, \text{"label"}]) \tag{15}$$

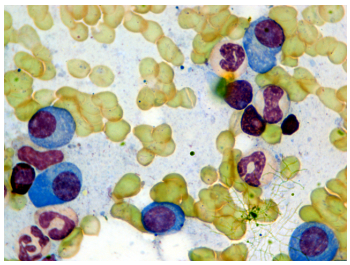
This final transformation ensures that the data types are compatible with the PyTorch framework.
Validation Transformations The validation pipeline is simpler and focuses on preparing the data without augmentation:

$$\mathcal{T}_{val} = T'_1, T'_2, \ldots, T'_m \tag{16}$$

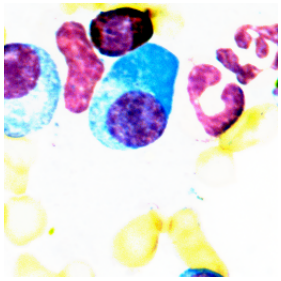
where each T'_j represents a validation-specific transformation operation.

$$T'_1 = \textbf{LoadImaged}(\text{keys} = [\text{"img"}, \text{"label"}]) \tag{17}$$
$$T'_2 = \textbf{AddChanneld}(\text{keys} = [\text{"label"}]) \tag{18}$$
$$T'_3 = \textbf{AsChannelFirstd}(\text{keys} = [\text{"img"}]), \tag{19}$$
$$T'_4 = \textbf{ScaleIntensityd}(\text{keys} = [\text{"img"}]), \tag{20}$$
$$T'_5 = \textbf{EnsureTyped}(\text{keys} = [\text{"img"}, \text{"label"}]) \tag{21}$$

The validation transformations maintain the same preprocessing steps but exclude the augmentation operations to ensure consistent evaluation.
Here is an example of the output after applying the pre-processing pipeline to an image:



(a) Original image, 640 x 480



(b) Transformed image, 216 x 216

Figure 1: Comparison of transformed image and original image

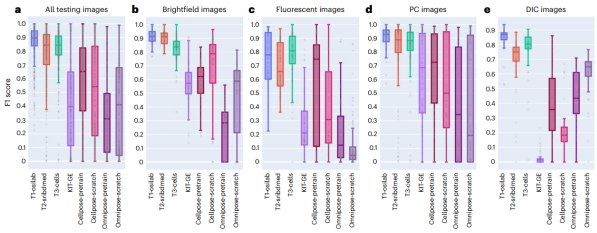
2 Models

2.1 Baseline U-Net

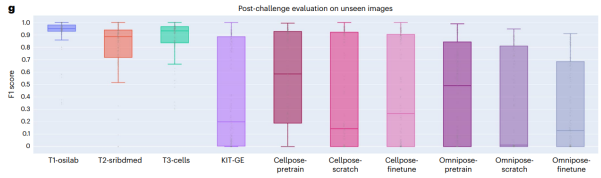
The U-Net convolutional neural network (CNN) is a widely used model for image segmentation. The competition authors provide a naive U-Net model as a baseline. We utilized this model to gain a better understanding of the data and the appropriate approach for working with it. On the validation data subset, the model achieved an F1 score of 0.418. We used their architecture as an inspiration for ours.

2.2 Cellpose & Omnipose

The Cellpose and Omnipose models are state-of-the-art (SOTA) models for generalized cell segmentation. Using these models was part of our initial project objectives. However, we observed that the competition authors had already experimented with them and found their results to be highly variable.
As a result, we plan to adapt our objectives by using these models in combination with various data augmentation techniques, such as GANs, to assess whether they reduce the variability. This also led us to consider other SOTA segmentation models, such as SAM-2 and YOLO.



(a) Dot and box plot of the F1 scores on the whole testing test



(b) Quantitative comparison on the post-challenge testing set

Figure 2: Results from the authors of the NeurIPS competition

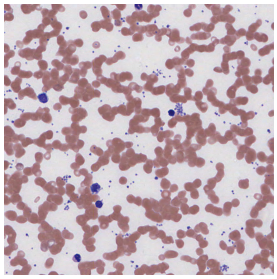
3 Data Augmentation

A challenge that was flagged by the other competitors was the inefficiency of using unlabeled data. We try a novel way to generate pseudo masks. One solution is to feed the unlabeled images into pretrained segmentation models (SAM-2 and YOLO) to generate the pseudo masks. Another solution is to train a cGAN (Conditional-GAN) that will be trained on the labeled pairs, and use the generator of this trained model to generate masks conditionally to the unlabeled image. These solutions are attempts to use the unlabeled for training the segmentation models. Furthermore, we have explored image captioning as a way to augment the images with natural language captions.

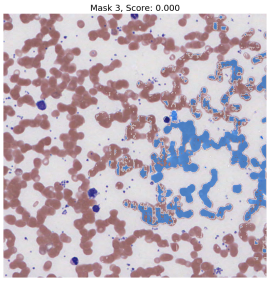
3.1 SAM-2

Meta's SAM-2 model is another state-of-the-art architecture for universal segmentation. The objective is to exploit an unbalanced dataset (majority of unlabeled images) using SAM2, which generates quality mask.

We began by applying the model directly, and it successfully identified a few cells (see figure below). Although the result is encouraging, it isn't good enough. One approach to explore in the coming weeks is fine-tuning the model using our own data.



(a) Original image



(b) Masks applied

Figure 3: SAM-2 pretrained segmentation results

On the numerous tests carried out, we can notice that the model produces masks comparable to data with a label for certain cell types. However, it tends to over-segment images with purple cells (images containing a very large number of cells overall but with few cells for which we want the masks).

The developed code responds to this challenge with a hybrid approach: post-SAM colorimetric clustering (K-means on HSV space) makes it possible to specifically isolate the masks corresponding to the target (purple) and non-target cells, thus improving the robustness of the model while capitalizing on the available unsupervised data.

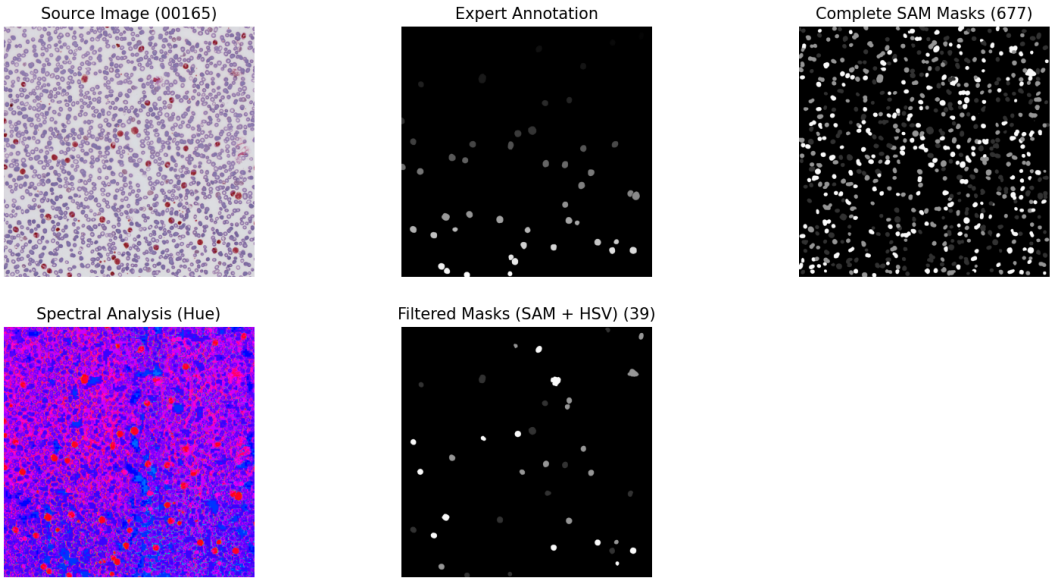


Figure 4: SAM-2 model vs SAM-2 + HSV clustering

Overall the SAM-2 model could be used for some cell types, although this will require additional filtering to match expert annotations.

3.2 YOLO11

The YOLO11 and YOLO12 models are state-of-the-art (SOTA) architectures designed for universal segmentation. We initially attempted to apply these models directly, but with no success, as none were able to detect a single cell. We then tried fine-tuning the models; however, this required more memory than our available GPU (RTX 3060) could handle, causing the training process to crash. We experimented with various models, including the smallest version (Nano), but unfortunately, all attempts resulted in failure. Further research is needed in the coming weeks to reduce memory requirements or explore alternative solutions, such as leveraging resources from Calcul Québec.

3.3 Pix2Pix

Generative Adversarial Networks (GANs) have been widely used for image-to-image translation tasks, and conditional GANs (cGANs) extend this concept by learning a mapping from a source domain X to a target domain Y given paired data. In this work, we leverage the Pix2Pix architecture, a cGAN designed for paired image translation, to generate pseudo-masks from raw images.

Instead of employing a traditional segmentation model, we fine-tune Pix2Pix using a corpus of labeled data consisting of *(raw image, segmentation mask)* pairs. The generator G in the Pix2Pix framework learns to map an input raw image $x \in X$ to a synthetic segmentation mask $\hat{y} \in Y$, while the discriminator D evaluates the realism of the generated mask by distinguishing it from real ground truth segmentation masks. This adversarial setup optimizes the generator to produce segmentation masks that closely resemble human-labeled annotations.

Once the Pix2Pix model is trained, we apply the generator to previously unlabeled raw images to produce synthetic segmentation masks, which we refer to as *pseudo-masks*. These pseudo-masks will then be integrated into a larger training set to improve the performance of a dedicated segmentation model. This approach allows us to expand the labeled dataset without requiring additional manual annotations, leveraging adversarial learning to refine mask generation. That said, we have issues stabilizing the training of this model, a recurring issue with GANs.

3.4 Image captionning

For image caption generation, we will use transformer-based vision-language models to generate descriptions for both labeled and unlabeled cell images. We will fine-tune a pre-trained ***BLIP-2*** or ***OFA*** model, which consists of a Vision Transformer (ViT) as the encoder and a text decoder, such as ***T5***, to generate captions. The model will be

trained using a cross-entropy loss function and evaluated using ***BLEU***, ***CIDEr***, and ***METEOR*** scores. To improve caption quality, we will apply a filtering process using cosine similarity between generated captions and existing ones, ensuring relevance and coherence. Additionally, we will augment captions by using ***T5*** for paraphrasing, ***WordNet*** for synonym replacement, and back-translation techniques to enhance diversity. The key libraries include transformers for model fine-tuning, sentence-transformers for similarity scoring, and ***deep-translator*** for back-translation. This approach will enrich our dataset with meaningful textual descriptions, potentially improving the performance of the cell segmentation model.

4 Next steps

Over the next two weeks, the team will focus on finalizing the data augmentation process. Kamen will work on stabilizing Pix2Pix training and generating pseudo-masks to be used for training the segmentation models. Johann will continue experimenting with pretrained segmentation models, validating their effectiveness both quantitatively and qualitatively. Samir will refine caption augmentation by cross-validating different models and prompts, while also enhancing preprocessing by developing a cropping script to extract more labeled samples from large images. Simon and Guillaume, who are focused on segmentation, will continue exploring new versions of the leading cell segmentation models, such as Cellpose 3.0 and Omnipose, while also actively contributing to finalizing the data augmentation step. By the end of these two weeks, we expect to have fully augmented datasets and models ready for training. The following phase will involve running trials with different augmentation strategies and model architectures, conducted in parallel with paper writing, presentation preparation, and demo development.