Organoids Segmentation using Self-Supervised Learning: How Complex Should the Pretext Task Be?

Bachelor's Project Thesis

Bart van der Woude, s3498891, b.r.van.der.woude@student.rug.nl,
Supervisor: L.R.B. Schomaker, l.r.b.schomaker@rug.nl
A. Haja, a.haja@rug.nl

Abstract: Deep learning methods are well-suited for biomedical image analysis, this research focuses specifically on progenitor liver organoid segmentation. Most popular supervised-learning approaches, however, require large annotated data sets that are time-consuming and costly to create. Self-supervised learning (SSL) has proven to be a viable method for increasing downstream performance, through pre-training models on a pretext task. However, the literature is not conclusive on how to choose the best pretext task. This research sheds light on how the complexity of the pretext task affects organoid segmentation performance, in addition to understanding whether a self-prediction or innate relationship SSL strategy is best suited for organoid segmentation. Eight novel self-prediction distortion methods were implemented, creating a number of simple and complex pretext tasks. Two well-known innate relationship pretext tasks, Jigsaw and Predict rotation, were implemented in order to compare strategies. Results showed that complexity of the pretext tasks do not correlate with segmentation performance. However, complex models (average F1-score = 0.862) consistently, albeit with a small effect size, outperform simple tasks (average F1-score = 0.848) possibly due to acquiring a wider variety of learned features after pretext learning despite not being necessarily more complex. The small effect size and high standard deviations of F1-score segmentation performances make the results non-conclusive. Similarly, self-prediction models (average F1-score = 0.856) consistently outperformed innate relationship models (average F1-score = 0.848). However, again, having a small effect size and high standard deviation make the observed effect non-conclusive. Lastly, results showed that more pretext training data improves downstream performance under the condition that there is a minimum amount of downstream training data available. Too little downstream training data combined with more pretext training data leads to a decrease in segmentation performance.

1 Introduction

Using deep-learning (DL) methods for biomedical image segmentation is of great value to medicine and biological research. From finding cancer in endobronchial ultrasounds Zang et al. (2016) and breast mammography Lotter et al. (2021), to diagnosing acute ischemic stroke lesions in CT perfusion maps Shi et al. (2022). This research specifically focuses on DL for performing organoid segmentation. Organoids are *in vitro* grown tissue cultures mimicking the structure and functionality of *in vivo* organs. Researching organoids gives the opporunity to understand organ function, growth and their response to potentially useful drugs Clevers (2016).

In the domain of biomedical image analysis, supervised learning methods have proven greatly successful Aljuaid & Anwar (2022). Despite this success, these methods require copious amounts of annotated data Huang et al. (2023); Li et al. (2022); Wallace & Hariharan (2020); Jaiswal et al. (2020). These data sets are challenging to acquire for biomedical imaging, as annotation is expensive, time-consuming and calls for expertise in this field Lotter et al. (2021); Huang et al. (2023); Bruch et al. (2020); Ronneberger et al. (2015). Similarly, creating data sets for clinical use cases is also limited as labeling often focuses on creating a data set that is suited for a single task, rather than a data set that can be used for more use cases Huang et al. (2023).

Self-supervised learning (SSL) tries to solve the problem of lacking annotated data by pre-training models on similar unlabeled data solving a pseudo-task, thereby having the pre-trained network learn relevant features of the data Huang et al. (2023); Li et al. (2022); Wallace & Hariharan (2020); Jaiswal et al. (2020); Gidaris et al. (2018); Noroozi & Favaro (2016). Existing literature shows numerous well-established pseudo-tasks, hereafter called pretext tasks Huang et al. (2023). The process of pre-training through solving pretext tasks is called pretext learning. After pretext learning, the pre-trained model is trained on solving the actual task, called the downstream task, in a process called downstream learning Huang et al. (2023); Li et al. (2022); Wallace & Hariharan (2020); Jaiswal et al. (2020); Gidaris et al. (2018); Noroozi & Favaro (2016).

It has been well established that the semantic information of the data used by the model to solve the pretext task affects its learned features Huang et al. (2023); Jaiswal et al. (2020); Noroozi & Favaro (2016). The latent feature representation encodes information that is used to solve the task it was trained on. In the case of pretext learning this means that non-relevant pretext tasks will potentially create features that are not beneficial, or even detrimental, for solving the downstream task Noroozi & Favaro (2016).

For example, Figure 1.1 shows that the cars and dogs have very similar shapes but different colors. The model should classify this as similar, but it might fail to do so if it was trained on colors Jaiswal et al. (2020).

Therefore, it is important to pick the right pretext task for the downstream task. However, research is lacking on how the complexity of the pretext task affects the downstream task performance. Here, complexity refers to the difficulty of the pretext tasks, often linked to the quantity and quality of features required to solve it. A more complex pretext task might negatively affect pretext task performance, but perhaps will be beneficial for solving the downstream task.

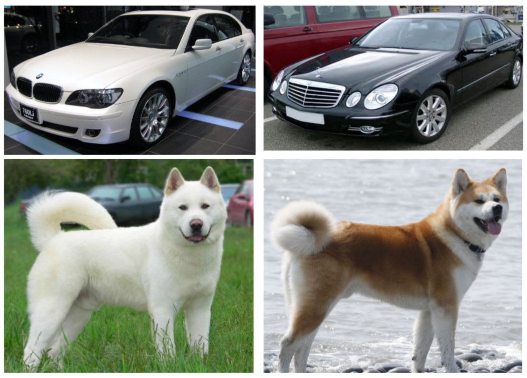
Figure 1.1: Shapes match, colors do not. Showing importance of type of pretext task Noroozi & Favaro (2016).

Furthermore, a lacking area of research is how the different SSL strategies affect downstream task performance. SSL strategies are categories of type of pretext tasks, categorized on how they augment or transform data as well as their expected output type. It is well known that existing strategies have been successful Huang et al. (2023); Li et al. (2022); Wallace & Hariharan (2020); Jaiswal et al. (2020), but it is unclear which strategy is the most effective.

Therefore, in order to improve the understanding on how the pretext task affects downstream task performance, a number of data transformation/augmentation techniques are proposed. These techniques represent the self-prediction SSL strategy. In addition, two well-known innate relationship pretext tasks are examined, i.e. jigsaw puzzle Noroozi & Favaro (2016); Wallace & Hariharan (2020) and predict rotation angle Jaiswal et al. (2020); Gidaris et al. (2018); Wallace & Hariharan (2020) in order to compare SSL strategies. The pretext tasks are implemented and used for pretext learning. This aids the model in training to perform organoid segmentation on a data set consisting of images of progenitor liver organoids.

This research aims to shed light on the topic of how the complexity of the pretext task affects the quality of the learned features after pretext learning, and thus organoid segmentation performance, with regards to the amount of data used in training. This research also aims to clarify which SSL strategy is better suited for organoid segmentation in the field of biomedical imaging, and again, with regards to the amount of data used in training. To summarize, the quality of segmentation of progenitor liver organoids is compared, where the models use transfer learned features from a network trained on simple or complex pretext tasks as well as networks trained on different SSL strategies.

In short, this research aims to answer the following research questions:

- How does the complexity of pretext tasks affect self-supervised learning of the segmentation of organoids?
- How does the pretext task strategy type affect self-supervised learning of the segmentation of organoids?
- What effect does the amount of training data, for both self-supervised and supervised learning, have on the quality of organoid segmentation in relation to the complexity of the pretext task and the pretext task strategy type?

The paper is organized into 5 sections and the structure is as follows: Section 2 represents a literature review; Section 3 contains information about the data, the structure of the proposed model and a description of the experiment design; Section 4 discusses the results of the experiments and Section 5 concludes this research and proposes possible future research.

2 Literature review

Organoid segmentation, which is the downstream task for this research, is a form of semantic image segmentation. Semantic image segmentation is the process of recognizing and localizing objects in an image by classifying each pixel in the image to a specific object-class from a predetermined set of classes Minaee et al. (2020).

Minaee et al. (2020) and Caicedo et al. (2019) list some of the more traditional methods of image segmentation. For example, thresholding, watershed and active contours were used to identify cells in an image. However, most of these traditional methods require expertise to set up and require researchers to account for imaging technique, scale and experimental conditions Caicedo et al. (2019). DL methods have proven to be a great alternative to traditional methods for semantic segmentation as they are more versatile and are more easily adapted to different experimental conditions Minaee et al. (2020); Caicedo et al. (2019); Bruch et al. (2020). Often also improving segmentation quality Minaee et al. (2020); Caicedo et al. (2019).

This research uses a U-Net type architecture, which is a fully-convolutional encoder-decoder type architecture, to segment the organoids from their background. Previous research succesfully used a U-Net for segmentation purposes Ronneberger et al. (2015); Minaee et al. (2020); Caicedo et al. (2019); Bruch et al. (2020).

Although there exists organoid research circumventing the need for segmentation Mergenthaler et al. (2021), segmentation can prove useful to solve issues with the high dimensionality of organoid data, acquisition artifacts, low contrast, and bright-field noise Louey et al. (2021). A decent number of existing organoid analysis techniques are based on techniques other than DL Ranjbaran & Nazemi (2023) such as thresholding Clevers (2016). However, DL methods, including DL used for segmentation, provide more stability and robustness at the drawback of requiring more effort to set up Ranjbaran & Nazemi (2023).

DL methods requiring more effort to set up refer to, among other things, requiring a large amount of annotated data. In order to address this issue this research uses a SSL approach.

SSL is the process of using pseudo-labels on unannotated data to train models to extract semantic information by creating meaningful feature representation of the input data in a process called pretext learning. After pretext learning, the pretrained model is trained on annotated data to perform the downstream task. In this case, the research focuses on progenitor liver organoids segmentation. Through using SSL the model is able to perform better, despite being trained on limited annotated data, as the model has already learned to extract data specific semantic information Huang et al. (2023); Li et al. (2022); Wallace & Hariharan (2020); Jaiswal et al. (2020); Gidaris et al. (2018); Noroozi & Favaro (2016).

As previously mentioned, there are a large number of clinical use cases per image data type. Having models pretrained on specific data could prove useful as the pretrained models can then be adapted to specific use cases. For example, models pre-trained on EEG data can be trained for emotion recognition Li et al. (2022), recognizing lesions Salim et al. (2021) or analyzing sleep activity Aboalayon et al. (2016).

In the case of organoid segmentation the proposed SSL approach could be adapted and used for many other research purposes in organoid research, and even biomedical image analysis in general.

There are a number of categories of SSL strategies such as generative, contrastive, innate relationship and self-prediction Huang et al. (2023). This research focuses on **innate relationship** and **self-prediction type strategies**.

In **self-prediction strategies** an input image is augmented and/or transformed on a portion of the image, creating a distorted image that serves as input during pretext learning. The pretext task consists of reconstructing the distorted image back to the original image, or ground truth (GT), using a reconstruction loss. The unaltered portions of the distorted image are meant to inform and aid the model in reconstructing back to GT.

Alternatively, **innate relationship strategies** use a pretext task that has pseudo labels not related to the original data. Instead, the labels are directly related to pretext task distortion method and expected output structure. In this case, the model uses the structural information of the data to solve the pretext task, which helps the model with learning a solid feature representation.

Numerous pretext tasks have already been proposed. Popular pretext tasks for self-prediction strategies include:

- Rotate image Jaiswal et al. (2020); Gidaris et al. (2018);
- Crop (and zoom) image Jaiswal et al. (2020); Wallace & Hariharan (2020);
- Remove portions of image Jaiswal et al. (2020);
- Drop pixels Haja et al. (2021);
- Change colors of image Jaiswal et al. (2020).

For innate relationship strategies, well-known strategies include:

- Jigsaw puzzle, where portions of image are shuffled Noroozi & Favaro (2016);
- Find neighbour of cutout portion of image (center) Jaiswal et al. (2020);
- Degree of rotation of image Jaiswal et al. (2020); Gidaris et al. (2018).

Although it has been clearly established that the chosen pretext task affects learned features, and thus downstream task performance Huang et al. (2023); Jaiswal et al. (2020); Noroozi & Favaro (2016), it is unclear how to choose the right pretext task for the data and downstream task. This research introduces 8 novel distortion techniques that are used in creating both simple and more complex self-prediction pretext tasks. These novel pretext tasks were evaluated in order to understand which distortion technique, or combination of distortion techniques, leads to the best performing pretext task as well as to understand how the complexity of the pretext task affects downstream task performance. This research implemented the jigsaw puzzle and predict rotation angle innate relationship pretext tasks and compares these to the aforementioned self-prediction pretext tasks in order to understand which of these two SSL strategies performs best on the segmentation of organoids.

3 Methods

This section aims to provide a structured and practical approach to accomplishing the research aims of understanding how complexity and strategy type of the pretext task affect the downstream task performance with regards to the amount of data used in training.

The methods section is structured by first introducing a general outline of the research approach in section 3.1. Followed by introducing the data set and how this data set is created and divided in section 3.2. In section 3.3, the model and its training process are explained. Afterwards, relevant pretext tasks are introduced in section 3.4. And lastly, the theory of the used data evaluation metrics is given in section 3.5.

3.1 General approach

This research uses separately trained U-Net models, with a transfer-learned ResNet50 encoder at the start of pretext learning. These models were trained on self-prediction and innate relationship type pretext tasks during pretext learning. Each pretext task, for both self-prediction and innate relationship, has models trained using 10%, 30% or 50% of the available pretext learning data. Each experimental condition pertaining to the pretext task and amount of data uses 5-fold cross validation. Self-prediction pretext tasks are divided into simple and complex tasks.

After pretext learning, the best performing model per experimental condition is transferred to downstream task training. The encoder portion of the model is frozen and the model is trained on the task of organoid segmentation using either 10%, 50% or 100% of the available downstream learning training data. Downstream learning also uses 5-fold cross validation.

After pretext learning and downstream learning, 38 (pretext tasks)*3 (pretext data amount)*5 (folds) = 570 pretext trained models were obtained based on the pretext task, amount of data for pretext learning and 5-fold cross validation. 38 (pretext tasks)*3 (pretext data amount)*3 (downstream data amount)*3 (folds) = 1,026 downstream trained models were obtained based on pretext task, amount of data for pretext learning, amount of data for downstream learning and 3 rotations of 5-fold cross validation. These models were then tested on their performance on organoid segmentation and compared given their experimental conditions.

Table 3.1: Amount of available images per CZI image. Data is separated into a pretext learning, downstream learning and test set.

CZI ID	Pretext	Main	Test	Total
1CA5KMCR7K7MT53OS7FG	3571	3572	1786	8929
5ZBQ5VZO6IA5VHDX3X0U	5274	5274	2638	13186
6QL9YXESHQ6UDYDYTG3U	5510	5510	2756	13776
48JUXOC36SAUYOEQSK0W	3240	3241	1621	8102
289RM7EZ02HD117WJ5IM	3532	3533	1767	8832
A5VHDX3X0UWN2VMLYD0Y	4622	4622	2311	11555
L9YXESHQ6UDYDYTG3UP2	4487	4487	2244	11218
NPCPNEZBA0U9BCP1SLTH	3521	3522	1761	8804
P2R2WP36RVDLHB15I48C	4622	4622	2311	11555
SUN6BXJ3O0O61Z9MYE5C	2252	2253	1127	5632
Total	40631	40636	20322	101589

3.2 Data set

As mentioned previously in section 1, this research uses a data set consisting of images of progenitor liver organoids provided by the University Medical Centre Groningen (UMCG) in Groningen in the Netherlands. The liver progenitor organoids were captured using light microscopes to create CZI images.

The observed organoids were grown under 2 different growing conditions; organoids grown in a complete medium for optimal growth and organoids grown in a medium where all amino acids were removed for stumped growth. For each condition 5 CZI images were taken at an interval of 24 hours for a total of 10 CZI images. Each CZI image consists of around 14 2D slices, which, when combined, creates a 3D representation of the organoid structure. Of these 14 2D slices only the middle 4 slices showed relevant information, as the outer slices were out of focus.

The remaining 40 slices were 3828x2870 pixels in size. These large images were divided into smaller images, 636x636 pixels in size, called crops, using the sliding window method with a step increment of 60 pixels. Furthermore, images with less than 5% of relevant information were discarded. As an augmentation technique, images were rotated by 90, 180 and 270 degrees. The total data set consists of 101,589 crops. Table 3.1 shows the total amount of available crops separated by CZI image as well as the division of data over separate training and testing sets.

In addition, for each organoid crop in the dataset, a corresponding mask was created using a Mask-RCNN trained on a similar dataset. These masks were used as labels in downstream task training.

Having images with dimensions 636x636 pixels would be too large for a DL model to effectively handle. Therefore, before training, the images were resized to 320x320 pixels.

3.3 Model architecture

3.3.1 Data usage and implementation details

The models used in this research are trained on varying amounts of data. There are 40,631 images available for pretext learning, representing 40% of the total amount of images available. Pretext tasks are trained on either 10%, 30% or 50% of this available pretext learning data, which is 4063, 12,189 or 20,315 images respectively. Not all available pretext

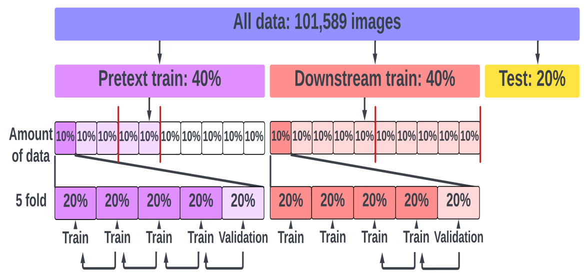

Figure 3.1: Showing division of data into pretext learning data set, downstream learning data set and testing data set. Training data sets are further divided based on experimental condition training data amount. Validation data is swapped using 5-fold cross validation.

learning data was used due to computation time restraints.

Similarly, downstream learning has 40,636 images available, which is 40% of the total data set. The downstream learning data set is distinct from the pretext learning data set and both sets do not overlap. Each model trained on a pretext task and pretext learning data amount was then trained on the downstream task using either 10%, 50% or 100% of the available downstream task data, which was 4063, 20,318 or 40,636 images respectively.

The above mentioned training was also done using k-fold cross-validation, specifically 5-fold cross-validation. 5-fold cross-validation is the process of switching the validation set of the training data 5 times over the whole training data set. Using this method ensures that the division of training-validation data does not trap the model in a local minimum. The downstream task training was divided using 5-folds, i.e. 20% validation data, but only swapped 3 times resulting in 3 models rather than 5. Figure 3.1 shows the division of data for pretext learning and downstream learning. The red vertical lines mark the amount of data used for training, and the curved arrows indicate the 5-fold cross-validation.

As previously mentioned, a U-Net was used for this research. A U-Net consists of a 'contracting path' (the encoder) that creates a latent feature representation of the input capturing semantic information and an 'expanding path' (the decoder) using this feature representation to solve the task at hand Ronneberger et al. (2015); Caicedo et al. (2019); Bruch et al. (2020).

Pretext learning was performed for both the self-prediction strategy pretext tasks and the innate relationship strategy pretext tasks using a Structural Similarity Index Measure and Spare Categorical Cross Entropy loss function respectively. To ensure that the encoder portion of the U-Net accurately represents relevant features, the encoder is frozen during downstream learning He et al. (2020); Chen et al. (2020). Downstream learning makes use of the Intersection over Union loss function. Both pretext learning and downstream learning use the Adam optimizer and have batch learning with batch-size of 16 over 50 epochs.

3.3.2 Loss functions

The Structural Similarity Index Measure (SSIM), used as a reconstruction loss for self-prediction pretext learning, is a metric for comparing a reconstructed predicted image to the GT image Wang et al. (2004). The metric outputs a value ranging from [-1, +1], where +1 represents perfect similarity and -1 represents extreme dissimilarity. Rather than comparing pixels from two images, SSIM compares patches of the reconstructed image to corresponding patches of the GT. Using this pixel neighborhood approach ensures a more human method of comparing image quality. The SSIM formula is a combination of calculating luminance, contrast and structure Wang et al. (2004):

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3.1}$$

x represents an image (or patch of an image) and y represents the corresponding GT image (again, or patch). μ_x and μ_y are the average pixel values of x and y, respectively. σ_x^2 and σ_y^2 are the pixel value variances of x and y, respectively.

$$C_1 = (K_1L)^2 \tag{3.2}$$

$$C_2 = (K_2L)^2 \tag{3.3}$$

Where K_1, K_2 are constants to avoid weak denominator, i.e. division by zero error, often $K_1 = 0.01$,

$K_2 = 0.03$. L is the dynamic range of pixel values, i.e. 255 in this case. σ_{xy} is the covariance of x and y.

$$\sigma_{xy} = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu_x)(y_i - \mu_y) \tag{3.4}$$

Where N is the total number of pixels in the image(-patch).

Generally, the SSIM is calculated for sections of the image and in the end the global mean is calculated for the complete image Wang et al. (2004). The loss function is calculated using the formula:

$$LossSSIM(x, y) = 1 - SSIM(x, y) \tag{3.5}$$

The Sparse Categorical Cross Entropy (SCCE) is used as a classification loss for both the jigsaw puzzle and prediction rotation angle innate relationship pretext task. The SCCE is based on the Categorical Cross Entropy (CCE) loss function Shah (2023):

$$CCE(p, q) = -\sum_{x=1}^{N} p(x) \cdot \log q(x) \tag{3.6}$$

Where p is the label vector containing value 1 for the correct class and 0 for other classes. q is the predicted softmax class probabilities. N represents the number of classes. SCCE, as opposed to CCE, uses an integer value to represent the class label (e.g. [2]) where CCE uses a hot encoded label vector (e.g. [0,0,1,0]).

The Jaccard Distance is used as a segmentation loss for downstream learning and is based on the Jaccard Index, or Intersection over Union (IoU). IoU is a metric used to compare two images on the precision of their classification. The formula for the IoU is:

$$IoU(x, y) = \frac{|x \cap y|}{|x| + |y| - |x \cap y|} \tag{3.7}$$

x represents an image, y represents the corresponding reference image.

The Jaccard Distance uses the IoU to express it as a loss function:

$$JD(x, y) = 1 - \frac{|x \cap y|}{|x| + |y| - |x \cap y|} \tag{3.8}$$

The Jaccard Distance will be referred to in this research as IoU.

3.3.3 Optimizer

Adam, or Adaptive Moment Estimation, was used as the optimizer for both pretext learning and downstream learning. Adam is an optimizer used to update learning rates over training steps Kingma & Ba (2014). Adam can be seen as an extension to stochastic gradient descent. Classical stochastic gradient descent has a single learning rate for all parameters in the model, but Adam has individual learning rates for all parameters Ruder (2016). Adam adapts learning rates based on the average first moment, the mean, as well as the average of the second moments of the gradients, the uncentered variance Ruder (2016). Overall, it is recommended to use Adam as the optimizer for most cases Ruder (2016).

3.4 Pretext task

3.4.1 Self-prediction

The self-prediction SSL strategy utilized in this research is based on altering portions of the input and using the unaltered portions as information on reconstructing the altered portions. Self-prediction uses images distorted using transformation techniques as input data, with the GT as labels. A pretext task consists of reconstructing the transformed image back to GT. Eight self-prediction transformation techniques are proposed:

- Blur (b): uses Gaussian blur to blur complete image;
- Drop (d): makes 4 randomly positioned boxes of 50x50 pixels black;
- Shuffle (s): swaps 4 randomly positioned boxes of 50x50 pixels;
- Rotate (r): rotates 4 randomly positioned circles with radius of 25 pixels by a random degree;
- Blur boxes (B): uses Gaussian blur to blur 4 randomly positioned boxes of 50x50 pixels;

- Drop pixels (D): turns 25% of pixels in 4 randomly positioned boxes of 50x50 pixels black;

- Shuffle and rotate (S): rotates and swaps 4 randomly positioned circles with radius of 25 pixels;

- Rotate boxes (R): rotates 4 randomly positioned boxes of 50x50 pixels by either 90, 180 or 270 degrees.

In this research, a pretext task using a singular transformation technique is considered a simple pretext task. A combination of two transformation techniques is considered a complex pretext task. The proposed pretext tasks, both simple and complex, are shown in Figure 3.2. Four sections of 50x50 pixels were distorted with no overlap per distortion technique. The distortion of four sections was chosen as it would be likely to distort an organoid in the image as well as leave sufficient information for the model to reconstruct the image to GT.

3.4.2 Innate relationship

The innate relationship SSL strategy uses pseudo-labels generated based on the pretext task, rather than data dependent labels Huang et al. (2023). This research implements two of the most popular innate relationship strategies: solving a jigsaw puzzle Noroozi & Favaro (2016); Wallace & Hariharan (2020) and predicting image rotation angle Gidaris et al. (2018); Wallace & Hariharan (2020). It is important to note that these pretext tasks are essentially (multi-class) classification tasks, rather than reconstruction tasks. Figure 3.3 shows examples of distorted images used as pretext task input for both innate relationship pretext tasks. As explained by Wallace & Hariharan (2020), using a predict rotation angle pretext task on texture type data does not lead to good results. Moreover, rotation was also used as an augmentation technique for the data set. Therefore, the rotation is done on a section of the image. The pretext task consists of predicting the correct rotation angle of this section.

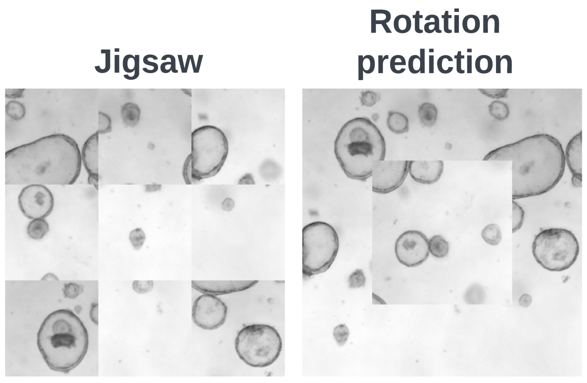

Figure 3.3: Examples of innate relationship Jigsaw and Predict rotation angle pretext input. Expected output structure is list of original section positions for Jigsaw and rotation angle for Predict rotation angle.

3.5 Evaluation metrics

3.5.1 F1-score

One of the most popular performance metrics is the F1-score, which is calculated using precision and recall.

$$Precision = \frac{TP}{TP + FP} \tag{3.9}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.10}$$

TP refers to the true positive fraction, FP to the false positive fraction and FN to the false negative fraction.

$$\text{F1-score} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{3.11}$$

3.5.2 PSNR

Peak Signal-to-Noise Ratio (PSNR) is a measure comparing peak value of an input to the noise comparing input to GT. Faragallah et al. (2021) PSNR can be calculated using the formula: Faragallah et al. (2021)

$$PSNR(x, y) = 10 \cdot \log(\frac{f_{max}^2}{\sqrt{MSE(x, y)^2}}) \tag{3.12}$$

Where $f_{max} = 255$, which is the maximum value of the range of possible pixel values. x represents an image, y represents the corresponding reference

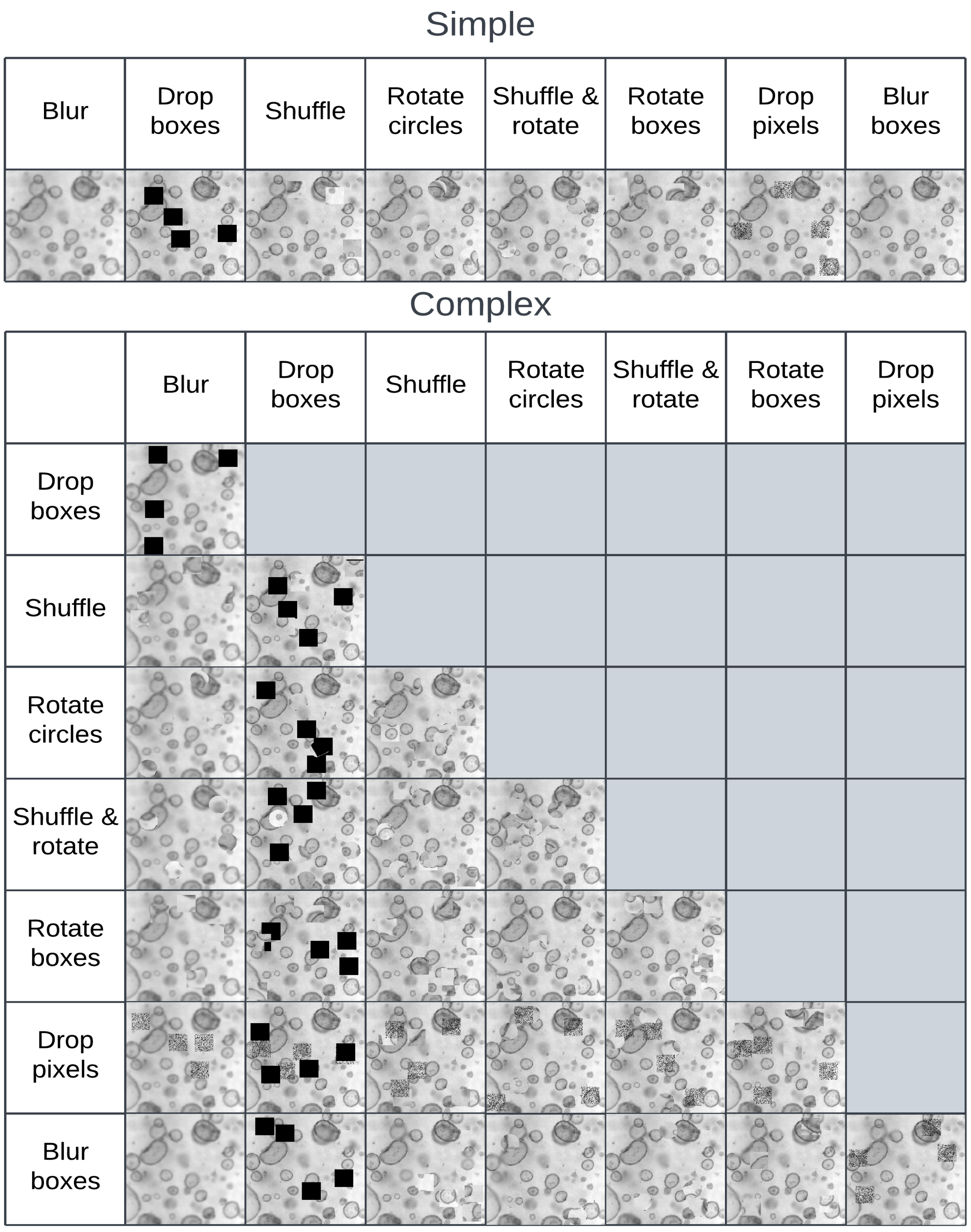

Figure 3.2: Top table presents proposed distortion methods of simple pretext tasks. Bottom table presents combinations of aforementioned pretext tasks, named complex pretext tasks.

image. The Mean Squared Error (MSE) is calculated by computing the average euclidean distance between all predicted pixel values and their corresponding GT pixel values.

$$MSE = \frac{1}{N \cdot N} \cdot \sum_{i=0,j=0}^{N,N} \|x_{i,j} - y_{i,j}\|^2 \qquad (3.13)$$

Where N is one side of the dimensions of the input matrices, i.e. $N = 320$ for a 320x320 pixel image. i,j represents positions in the input matrices.

4 Results

This section provides the results for the three research questions defined in Section 1. Each of the following sub-sections provide the results for one of the aforementioned research questions.

4.1 How does the complexity of pretext tasks affect self-supervised learning of the segmentation of organoids?

This sub-section presents the results of the effects of simple and complex self-prediction pretext tasks on pretext and downstream performance.

After pretext learning, models trained on simple and complex self-prediction pretext tasks were tested on their pretext task performance. In the case of self-prediction, pretext task performance is the ability of a model to reconstruct a distorted image to GT. Performance was measured using the PSNR metric. Figure 4.1 shows box plots of **pretext PSNR performance** of **simple and complex self-prediction** models. These box plots include models of all pretext and downstream training data amounts. The models trained on simple tasks have overall higher pretext performance (average PSNR $= 19.488 \pm 3.414$) compared to models trained on complex tasks (average PSNR $= 18.402 \pm 3.280$). This shows that models trained on reconstructing a single distortion to GT, rather than a combination of two distortions, have reconstructions more similar to GT. In turn, this suggests that simple tasks are overall less complex for the model to learn to solve compared to complex tasks, regardless of variability of complexity of pretext tasks.

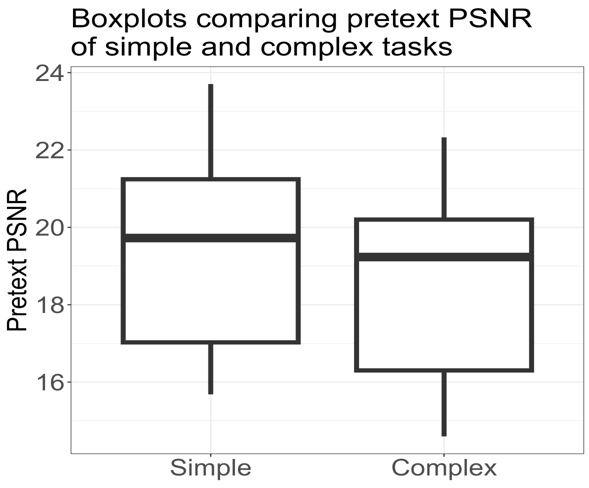

Figure 4.1: Box plots showing average pretext PSNR performance on solving the pretext task separated by simple and complex self-prediction models.

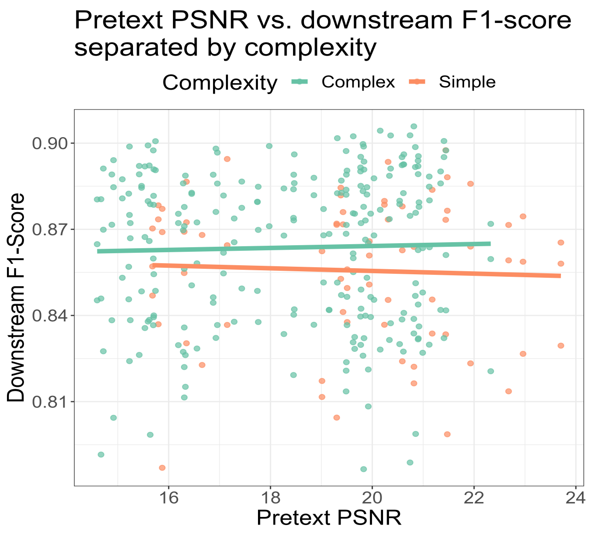

Figure 4.2: Scatter plot showing average downstream F1-score performance (x-axis) and pretext PSNR performance (y-axis). Data points represent pretext trained models. Data points and trend lines are separated using color representing simple (orange) and complex (green) models.

After downstream learning, self-prediction models were tested on their downstream task performance. The F1-score was calculated comparing predicted test set segmentation masks to the respective GT segmentation masks. Figure 4.2 shows models of all training data amounts placed using **pretext PSNR performance** as the x-axis and **downstream F1-score performance** as the y-axis separated by **simple and complex** using color. The trend line represents downstream F1-score change based on pretext performance, again, divided into simple and complex models using color. The trend line for models trained on complex tasks shows a higher F1-score compared to the trend line for models trained on simple tasks. There appears to be a slight rate of change in downstream performance given pretext performance for both the simple and complex conditions. These slopes, however, are very minimal. Therefore, pretext performance measured in PSNR does not seem to correlate with downstream performance measured in F1-score. However, models trained on complex tasks do seem to outperform models trained on simple tasks.

Due to varying complexity of distortion methods, and therefore their combinations, the only true measure of complexity of the pretext tasks is the pretext PSNR performance, as lower scoring pretext performance suggests a more complex pretext task. Given that there appeared to be no correlation between pretext performance and downstream performance, it suggests that complexity of the pretext task does not directly correlate to any difference in downstream performance.

Table 4.1 and Figure 4.3 report similar results. Table 4.1 shows average and standard deviation of **downstream F1-score performance** separated by simple/complex and training data amounts. Models trained on complex tasks outperformed models trained on simple tasks on average downstream performance. Figure 4.3 shows box plots of average **downstream F1-score performance** including all training data amounts separated into **simple and complex**. These box plots show a higher median downstream performance of complex models (median F1-score = 0.870) compared to simple models (median F1-score = 0.862). In addition, 57.1% of complex models perform above the simple median compared to 36.1% of simple models that perform above the complex median. Figure 4.1 showing pretext PSNR performance shows average simple model performance (average PSNR = 19.488 ± 3.414) outperformed average complex model performance (average PSNR = 18.402 ± 3.280). Figure 4.3 shows downstream F1-score performance with opposite results compared to pretext performance, average complex model performance (average F1-score = 0.862 ± 0.031) outperformed average simple model performance (average F1-score = 0.848 ± 0.059).

The tasks using a specific pretext distortion method, both simple and complex, proves use-

Table 4.1: Average downstream F1-score for simple and complex models and amount of data used in training. Rows represent the amount of downstream training data, columns represent the amount of pretext training data, both are grouped by simple and complex self-prediction models. The overall block shows the average downstream F1-score over all training data amounts.

Simple tasks	10% pretext	30% pretext	50% pretext
10% downstream	0.831 ± 0.022	0.824 ± 0.030	0.770 ± 0.145
50% downstream	0.864 ± 0.011	0.865 ± 0.013	0.867 ± 0.022
100% downstream	0.868 ± 0.024	0.877 ± 0.013	0.866 ± 0.021
Complex tasks	10% pretext	30% pretext	50% pretext
10% downstream	0.836 ± 0.014	0.828 ± 0.022	0.830 ± 0.019
50% downstream	0.871 ± 0.013	0.880 ± 0.010	0.870 ± 0.030
100% downstream	0.876 ± 0.021	0.876 ± 0.038	0.888 ± 0.017
Overall			
Simple tasks	$\mathbf{0.848 \pm 0.059}$	Complex tasks	$\mathbf{0.862 \pm 0.031}$

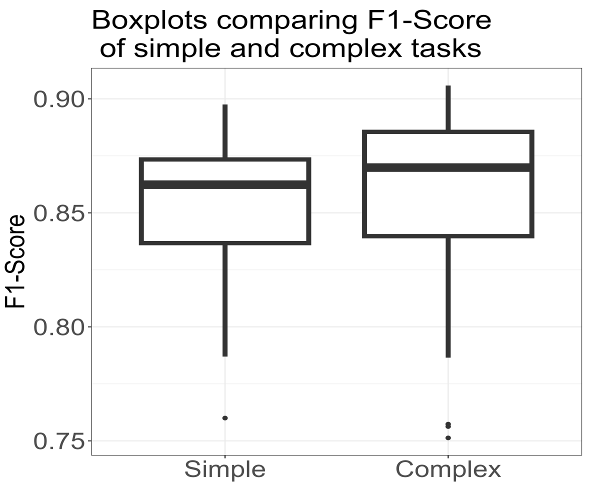

Figure 4.3: Box plots showing average downstream F1-score performance of self-prediction models separated into simple and complex.

Table 4.2: Average downstream F1-score performance for all pretext distortion methods and amount of data used in training. Rows show self-prediction, simple and complex combined, models grouped by distortion method used in the pretext task, e.g. pretext task *b_d_4* is counted as blur *(b)* and drop boxes *(d)* distortion method. Furthermore, performance was separated over columns based on pretext training data mount and grouped based on downstream training data amount. The block showing overall performance is the average downstream performance of distortion methods over all pretext and downstream training data amounts.

Downstream 10%			
Distortion method	Pretext 10%	Pretext 30%	Pretext 50%
b	0.830 ± 0.126	0.822 ± 0.135	0.779 ± 0.137
d	0.843 ± 0.119	0.838 ± 0.118	0.837 ± 0.126
s	0.837 ± 0.125	0.825 ± 0.130	0.830 ± 0.126
r	0.839 ± 0.124	0.829 ± 0.131	0.836 ± 0.127
S	0.842 ± 0.124	0.833 ± 0.127	0.832 ± 0.127
R	0.833 ± 0.127	0.839 ± 0.121	0.832 ± 0.132
D	0.834 ± 0.125	0.819 ± 0.137	0.819 ± 0.136
B	0.825 ± 0.126	0.818 ± 0.132	0.819 ± 0.133
Downstream 50%			
Distortion method	Pretext 10%	Pretext 30%	Pretext 50%
b	0.831 ± 0.110	0.824 ± 0.104	0.770 ± 0.115
d	0.864 ± 0.105	0.865 ± 0.102	0.867 ± 0.110
s	0.868 ± 0.109	0.877 ± 0.106	0.866 ± 0.113
r	0.831 ± 0.104	0.824 ± 0.104	0.770 ± 0.111
S	0.864 ± 0.108	0.865 ± 0.112	0.867 ± 0.103
R	0.868 ± 0.106	0.877 ± 0.103	0.866 ± 0.117
D	0.831 ± 0.115	0.824 ± 0.109	0.770 ± 0.115
B	0.864 ± 0.108	0.865 ± 0.108	0.867 ± 0.117
Downstream 100%			
Distortion method	Pretext 10%	Pretext 30%	Pretext 50%
b	0.872 ± 0.094	0.881 ± 0.125	0.868 ± 0.100
d	0.877 ± 0.095	0.883 ± 0.100	0.871 ± 0.091
s	0.865 ± 0.095	0.877 ± 0.089	0.863 ± 0.095
r	0.875 ± 0.097	0.881 ± 0.093	0.879 ± 0.093
S	0.867 ± 0.095	0.870 ± 0.097	0.885 ± 0.094
R	0.875 ± 0.094	0.886 ± 0.099	0.860 ± 0.088
D	0.861 ± 0.111	0.872 ± 0.117	0.865 ± 0.095
B	0.869 ± 0.102	0.874 ± 0.095	0.867 ± 0.098
Overall			
b	$\mathbf{0.850 \pm 0.035}$	S	0.864 ± 0.0222
d	$\mathbf{0.866 \pm 0.020}$	R	0.863 ± 0.024
s	0.862 ± 0.025	D	0.853 ± 0.023
r	$\mathbf{0.866 \pm 0.022}$	B	0.856 ± 0.027

ful for understanding the best pretext distortion method for downstream performance. Table 4.2 shows average and standard deviation of **downstream F1-score performance** of collections of tasks using a specific distortion method. Both simple and complex tasks are included in the average downstream F1-score performance of a distortion method. The drop boxes *(d)* (average F1-score = 0.866 ± 0.020) and rotate circles *(r)* (average F1-score = 0.866 ± 0.022) distortion methods were the two best performing distortion methods. Blur *(b)* (average F1-score = 0.850 ± 0.035) was overall the worst performing distortion method.

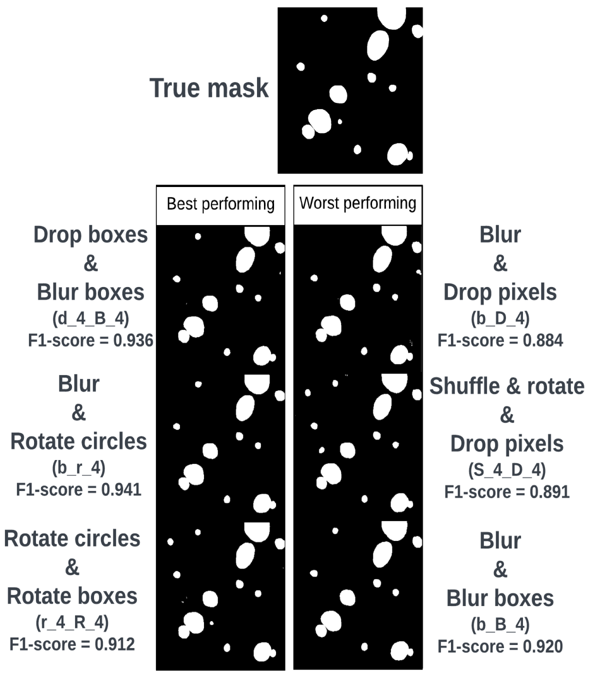

Figure 4.4: Examples of segmentation of best and worst downstream performing pretext tasks models trained on 50% pretext training data and 100% downstream training data.

Figure 4.5 shows the **downstream F1-score performance** of all tasks, both simple and complex, and their ranking from highest to lowest F1-score median. Notably, most of the top performing self-prediction pretext tasks were complex pretext tasks (ranked top 15 / 36). In addition, despite blur *(b)* being the worst distortion method, there are a number of best performing pretext tasks that use the blur distortion method.

Figure 4.4 shows examples of organoid segmentation using the 50% pretext - 100% downstream data amount models from the top three best performing self-prediction pretext tasks and the three worst performing self-prediction pretext tasks. The predicted mask of the best performing models is closer to the true mask with higher F1-scores compared to the predicted masks of the worst performing models. These examples show that the difference in F1-

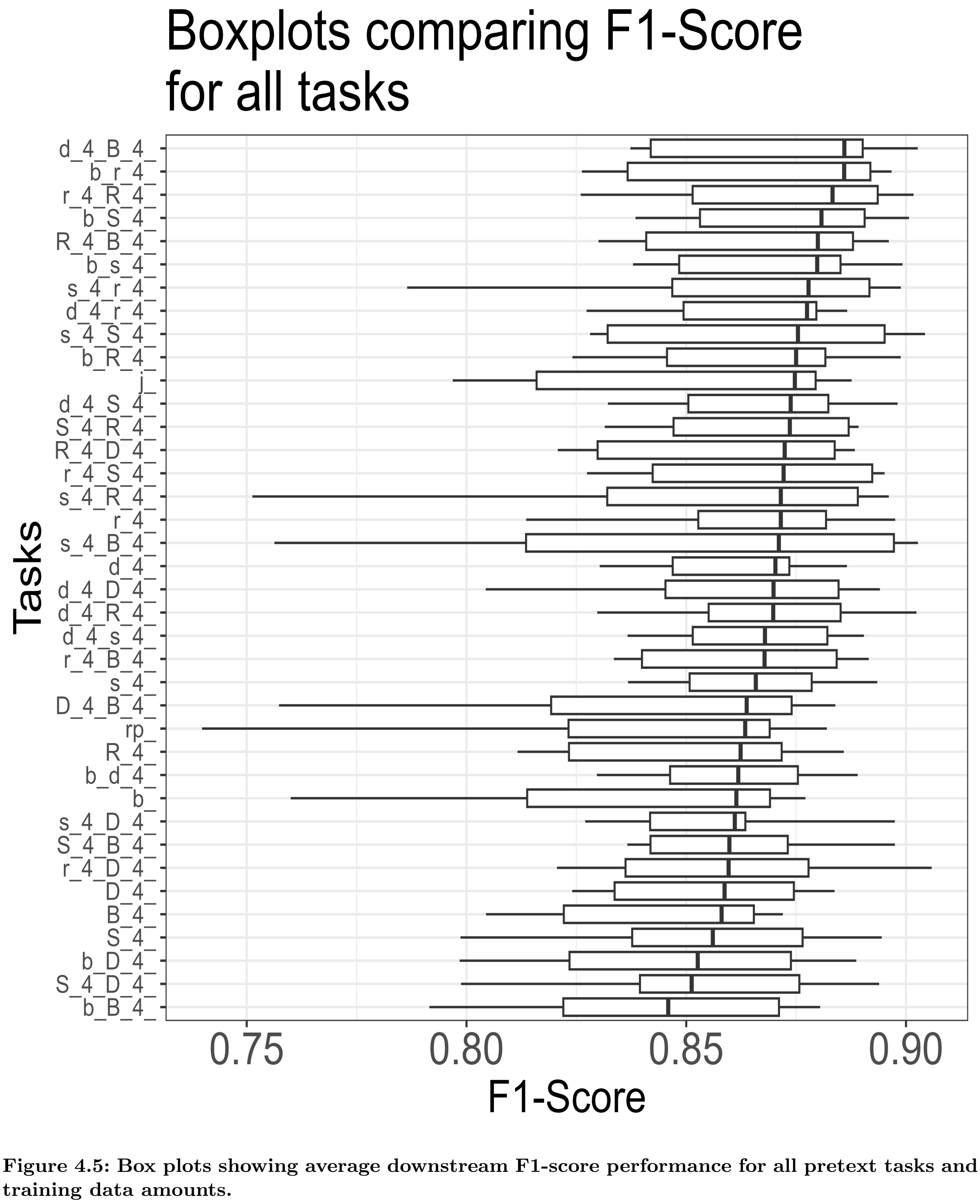

Figure 4.5: Box plots showing average downstream F1-score performance for all pretext tasks and training data amounts.

scores of predicted masks is mostly caused by segmenting organoids that are not present in the true mask as well as not segmenting organoids that are in the true mask. In addition, masks of segmented organoids of the worst performing models are more jagged and less polished.

To summarize, results of Figure 4.1 showed that, although there was a high variability among the pretext models in complexity, complex models were indeed overall more complex for the model to learn to solve compared to simple models.

Figure 4.2, Figure 4.3 and Table 4.1 all showed complex models outperformed simple models on downstream F1-score performance of the segmentation of organoids. Figure 4.5 showed that complex models were ranked higher in median downstream performance compared to simple models. Although this effect was consistent through multiple visualizations, the effect size of the difference in downstream performance was small. In addition, the standard deviations of average downstream performance of simple and complex models were relatively high to the extent that any significant difference cannot be safely concluded.

Figure 4.2 showed no conclusive relation between pretext PSNR performance of simple and complex models and downstream F1-score performance. Due to varying complexity of distortion methods, and therefore their combinations, the only true measure of complexity of the pretext tasks is the pretext PSNR performance. Lower scoring pretext performance suggests a more complex pretext task. Given that there appeared to be no correlation between pretext performance and downstream performance, it cannot be concluded that complexity of the pretext task directly correlates to any difference in downstream performance. Differences between simple and complex downstream performance are therefore most likely due to the effect of (combinations of) distortion methods on learned features, rather than the complexity of the pretext tasks.

4.2 How does the pretext task strategy type affect self-supervised learning of the segmentation of organoids?

This sub-section presents the results of the difference in downstream performance of the previously discussed self-prediction strategy and the innate relationship strategy.

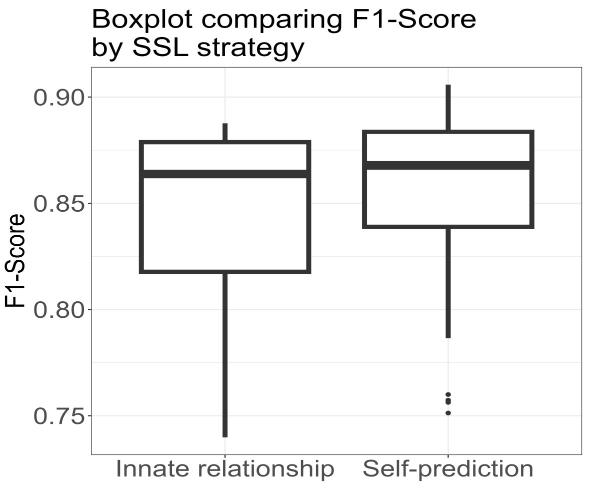

Figure 4.6: Box plots showing average downstream F1-score performance separated by self-prediction models (simple and complex self-prediction models combined) and innate relationship models (Jigsaw *(j)* and Predict rotation angle *(rp)* models combined).

After downstream learning innate relationship models were tested on their downstream F1-score performance. The downstream performance of the self-prediction and innate relationship strategies are compared. Figure 4.6 shows box plots of **downstream F1-score performance** separated by **SSL strategy**. The difference in median appears relatively small, however, 53.7% of self-prediction models perform better than the innate relationship median, where 44.4% of innate relationship models perform better than the self-prediction median. In addition, 46.3% of the self-prediction models perform worse than the innate relationship median and 55.6% of the innate relationship models perform worse than the self-prediction median. The self-prediction strategy (average F1-score = 0.859 ± 0.112) seems to outperform the innate relationship strategy (average F1-score = 0.848 ± 0.118).

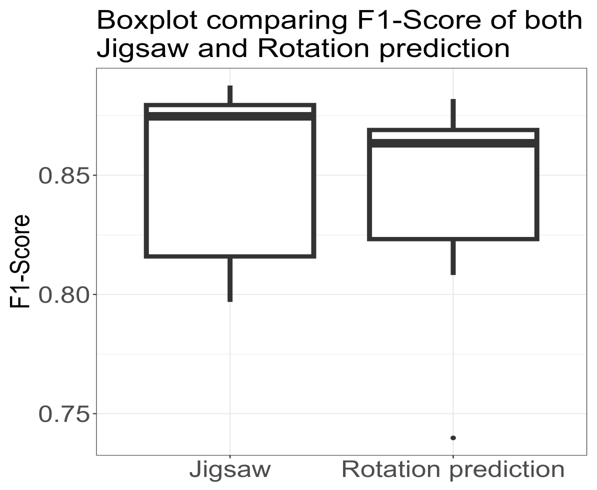

Figure 4.7: Box plots showing average downstream F1-score performance of the Jigsaw *(j)* and Predict rotation angle *(rp)* pretext tasks over all training data amounts.

Table 4.3: Average downstream F1-score performance for the Jigsaw *(j)* and Predict rotation angle *(rp)* pretext tasks. Rows represent the respective pretext task, columns represent the pretext training data amount and both are grouped based on downstream training data amount. The overall block shows average downstream F1-score performance over all training data amounts for both pretext tasks.

Downstream 10%			
Innate relationship	Pretext 10%	Pretext 30%	Pretext 50%
Jigsaw	0.800 ± 0.153	0.809 ± 0.147	0.816 ± 0.125
Rotation prediction	$\mathbf{0.823 \pm 0.141}$	$\mathbf{0.808 \pm 0.142}$	$\mathbf{0.740 \pm 0.175}$
Downstream 50%			
Innate relationship	Pretext 10%	Pretext 30%	Pretext 50%
Jigsaw	0.863 ± 0.093	0.877 ± 0.095	0.875 ± 0.097
Rotation prediction	0.864 ± 0.095	0.869 ± 0.095	0.863 ± 0.119
Downstream 100%			
Innate relationship	Pretext 10%	Pretext 30%	Pretext 50%
Jigsaw	0.879 ± 0.102	0.888 ± 0.103	0.880 ± 0.098
Rotation prediction	0.881 ± 0.111	0.882 ± 0.114	0.843 ± 0.119
Overall			
Jigsaw	$\mathbf{0.854 \pm 0.036}$	Rotation prediction	$\mathbf{0.842 \pm 0.047}$

Figure 4.7 shows the average **downstream F1-score performance** for the Jigsaw puzzle and the rotation angle prediction pretext tasks. Models trained on the Jigsaw puzzle *(j)* (median F1-score = 0.875) outperformed the models trained on Predict rotation angle *(rp)* (median F1-score = 0.863).

Similarly, Table 4.3 shows the average and standard deviation of **downstream F1-score performance** for all pretext and downstream data amounts. Again, models trained on the Jigsaw puzzle *(j)* (average F1-score = 0.854 ± 0.036) outperformed models trained on Predict rotation angle *(rp)* (average F1-score = 0.842 ± 0.047) on average downstream performance.

The box plot showing all tasks, Figure 4.5, shows that both Jigsaw *(j)* and Predict rotation angle *(rp)* performed decent. And again, the Jigsaw *(j)* (ranked 11/38) clearly outperformed Predict rotation angle *(rp)* (ranked 26/38) in terms of ranking.

To summarize, Figure 4.7 and Table 4.3 showed that the Jigsaw puzzle *(j)* outperformed Predict rotation angle *(rp)*. Figure 4.6 showed that the self-prediction strategy outperformed the innate relationship strategy on average downstream F1-score performance. Self-prediction had better high performing models, where innate relationship had worse low performing models. In addition, Figure 4.5 showed that while both Jigsaw and Predict rotation angle models performed decent, they still had seemingly worse performance compared to self-prediction models.

4.3 What effect does the amount of training data, for both self-supervised and supervised learning, have on the quality of organoid segmentation in relation to the complexity of the pretext task and the pretext task strategy type?

This sub-section presents the results of the effects of different amounts of pretext and downstream training data on downstream performance. Here, the difference between simple and complex self-prediction performance as well as self-prediction and innate relationship strategy performance in relation to different amounts of training data is discussed.

Both simple-complex self-prediction models and innate relationship models were trained on varying amounts of pretext and downstream training data. For each pretext task, pretext training data amount and downstream training data amount the downstream F1-score performance of the model was tested. Figure 4.8 shows **downstream F1-score**

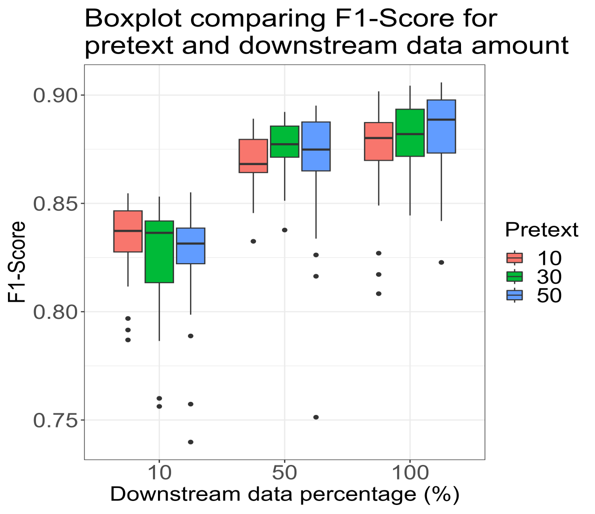

Figure 4.8: Box plots showing downstream F1-score performance for all pretext tasks on the y-axis separated by downstream training data amount on the x-axis and pretext training data amount for the color. Here, red, green and blue represent 10%, 30% and 50% pretext training data amount respectively.

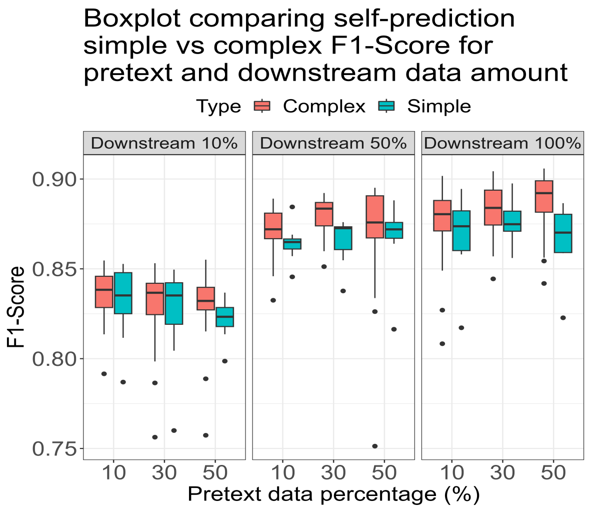

Figure 4.9: Box plots showing average downstream F1-score performance on the y-axis separated by pretext training data amounts on the x-axis, grouped by downstream training data amounts and lastly, the colors blue and red representing simple and complex models respectively.

performance on the y-axis and **downstream training data amount** on the x-axis, further separated by **pretext training data amount** using color. These box plots show that models performed better with more **downstream training data** regardless of **pretext training data**. 10% downstream training data (average F1-score = 0.825 ± 0.129), 50% downstream training data (average F1-score = 0.872 ± 0.109) and 100% downstream training data (average F1-score = 0.878 ± 0.098) have continuously increasing downstream F1-score performance. In addition, adding more pretext training data benefited downstream performance on the condition that there was a decent amount of downstream training data available. With 10% downstream training data, 10% pretext training data (average F1-score = 0.834 ± 0.126), 30% pretext training data (average F1-score = 0.826 ± 0.130) and 50% pretext training data (average F1-score = 0.815 ± 0.132) average downstream F1-score performance continuously decreased. This suggests that an increase in pretext training data, given too little downstream training data, is detrimental to downstream performance.

Figure 4.9 shows self-prediction models with **downstream F1-score performance** on the y-axis and **pretext training data amount** on the **x-axis**, separated by **downstream training data amount** into multiple plots, and lastly, separated into **simple and complex** using color. This Figure shows the same effect previously discussed. A large amount of pretext training data is detrimental given little downstream training data, see Table 4.1. For all pretext and downstream training data combinations models trained on complex tasks outperformed simple tasks, again, see Table 4.1. This adds credibility to the claim that complex models outperformed simple models, as this effect is constant over all training data amounts.

Similarly, Figure 4.10 shows innate relationship and self-prediction models with **downstream F1-score performance** on the y-axis and **pretext training data amount** on the x-axis, separated by **downstream training data amount** into multiple plots, and lastly, separated by **SSL strategy** using color. Models trained using the self-prediction strategy outperformed models trained using the innate relationship for all pretext and downstream training data combinations. Adding credibility to the claim that self-prediction is a better suited SSL

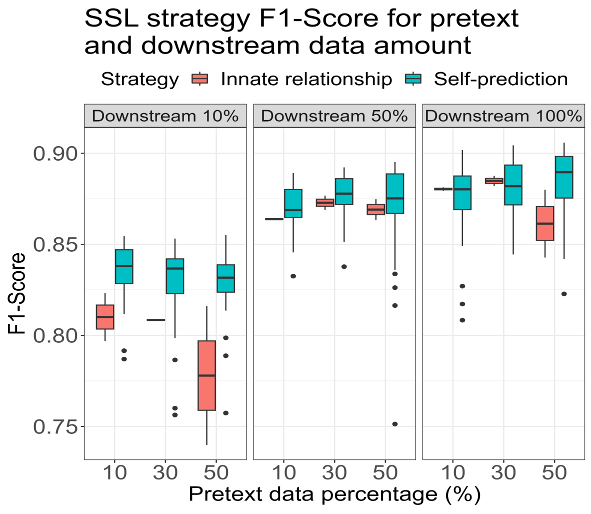

Figure 4.10: Box plots showing average downstream F1-score performance on the y-axis separated by pretext training data amounts on the x-axis, grouped by downstream training data amounts and lastly, colors blue and red representing self-prediction and innate relationship models respectively.

strategy for organoid segmentation compared to the innate relationship strategy. Notably, models trained using the innate relationship strategy performed significantly worse than models trained using the self-prediction strategy when there is little downstream training data available. Table 4.3 shows that with 10% downstream training data the Jigsaw puzzle *(j)* downstream performance increased with more pretext training data (average F1-score = 0.800 to 0.816), but the Predict rotation angle *(rp)* downstream performance sharply declined with more pretext training data (average F1-score = 0.823 to 0.740).

To summarize, results showed that complex models outperformed simple models for all training data amounts, as shown in Figure 4.9. The consistency of complex models outperforming simple models adds credibility to the claim that complex models are better suited for pretext learning with the intention of organoid segmentation compared to simple models. Results also showed that self-prediction downstream performance mostly outperformed innate relationship downstream performance for all training data amounts, as shown in Figure 4.10. The consistency of the self-prediction strategy outperforming the innate relationship strategy adds credibility to the claim that the self-prediction strategy of pretext learning is better suited for organoid segmentation. Lastly, results shown in Figure 4.8, Figure 4.9 and Figure 4.10 all show that more pretext training data is beneficial to downstream performance given a minimum amount of downstream training data. For 10% downstream training data, more pretext training data negatively impacted downstream performance. Especially so for the Predict rotation angle *(rp)* innate relationship pretext task.

5 Discussion

This section provides a discussion of the found results, the limitations of this research and several proposals for possible future research.

5.1 Discussion

Given that there appeared to be no correlation between pretext performance and downstream performance, it cannot be concluded that complexity of the pretext task directly correlates to any difference in downstream performance. Results show that there is a consistent difference in performance of self-prediction complex models compared to simple models, despite there being no obvious correlation between pretext PSNR performance and downstream F1-score performance. The perceived difference in downstream performance between simple and complex models could be attributed to requiring a wider variety of learned features to solve the pretext task. As complex tasks require the reconstruction of two different distortion methods, it would require different features to solve. This in turn perhaps creates a more robust latent feature representation, which, in turn, benefits downstream performance.

In addition, results suggest that despite drop boxes *(d)* and rotate circles *(r)* being the best performing distortion methods, it cannot be concluded that a combination of the two would result in the best pretext task. The found optimal pretext task, drop boxes **(d)**+blur boxes *(B)*, is not a combination of the two best performing distortion methods. Rather, it uses the blur boxes *(B)* distortion method, which is one of the worst performing distortion methods. Pretext task, data type and downstream task specific research is required to ensure choosing the pretext task that has pretext learned features well suited to the data type and downstream task.

Furthermore, results suggest that more pretext training data is beneficial as long as there is a decent amount of downstream training data available. This could be the effect of over fitting on the pretext task. The model is specialized in the pretext task to the extent that the little downstream training data available is not enough to adjust to the downstream task.

5.2 Limitations

Due to computational limitations in combination with time restraints there were a number of drawbacks to this research. For pretext learning not all levels of pretext training data were examined, instead focusing on 10%, 30% and 50% of the total available pretext training data. For similar reasons, downstream learning did not evaluate all rotations of the 5-fold cross validation. Opting for 3 rotations instead of the full 5 rotations. The validation data division was still in line with 5-fold cross validation. Additionally, the scope of this research was purely focused on the segmentation of organoids. All results are therefore only linked to organoid research and cannot be safely extrapolated to other research domains. Lastly, this research did not examine the generative and contrastive SSL strategies. Therefore, the findings only relate to the self-prediction and innate relationship strategies. It is uncertain how generate and contrastive SSL strategies would compare.

5.3 Future research

Proposed future research would be to evaluate other SSL strategies. As previously mentioned, this research does not focus on generative and contrastive SSL strategies. Therefore, it is uncertain how these strategies would compare to the self-prediction and innate relationship strategies.

Despite this research providing more insight into using a SSL approach for organoid research, it is unclear how the performance of the examined pretext tasks would compare when used on different biomedical image analyses and different data types.

Lastly, the SSL approach provides a method for creating data type specific pre-trained models. Given that for a specific data set there can be multiple use cases, it would prove interesting how performance compares when training separate downstream tasks from the same pretext model.

In the same line, training from the same pretext model provides the opportunity to implement a multi-downstream task set-up. Research into the effect of combining multi-task learning and SSL could prove worthwhile with creating models than can perform more than one downstream task.

References

Aboalayon, K., Faezipour, M., Almuhammadi, W., & Moslehpour, S. (2016, 08). Sleep stage classification using eeg signal analysis: A comprehensive survey and new investigation. *Entropy*, *18*. doi: 10.3390/e18090272

Aljuaid, A., & Anwar, M. (2022, May 17). Survey of supervised learning for medical image processing. *SN Computer Science*, *3*(4), 292. Retrieved from https://doi.org/10.1007/s42979-022-01166-1 doi: 10.1007/s42979-022-01166-1

Bruch, R., Rudolf, R., Mikut, R., & Reischl, M. (2020). Evaluation of semi-supervised learning using sparse labeling to segment cell nuclei. *Current Directions in Biomedical Engineering*, *6*(3), 398–401. Retrieved 2023-06-21, from https://doi.org/10.1515/cdbme-2020-3103 doi: doi: 10.1515/cdbme-2020-3103

Caicedo, J. C., Goodman, A., Karhohs, K. W., Cimini, B. A., Ackerman, J., Haghighi, M., … Carpenter, A. E. (2019, Dec 01). Nucleus segmentation across imaging experiments: the 2018 data science bowl. *Nature Methods*, *16*(12), 1247-1253. Retrieved from https://doi.org/10.1038/s41592-019-0612-7 doi: 10.1038/s41592-019-0612-7

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. *ICML 2020-1*.

Clevers, H. (2016, June). Modeling development and disease with organoids. *Cell*, *165*(7), 1586–1597.

Faragallah, O. S., El-Hoseny, H., El-Shafai, W., El-Rahman, W. A., El-Sayed, H. S., El-Rabaie, E.-S. M., … Geweid, G. G. N. (2021). A comprehensive survey analysis for present solutions of medical image fusion and future directions. *IEEE Access*, *9*, 11358-11371. doi: 10.1109/ACCESS.2020.3048315

Gidaris, S., Singh, P., & Komodakis, N. (2018). Unsupervised representation learning by predicting image rotations. *CoRR*, *abs/1803.07728*.

Haja, A., Brouwer, E., & Schomaker, L. (2021). Self-supervised versus supervised training for segmentation of organoid images. *Elsevier*.

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. *IEEE*.

Huang, S.-C., Pareek, A., Jensen, M., Lungren, M. P., Yeung, S., & Chaudhari, A. S. (2023, Apr 26). Self-supervised learning for medical image classification: a systematic review and implementation guidelines. *npj Digital Medicine*, *6*(1), 74. Retrieved from `https://doi.org/10.1038/s41746-023-00811-0` doi: 10.1038/s41746-023-00811-0

Jaiswal, A., Babu, A. R., Zadeh, M. Z., Banerjee, D., & Makedon, F. (2020). A survey on contrastive self-supervised learning. *CoRR*, *abs/2011.00362*.

Kingma, D., & Ba, J. (2014, 12). Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Li, Y., Chen, J., Li, F., Fu, B., Wu, H., Ji, Y., … Zheng, W. (2022). Gmss: Graph-based multi-task self-supervised learning for eeg emotion recognition. *IEEE*.

Lotter, W., Diab, A. R., Haslam, B., Kim, J. G., Grisot, G., Wu, E., … Sorensen, A. G. (2021). Robust breast cancer detection in mammography and digital breast tomosynthesis using an annotation-efficient deep learning approach. *Nature medicine*, *27*(2), 244-249. doi: https://doi.org/10.1038/s41591-020-01174-9

Louey, A., Hernández, D., Pébay, A., & Daniszewski, M. (2021). Automation of organoid cultures: Current protocols and applications. *SLAS Discovery*, *26*(9), 1138-1147. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2472555222067569` doi: https://doi.org/10.1177/24725552211024547

Mergenthaler, P., Hariharan, S., Pemberton, J. M., Lourenco, C., Penn, L. Z., & Andrews, D. W. (2021, February). Rapid 3D phenotypic analysis of neurons and organoids using data-driven cell segmentation-free machine learning. *PLoS Comput. Biol.*, *17*(2), e1008630.

Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., & Terzopoulos, D. (2020). Image segmentation using deep learning: A survey. *CoRR*, *abs/2001.05566*.

Noroozi, M., & Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. *CoRR*, *abs/1603.09246*.

Ranjbaran, A., & Nazemi, A. (2023). A survey on organoid image analysis platforms. *arXiv*.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *CoRR*, *abs/1505.04597*.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, *abs/1609.04747*.

Salim, A. A., Ali, S. H., Hussain, A. M., & Ibrahim, W. N. (2021, August). Electroencephalographic evidence of gray matter lesions among multiple sclerosis patients: A case-control study. *Medicine (Baltimore)*, *100*(33), e27001.

Shah, D. (2023). Cross entropy loss: Intro, applications, code. *V7Lab*. Retrieved from `https://www.v7labs.com/blog/cross-entropy-loss-guide`

Shi, T., Jiang, H., & Zheng, B. (2022). C2ma-net: Cross-modal cross-attention network for acute ischemic stroke lesion segmentation based on ct perfusion scans. *IEEE transactions on bio-medical engineering*, *69*(1), 108-118. doi: https://doi.org/10.1109/TBME.2021.3087612

Wallace, B., & Hariharan, B. (2020). Extending and analyzing self-supervised learning across domains. *CoRR*, *abs/2004.11992*.

Wang, Z., Bovik, A., Sheikh, H., & Simoncelli, E. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600-612. doi: 10.1109/TIP.2003.819861

Zang, X., Bascom, R., Gilbert, C., Toth, J., & Higgins, W. (2016). Methods for 2-d and 3-d endobronchial ultrasound image segmentation. *IEEE transactions on bio-medical engineering*, *63*(7), 1426-1439. doi: https://doi.org/10.1109/TBME.2015.2494838