Technical University of Applied Sciences Würzburg-Schweinfurt (THWS)
Faculty of Computer Science and Business Information Systems

Master Thesis

Determination of Drug Efficacy on Pancreatic Tumor 3D Spheroidal Tissues

submitted to the Technical University of Applied Sciences Würzburg-Schweinfurt in the Faculty of Computer Science and Business Information Systems to complete a course of studies in MAI

Bibin Babu

Submitted on: January 15th 2025

Initial examiner: Prof. Dr. Magda Gregorová
Secondary examiner: Prof. Dr. Jan Hansmann

Abstract (en)

Pancreatic tumor treatment is hindered by the intricate nature of tumors and their diverse microenvironments. This complexity necessitates an exploration into identifying optimal drug combinations and concentrations tailored to each patient's specific tumor characteristics. This thesis aims to assess drug efficacy by ranking these various drug combinations and concentrations. The ranking is based on features extracted from bright-field microscopy images of three-dimensional tumor tissue models using representation learning. The core challenge is to learn robust features that accurately characterize alterations in these tumor tissue models induced by drug application over time. This research seeks to develop a standardized and effective approach for evaluating drug efficacy, potentially improving treatment outcomes for pancreatic tumor patients.

Abstract (de)

Die Behandlung von Bauchspeicheldrüsentumoren wird durch die komplexe Natur der Tumore und ihre vielfältigen Mikroumgebungen behindert. Diese Komplexität erfordert eine Untersuchung zur Identifizierung optimaler Medikamentenkombinationen und -konzentrationen, die auf die spezifischen Tumoreigenschaften jedes Patienten zugeschnitten sind. Diese Masterarbeit zielt darauf ab, die Wirksamkeit von Medikamenten zu bewerten, indem sie diese verschiedenen Medikamentenkombinationen und -konzentrationen einstuft. Die Bewertung basiert auf Merkmalen, die aus Helligkeitsmikroskopiebildern dreidimensionaler Tumorgewebsmodelle mittels Repräsentationslernen extrahiert werden. Die zentrale Herausforderung besteht darin, robuste Merkmale zu erlernen, die Veränderungen in diesen Tumorgewebsmodellen genau charakterisieren, die durch die Anwendung von Medikamenten über Zeit induziert werden. Diese Forschung zielt darauf ab, einen standardisierten und effektiven Ansatz zur Bewertung der Medikamenteneffizienz zu entwickeln, der möglicherweise die Behandlungsergebnisse für Patienten mit Bauchspeicheldrüsentumoren verbessert.

Acknowledgment

Thank all who believed in me. Danke.

Würzburg, on 15.01.2025

_____

Bibin Babu

Contents

List of Figures

List of Tables

1 Introduction

Pancreatic tumor presents a significant challenge in terms of treatment due to its heterogeneous nature and the mutations that occur during its progression within the human body. Clinicians rely on case studies, human trials, and their own expertise gained from past patient treatments to select drugs for new patients. However, this approach is often based on trial and error, with varying outcomes. Patients may experience either successful treatment or severe side effects such as hair loss and damage to other organs. Since each patient's tumor cells exhibit unique characteristics influenced by factors such as age and genetics, treatments that have worked for one patient may not be effective for another. Consequently, clinicians may need to change the prescribed drugs or try different combinations, which can lead to delays and increased risks for the patient, including mortality.

In light of these challenges researchers at Fraunhofer Translational Center for Regenerative Therapy TLZ-RT Wuerzburg, propose a vision for the future: cultivating multiple three-dimensional tumor tissue models for each patients in the lab using biopsy samples and studying the efficacy of drugs on these three-dimensional tumor tissue models first.*(Note: In this thesis, "3D tumor tissue models or tumor tissue models" refers to physical, lab-grown tissues and not computational or AI models.)* By conducting drug development experiments and analyses on these tissue models, they aim to find the optimum or best drug combination tailored to each patient's specific tumor characteristics.This approach can minimize direct side effects on human patients and reduce the time needed to select the most effective personalized treatment, thereby decreasing the risk of that patient's mortality. Additionally, it can significantly reduce the cost and time of preclinical testing in the drug development process. Ultimately, these information obtained from drug efficacy assessment experiments can inform clinicians' decisions, enabling them to select the most effective drug combination before administering it to the patient.

As a proof of concept, The Fraunhofer TLZ-RT Würzburg laboratory utilizes a modular dual-arm robot-based system [3], equipped with incubators and bioreactors (see Figure 1.1 and Figure 1.2) under physiological conditions to study drug efficacy for the long-term culture of these three-dimensional tumor tissue models. One advantage of this platform is its ability to capture bright-field microscopy images (detailed explanation in Chapter 5) of 3D tumor tissue models using a customized microscope setup integrated into a robotic platform, providing flexibility in image acquisition to meet experimental requirements.

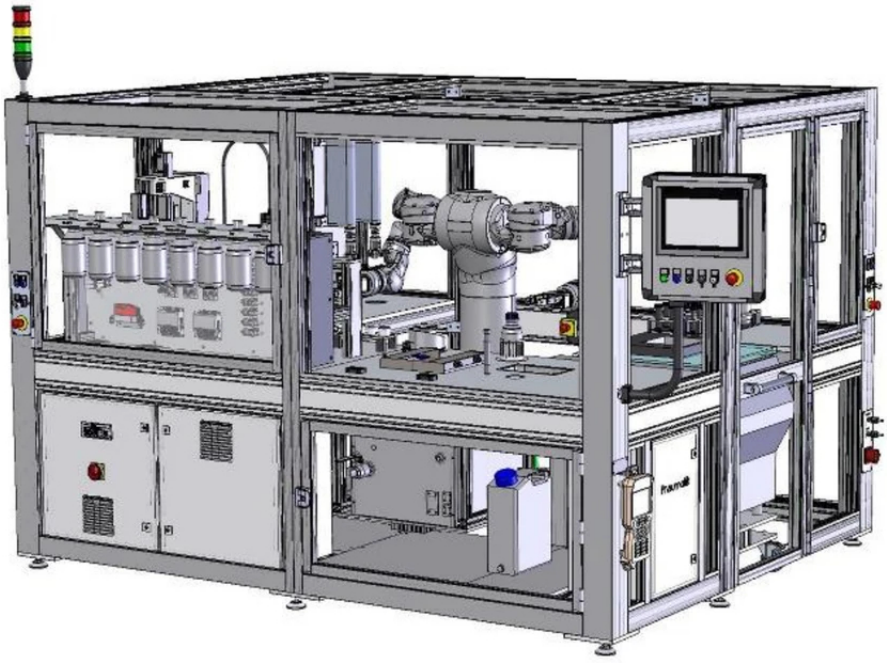

Figure 1.1: Robo platform

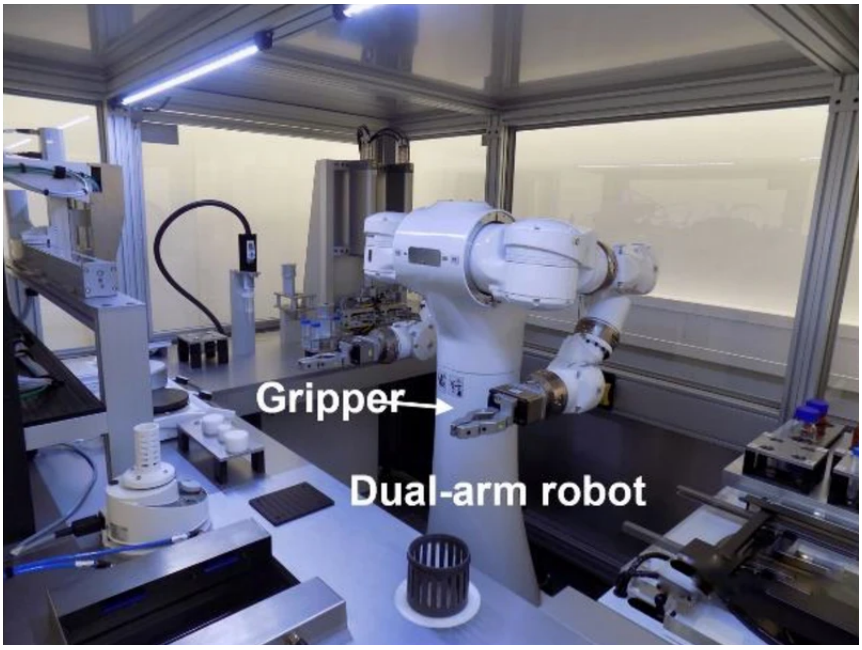

Figure 1.2: Dual-arm robot

Although the vision for the future is to simulate the identical interaction environment of drugs with tumor cells as it occurs in the human body, current technology has not yet achieved this. The current three-dimensional tumor tissue models developed in the lab do not fully resemble real pancreatic tumor cells found in the human body. These 3D tumor tissue models only contain pure tumor tissues, whereas real human pancreatic tumor cells exist within a complex microenvironment comprising tumor cells, blood vessels, other tissues, and various cell types. Fortunately, if human body tumor cells can be replicated in the lab in the future, the techniques currently used to study bright-field microscopy images will still be applicable. However, the fact that bright-field microscopy images are two-dimensional limits the ability to perform a comprehensive analysis of the drug's impact on the entire 3D structure of the cultivated tumor tissue models. Despite this limitation, this research serves as a valuable starting point for studying drug efficacy in a controlled environment.

Alternatives to bright-field microscopy images include 3D fluorescence microscopy and luminescent cytotoxicity assays. However, both methods are invasive. Fluorescent molecules tend to generate reactive chemical species under illumination, enhancing phototoxic effects. This chemical reaction with the 3D tumor tissue model may alter its structure, making it not suitable to isolate the drug's effect over time. Similarly, luminescent cytotoxicity assays result in a dead culture, rendering them unsuitable for longitudinal studies. Additionally, both meth-

ods require removing the well plate from the isolated culture environment for extended periods, making the samples susceptible to external environmental factors. For instance, in fluorescence microscopy, cells are particularly vulnerable to phototoxicity from short wavelength light. In contrast, bright-field microscopy images are non-invasive, allowing continuous culture and the possibility of creating time series of images to study dynamic changes. Therefore, we rely on bright-field microscopy images to study the time-evolutionary effects of drugs.

1.1 Laboratory Setup

3D tumor tissue models are cultured in well plates containing 96 wells, each providing a nutrient medium that allows them to maintain their tissue-specific functions in vitro. Although each plate can yield 96 pure 3D tumor tissue models, the edge effect is accounted for, where outer wells may be exposed to variable conditions such as temperature fluctuations, increased evaporation rates, and other environmental factors. Consequently, we restrict our analysis to the 60 inner wells per plate as in figure 1.3, adhering to standard procedures to ensure consistent and reliable experimental data.

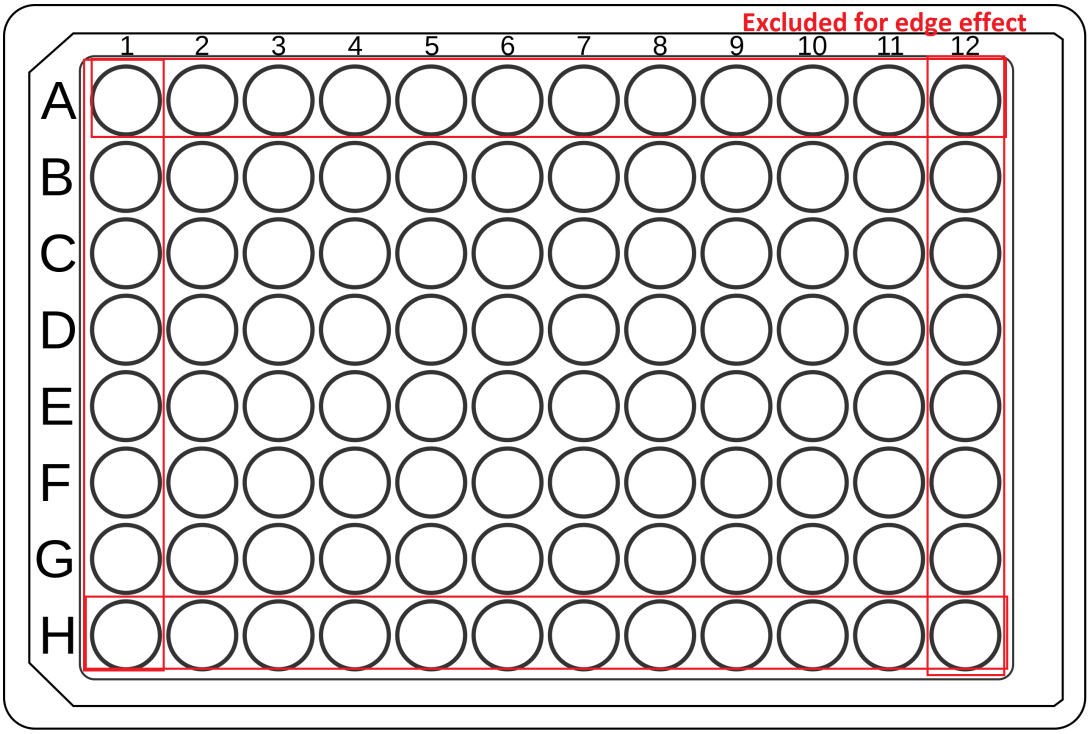

Figure 1.3: A well plate containing 96 wells where rows A, H and columns 1, 2 are excluded due to edge effects.

Based on the drug concentration applied to 3D tumor tissue models, the bright-field microscopy images we capture can be categorized into three:

Images of

1. Control (0 percentage drug applied)nFor easiness, we refer to this category as “Untreated”

2. Single concentration (theoretically recommended single concentration of drug treatment) For easiness, we refer to this category as “Single dose”

3. Drug screening: different drug combinations and concentrations used for experimental study of drug efficacy, which may or may not result in the killing of surrounding non-tumor cells in the human body with potential side effects. For easiness, we refer to this category as “Drug screened”

We apply single dose and drug screened combinations in different well plate settings as illustrated inFigure 1.4 and figure 1.5

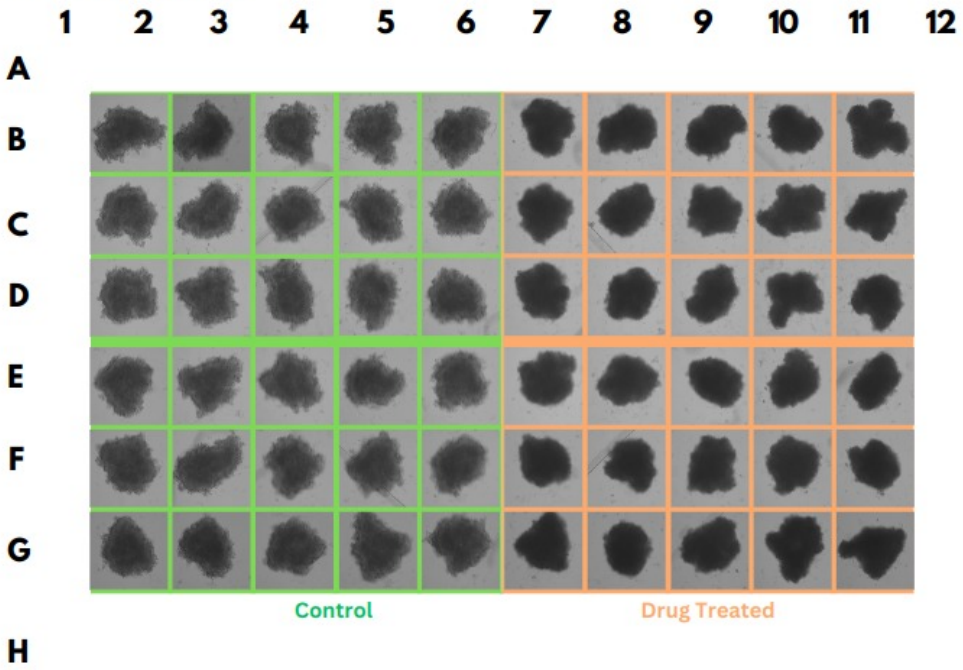

Figure 1.4: Well plate setup for the single-dose experiment where the left half remains untreated and the right half is treated with a single drug concentration. This image was taken three days after drug application.

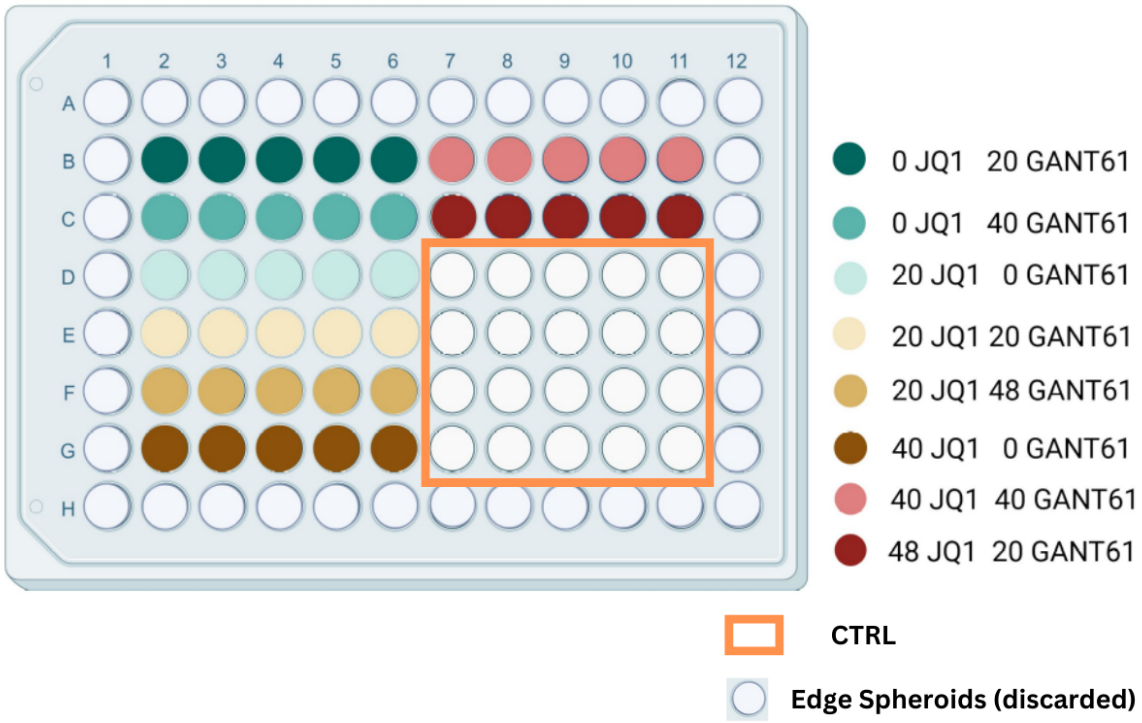

Figure 1.5: Well plate setup for the drug screening experiment where the majority of tumor tissues are treated with different combinations of drug concentrations (multi-colored wells), while some are left untreated (white wells bounded by orange box).

1.6 Illustrates the flow chart of time evolutoion of 3D tumor tissues. The 3D tumor tissue models develop in the wellplate progressively from day 1 reaching their maximum cancerous state by day 7, at which point the drug is administered. By day 10, the drug's effect on the cancerous tissue is expected to peak as nutrient availability gradually decreases, causing the tumor to diminish. Therefore, to isolate the drug's effects, changes in tumor tissue deterioration are assessed on the peak effect day, i.e., day 10, in accordance with established medical protocols and previous research findings.

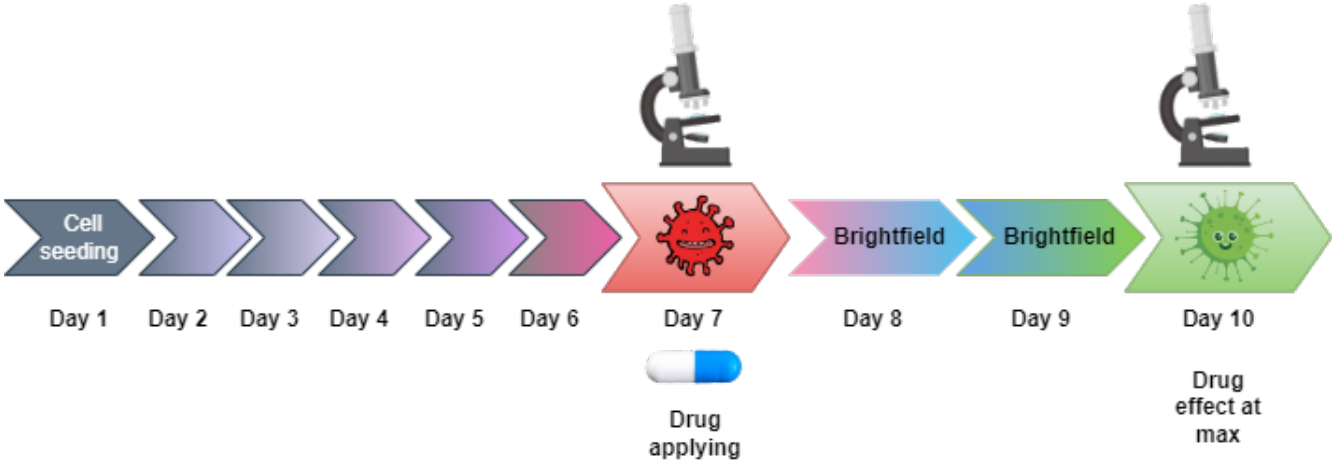

Figure 1.6: Illustrates the flow chart of time evolutoion of 3D tumor tissues.

2 Motivation

We assess the efficacy of the drug by comparing the changes it induces in the bright-field microscopy images over a period of time. The current methods to differentiate these changes involve studying the alterations from day 7 to day 10. These changes are typically observed in three main parameters:

1. Size/Area

2. Circularity/Diameter/Perimeter

3. Pixel intensity or color change

These parameters serve as human-interpretable metrics for assessing the efficacy of the drug. However, these manual methods are inherently limited to observable features, which may overlook subtle or complex patterns within the data.

Representation learning techniques, such as those based on deep learning, can be employed to extract features that are not immediately interpretable by humans. These methods leverage neural networks to learn high-dimensional feature representations directly from the images, capturing intricate patterns, relationships, and variations that may correspond to biological phenomena. Furthermore, these learned representations enable a standardized approach to analysis. Unlike manual assessments, which can vary due to human subjectivity, representation learning models produce consistent results once trained.

This is the focus of my master thesis, where I take on the crucial role of developing and applying representation learning techniques to uncover these hidden patterns and standardize the analysis process.

3 Research Objective and Questions

Research objective

This thesis aims to assess drug efficacy by ranking different drug combinations and concentrations on lab-cultivated pancreatic tumor tissue models. The ranking is based on features extracted from bright-field microscopy images of these tumor tissue models using SimCLR self supervised learning. Since we lack ground truth labels, ranking to get exact efficacy value and evaluation of our ranking system to exact drug efficacy metric will be out of focus, instead we rank the images as a transition from untreated to single dose images or untreated to explod images. This is our objective.

My plan is to rewrite the introduction to include why we use SimCLR as self supervised learning for learning representation by including : basically in simclr the loss function is defined such a way that it reduce the cosine distance between the augmentations derived from one image and no penalisation for pushing or no pushing away of augmentation from one image to augmentation from another image. this is perfect in our case where we don't want to push away augmented images derived from different images, especially when we don't know if those images have same drug efficay or not.

Doubt: Should I need to answer the below questions here it self or can I answer as i write through out the the whole section?

Research Questions

1. Can unsupervised learning be used to tackle this ranking problem, given the lack of labels as our primary challenge?

2. Can we learn latent features that capture the alterations induced in tumor tissue models by drug application from bright-field microscopy images?

3. Will these features effectively establish a ranking of drug efficacy?

4. What methodologies and frameworks can be employed to extract and learn these hidden representations efficiently?

5. What could be reasonable metrics, such as L2 loss or cosine similarity, to support the relative assessment of drug efficacy?

6. Do we actually need to learn latent features to establish an effective ranking, or is it possible to achieve this directly using the images themselves?

8. What should be done if the same drug concentration has different effects on cancer cells across different experiments? (For instance, we may not know whether the observed differences are due to real efficacy or visual effects, such as debris.)

9. How do we deal with the position change in the image collection of day 10 if we want to use prediction model as one of our ranking strategy?

10. How do we incorporate the fact about brightness/blur change in the image collection due to environmental or microscope variations?

11. Will strong data augmentation on medical grayscale images improve performance, or will it degrade it due to sensitivity to the original distribution?

12. If yes, If we reduce the intensity of augmentation, will it still be able to learn the features effectively?

13. Does data augmentation function as a tool for learning invariant features, or does it pri-

marily act as a mechanism to increase the sample size, enabling the model to see all possible distributions?

4 Literature Review

You don't need to read this chapter as I need to change this whole

Base neural network architecture for representation learning. Learning visual representations of medical images, such as X-rays (radiographic images) and bright-field microscopy images, is crucial for medical image understanding. However, progress in this area has been hindered by the heterogeneity and complexity of subtle features in these images, especially when they don't have labels. Existing work often relies on fine-tuning weights transferred from ImageNet pretraining (Wang et al., 2017 [11] ; Esteva et al., 2017 [4] ; Irvin et al., 2019 [7]), which is suboptimal due to the drastically different characteristics of medical images. Recent studies have shown promising results using unsupervised contrastive learning on natural images, but these methods have limited effectiveness on medical images because of their high inter-class similarity.

To address these challenges, researchers have proposed various innovative approaches. ConVIRT [13] offers an alternative unsupervised strategy for learning medical visual representations by exploiting naturally occurring paired descriptive text. This method introduces a new approach to pretraining medical image encoders using paired text data via a bidirectional contrastive objective between the two modalities. It is domain-agnostic and requires no additional expert input. However, given the absence of specific paired text data for our image dataset, ConVIRT does not offer a solution tailored to our specific problem.

The contrastive loss used in ConVIRT is derived from the SimCLR [2] self supervised learning framework. SimCLR learns representations by maximizing agreement between differently augmented views of the same data example via a contrastive loss in the latent space. The framework consists of a neural network base encoder that extracts representation vectors from augmented

data examples. The framework allows for various choices of network architecture without any constraints. The authors opt for simplicity and adopt ResNet, introducing a learnable nonlinear transformation between the representation and the contrastive loss to substantially improve the quality of the learned representations. However, these methods require careful treatment of negative pairs, typically relying on large batch sizes to retrieve them. Additionally, their performance is highly dependent on the choice of image augmentations. BYOL (Bootstrap Your Own Latent) [5] addresses these limitations by using an architecture with online and target neural networks, which does not require negative pairs and is more robust to the choice of image augmentations compared to contrastive methods.

While SimCLR has achieved impressive success in the computer vision field, directly applying it to the time series domain often yields poor performance due to its data augmentation and feature extractor not being tailored to the temporal dependencies inherent in time series data. To address this limitation and to obtain high-quality representations of univariate time series, [12] proposed TimeCLR, a framework that combines the strengths of Dynamic Time Warping (DTW) and InceptionTime. Drawing inspiration from the DTW-based k-nearest neighbor classifier, they introduced DTW data augmentation. This technique generates phase shifts and amplitude changes targeted by DTW, preserving the time series structure and feature information. By integrating the advantages of DTW data augmentation and InceptionTime, TimeCLR method extends SimCLR and adapts it effectively to the time series domain. [9] conducted a comprehensive comparative analysis between contrastive and generative self-supervised learning methods for time series data, focusing specifically on SimCLR and MAE (Masked Autoencoder). They observed that, overall, MAE tends to converge more rapidly and delivers impressive performance, particularly when the fine-tuning dataset is relatively small (around 100 samples). However, in scenarios with larger datasets, SimCLR demonstrates a slight but consistent outperformance over its generative counterparts.

Another recent alternative study for self-supervised visual representation is DINO [1], which can be also interpreted as a form of self-distillation with no labels. DINO provides new properties to Vision Transformers that stand out compared to convolutional networks.

SupCon [8] extends the self-supervised batch contrastive approach to the fully-supervised setting, allowing us to effectively leverage label information. Clusters of points belonging to the

same class are pulled together in embedding space, while simultaneously pushing apart clusters of samples from different classes. The drug applied to tumor samples could be used as labels for the bright-field microscopy images.

5 Data Description

5.1 Data set

As explained in the chapter 1, the dataset consists of images taken by bright-field microscopy, which operate as follows:

The sample is illuminated by transmitted white light (i.e., light is projected from below and observed from above). The contrast in the image is created due to the reduction in light intensity as it passes through denser regions of the sample, where more light is absorbed or scattered. This results in a typical bright-field microscopy image where the sample appears darker against a bright background, giving the technique its name. In our case, the 3D tumor model appears as a dark grayish structure on a bright background. For simplicity, bright-field microscopy images will be referred to as 'images'.

Since the samples are cultivated using robotic arms, the process sometimes fails to replicate the natural shapes and patterns that typically occur when laboratory personnel manually cultivate the samples. To ensure the dataset's quality and consistency, Dalia pre-processed the images through a machine learning model to filter out invalid ones. By 'invalid images' , we mean those that variates from the expected morphology of the tumor tissues as cultivated by lab personnel. These images are typically elongated either in height or width as shown in second image of figure 5.1. I collected these filtered images via USB and Google Drive in their original TIFF format.

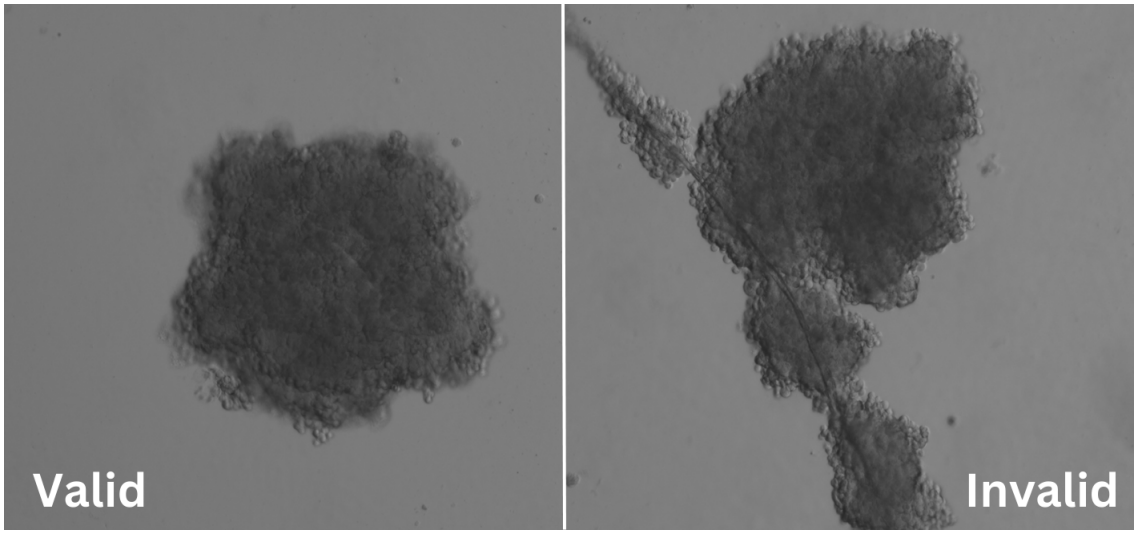

Figure 5.1: invalid vs valid

The original images are 2456x2054 pixels in size, in 16-bit grayscale, and consist of multiple layers. These layers are obtained by capturing images at different focal planes in brightfield microscopy. The number of layers can vary, as images can be taken at any number of focal planes. The sharpness of each layer of an image depends on how well the focal plane of the microscope aligns with the depth of the sample. Only one layer of this 3D tumor tissue model will be perfectly in focus, while others may appear blurry because they are slightly above or below the focal point as you can see in figure 5.2. Combining these focal planes later in computational analysis can provide richer data, even if some layers are blurry individually. This is one of the reasons why I decided to use multiple layers. The images I received from the lab mostly have three layers, while a few have one or five layers.

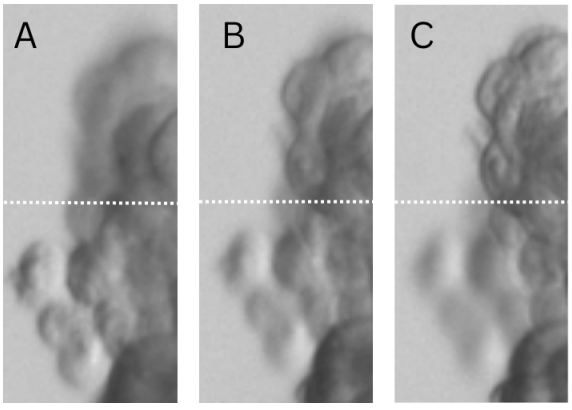

Figure 5.2: invalid vs valid

Each image in the dataset consists of different layers captured at varying focal lengths. For

example, if Image 1 has layers corresponding to focal lengths A, B, and C, Image 2 will also have layers corresponding to the same focal lengths A, B, and C, as all images within the same experiment are captured using consistent acquisition settings. However, for images from different experiments, the focal lengths may differ. For instance, images from Experiment 1 (single dose) may have slightly different acquisition settings compared to images from Experiment 2 (drug screening), leading to variations in the focal lengths and corresponding layers. Secondly, the layers in each image are slightly misaligned when stacked, as shown in the figure 5.3. This misalignment may or may not affect the performance of ranking.

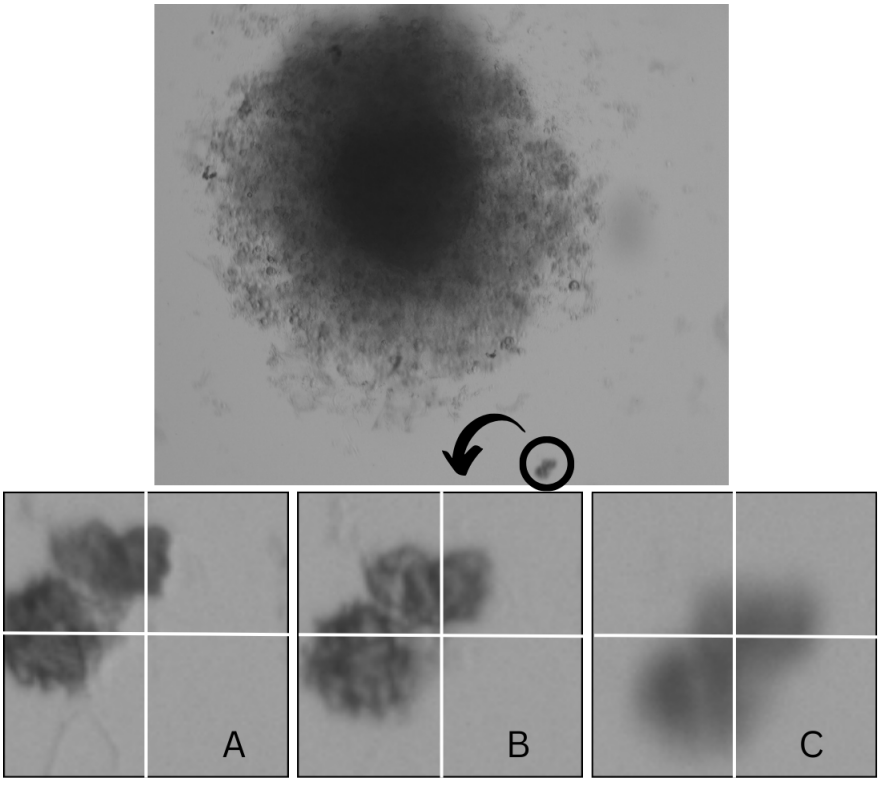

Figure 5.3: Misalighned

We have some time noise as in figure in the images that happens during the process of whole experiment. It could happen due to the mishandling of the environmental settings. talk about noise that we ignored as our limitation of work

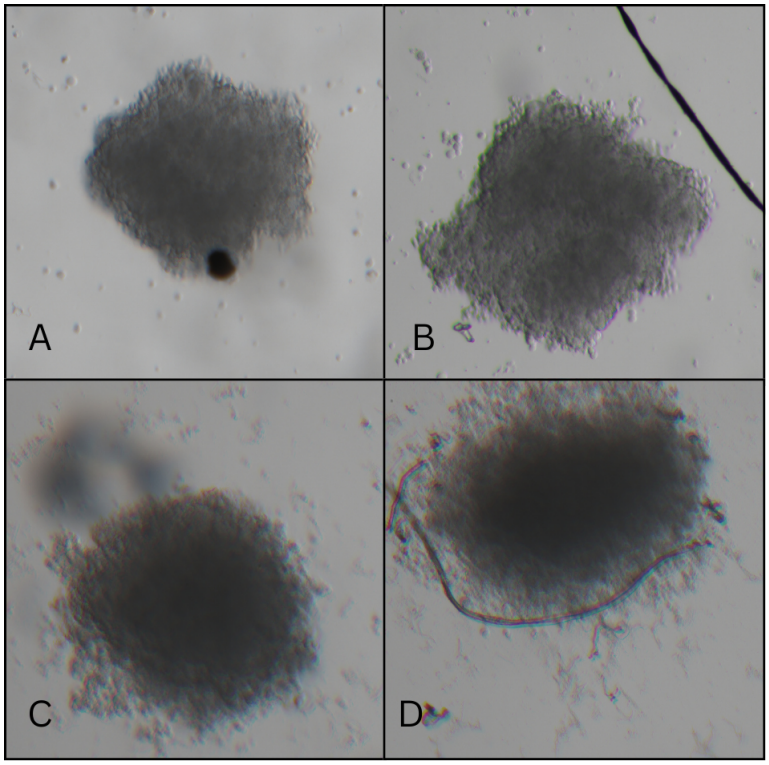

Figure 5.4: Noise

organised folders

Figure **??** illustrates that, even with the application of the same drug at the same concentration, the morphology of 3D tumor tissues changes differently.

show drug screen with same drug concentration we have diferent effect between different experiment. this maybe due to environmental factors or anything

rewrite intro lab to include all type explained in tabel below

The table below shows the division of different types of image datasets we have, as explained in the section 1.1.

Class	**Drug Screened**	**Single Dose**	**Cond 10**	**Untreated**	**Day 8 & 9**	**Total**
No. of Images (%)	200 (15.94%)	103 (8.21%)	201 (16%)	510 (40.67%)	240 (19.14%)	1254

Table 5.1: Dataset Class Overview

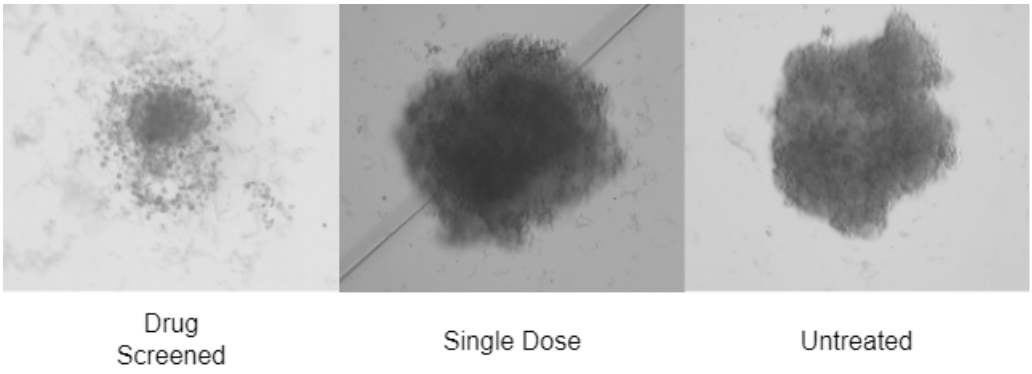

Figure 5.5: Three different types of images: Drug Screened, Single Dose, and Untreated as mentioned in section 1.1.

An 8-bit image encompasses 256 color tones (ranging from 0 to 255) per channel, whereas a 16-bit image accommodates 65,536 color tones (ranging from 0 to 65,535) per channel, in our case 65,536 shades of gray. Retaining the original 16-bit depth is crucial for two primary reasons:

1. Converting it to an 8-bit image for faster and more efficient computation can lead to significant information loss in intensity details. Since 8-bit images only allow 256 possible values, the finer variations in intensity that are present in 16-bit images become compressed as illusstrated in 5.6. For example, two distinct values in 16-bit (such as 30,000 to 30,048) could map to the same 8-bit value (for instance, both might be mapped to 117). This results in the loss of subtle intensity differences, which can be crucial in our image task, where minute variations in intensity can be indicative of important features such as gradual trasnsition of dark color frfom center to border or amount of debris surrounded to the tumor cell.

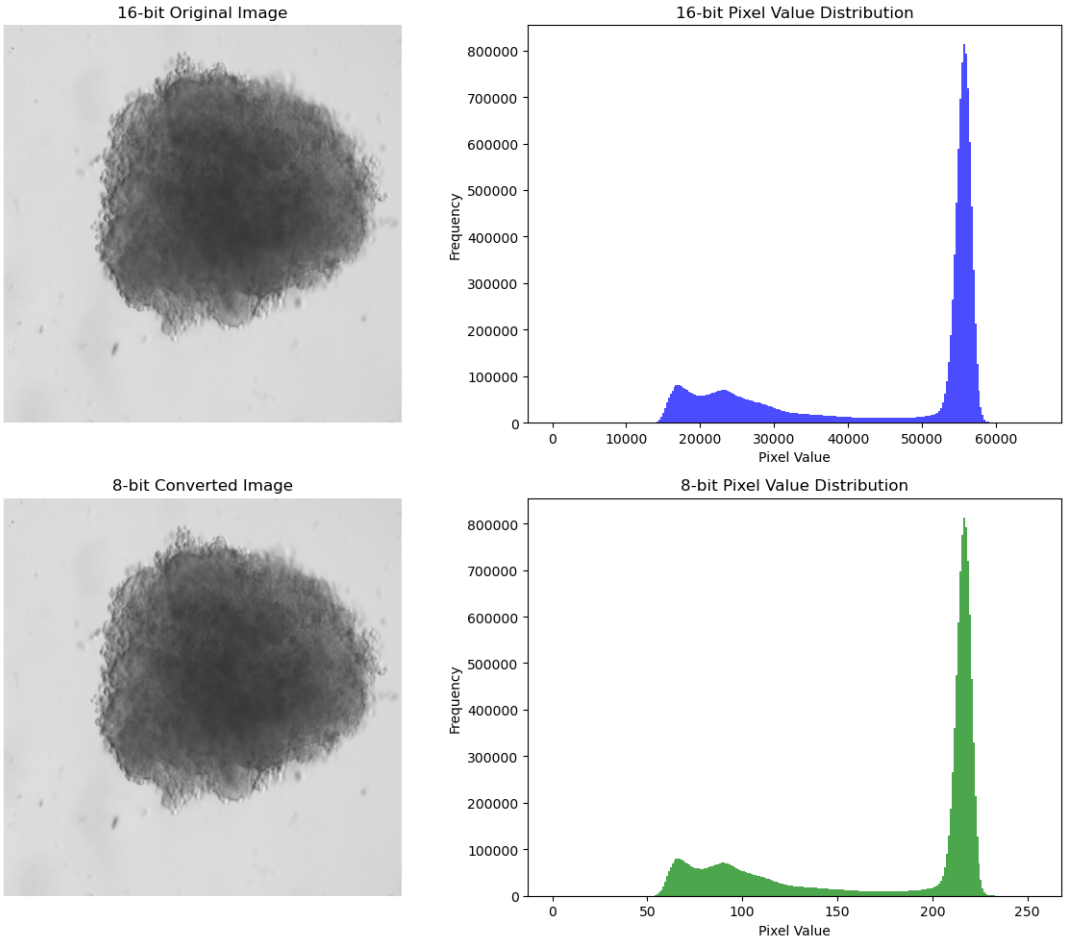

Figure 5.6: 8-bit vs 16 bit data lose comparison

2. During data augmentation processes that involve substantial alterations in brightness, contrast, or color, an 8-bit image—already limited to 256 tones—could lose up to 50 percentage of these tones, leaving only 128 levels of color and tone. This reduction can lead to ”banding,” where areas with smooth transitions in tone exhibit visible stripes with jagged edges. In contrast, a 16-bit image, even with a 50 percentage reduction in tones, would retain over 32,000 levels. This higher tonal range allows for smoother transitions, better edge preservation, and enhanced accuracy in color and hue representation. As a result, the dynamic range—the difference between the lightest and darkest areas of the image—remains much more effectively preserved in 16-bit images than in 8-bit images.

In our case, the maximum reduction in unique pixel values for the 8-bit images, regardless number of channels was found to be 99.27 percentage after 3000 epochs of random color jitter applied using `torch.transforms.RandomApply([transform.ColorJitter(brightness=1,`

`contrast=1, saturation=1, hue=0)], p=1)` as shown in figure 5.7 and 5.8, whereas for the 16-bit single-channel images (where one sharp layer was extracted from all three layers and considered as input for data augmentation), the reduction in unique pixel values was only 49 percentage after 3,000 epochs of random color jitter, as shown in Figure 5.9. Hence its interesting to experiment the entire methods using single channel as future work, also because single channel remove the problem of stacking inconsistant layers

8-bit three-channel image example before and after data augmentation:

- Number of unique pixel values in the original image: 137
- Number of unique pixel values in the augmented image: 3
- Original Image - Minimum pixel value: 33, Maximum pixel value: 170
- Augmented Image - Minimum pixel value: 0, Maximum pixel value: 2

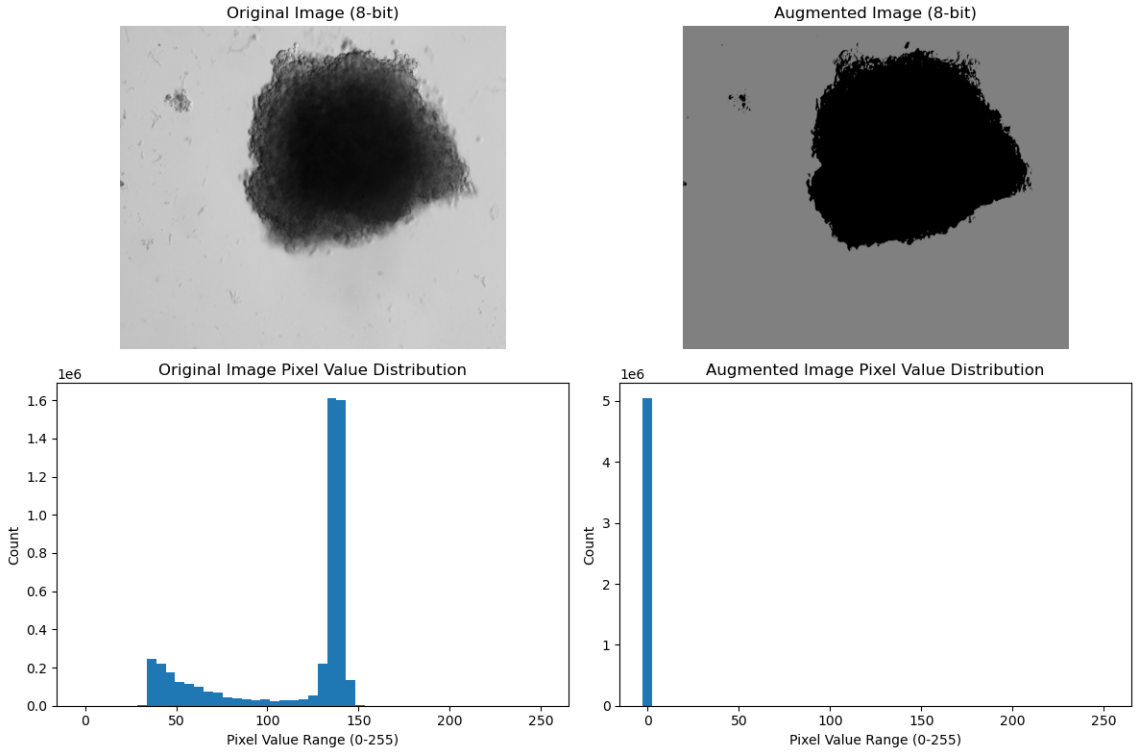

Figure 5.7: 8-bit three-channel image after 3000 epochs of random color jitter applied using PyTorch. Reduction percentage in unique pixel values: 97.81%

8-bit single-channel (sharp layer) image before and after data augmentation:

- Number of unique pixel values in the original image: 137
- Number of unique pixel values in the augmented image: 3
- Original Image - Minimum pixel value: 33, Maximum pixel value: 170
- Augmented Image - Minimum pixel value: 3, Maximum pixel value: 3

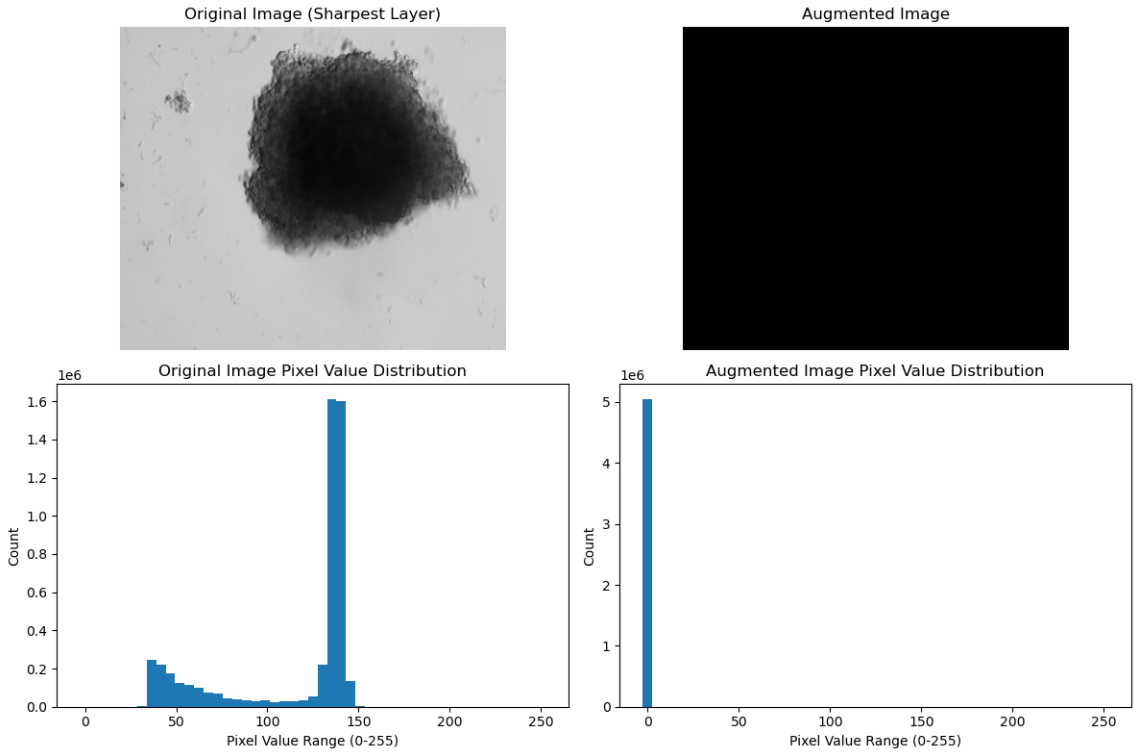

Figure 5.8: 8-bit single-channel image after 3000 epochs of random color jitter applied using PyTorch. Reduction percentage in unique pixel values: 99.27%

16-bit single-channel image before and after data augmentation:

- Number of unique pixel values in the original image: 2111
- Number of unique pixel values in the augmented image: 1058
- Original Image - Minimum pixel value: 0.13064774870872498, Maximum pixel value: 0.6666666865348816
- Augmented Image - Minimum pixel value: 0, Maximum pixel value: 1

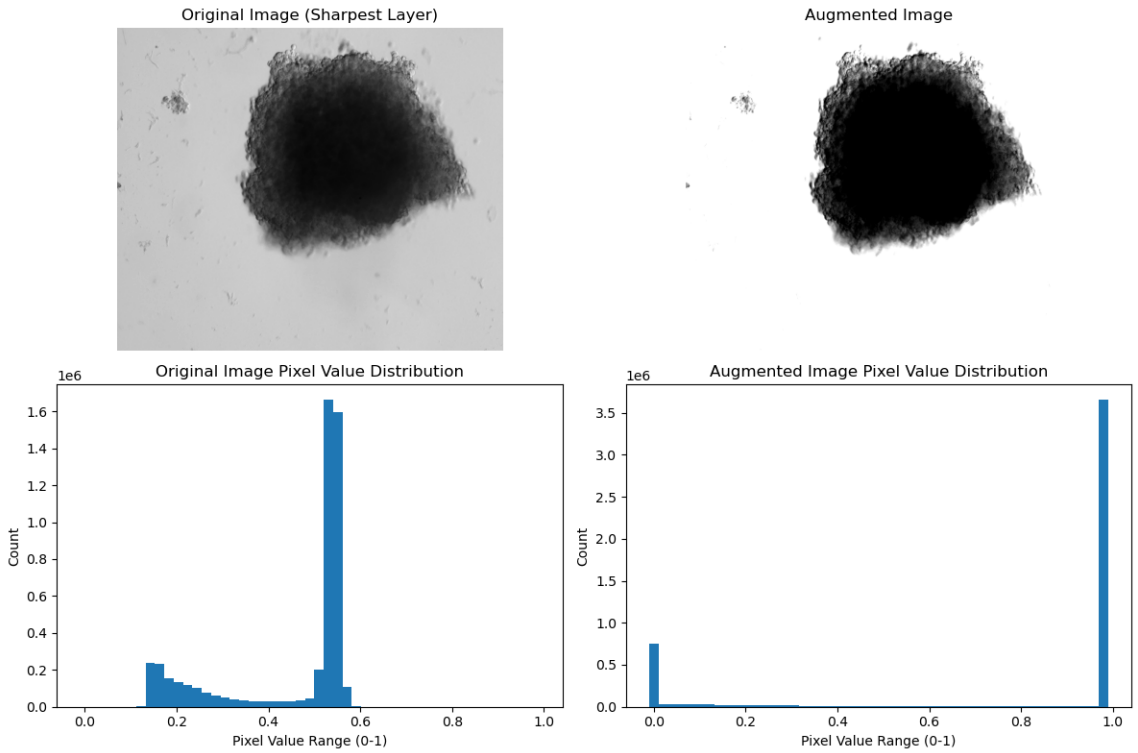

Figure 5.9: 8-bit single-channel image after 3000 epochs of random color jitter applied using PyTorch. Reduction percentage in unique pixel values: 49.88%

1. challenges: 1.1 limited data set for day 7 to day 10 prediction model 1.2 day 10 image can be flipped, blured, brightness change, position change (position cahnge is due to because when we take day 10 images we have to bring the well plate outside and sometimes when we put that into microscope the position is already changed). Relativly small position of tumor cell in the image from center to other directions change can be resolved for some extend using center crop approach, but if the cell is in the edge then center crop won't help. i didn't do any data preprocessing specifically for that problem. but I expect the croping and resizing to 96*96 will be a solution. because when we do that data transformation, augmented pairs will be positioned differently. only state if you can show the proof horizontal, vertical, rotation = 90 and 270, Horiflip+rotation 90, Horiflip+rotation270 augs also helps to make invariate to position changes.

1.3 same drug can have different effect on day 10 (example: huge variation in: ds 61 g6 and 41 gp6. less variation: RBTDS 4.1 and 4.2 gp 3) initially i thought I will be able to use as drug gp to evaluate, because of this difference, the difference can be due to different patient cancer cell? I choosed images from different gps which have huge debris amount visually. it was few

so I didn’t code my self.

6 Structure of the Thesis

The goal of this thesis is to leverage representation learning of bright-field microscopy images using SimCLR to develop a ranking/ordering scale (1 to n) for all images.

In this chapter, I will explain the framework designed to address this problem. The task is divided into three main pipelines.

First, we learn latent representations from the images using SimCLR as a self-supervised learning (SSL) model. The details will be explained in the *Methodology for SimCLR* chapter.

Next, we perform an intermediate evaluation to assess whether the features have been effectively learned. The intermediate evaluation aims to:

1. Classify the images,

2. Cluster the images, and

3. Ensure that the same features are extracted for identical images, even after transformations such as flipping, rotation, blurring, or brightness changes, using a direct distance measure approach.

The details of this step will be explained in the *Methodology for Intermediate Evaluation* chapter.

Subsequently, we use the learned features to establish a ranking scale. To achieve this, the following methods are employed:

1. Prediction model,

2. K-means centroid approach, and

3. Softmax approach.

The details of these methods will be discussed in the *Methodology for Ranking Strategy* chapter.

Finally, we address an important question: why do we need to learn feature vectors from the images? Could the ranking task perform better using the raw images directly instead of SimCLR features? To answer this, we conduct a comparative study to evaluate the performance of both approaches.

The details of this comparative study will be covered in the *Methodology for Comparative Study* chapter.

The overall structure of the thesis is illustrated in Figure 6.1.

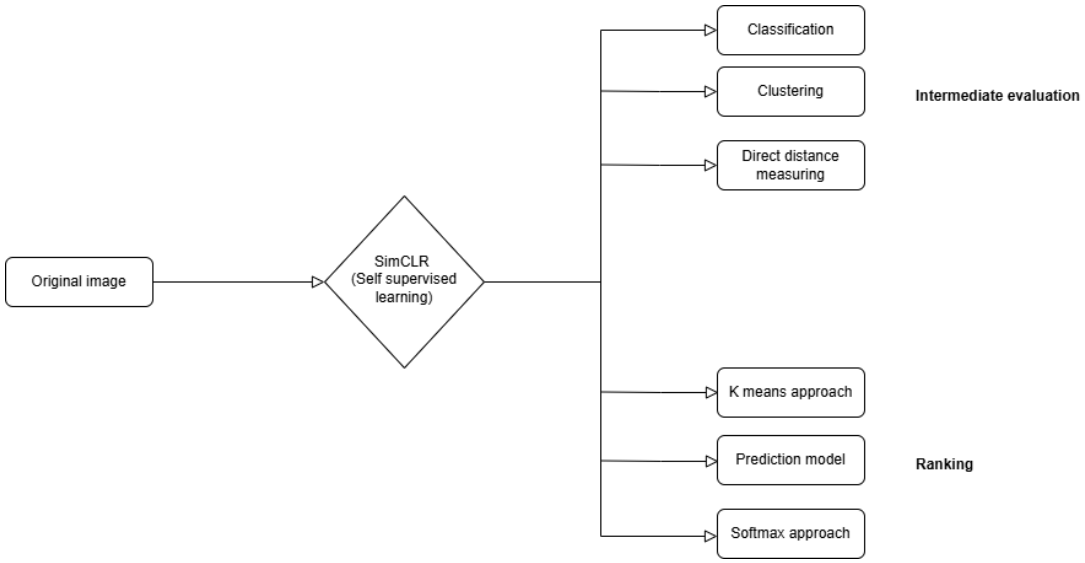

Figure 6.1: Overall framework of the thesis.

7 Methodology for SimCLR

7.1 Data preprocessing

However, for the ease of use to integrate with our pretarined architecture resnet18 as it trained on 3 channels, I determined to start with 3 layers per image. Hence from now on each layers of the tiff image are refered as channels. what matters is most sharped because most sharpest channels have more texture/edge information information than less sharpness less information (less texture/edge information) add the details of no of channels/image and how does it calcualtes the sharpness intuiton behind it: https://chatgpt.com/share/675f440d-3868-8010-ae82-ad6cdca13c5d last explanation

the images with one channel will duplicate the channels to make upto 3.

add croping image below?

1. We can't center crop it because for some of cancer cells are not centered in the image instead they are close to egdes. so we have to make sure that when we crop it it should include the cacne cell fully. the solution is find the boundary of cancer cell and get the bounding box of cancer cell then make crop to required size. original image width size is: 2456*2054 (H*W). crop the original image to have H = W. since the cancer cell debris spread across the width, we didn't change the width to include the debris, instead we reduced the hight to same size of width. hence we get H = W = 2054. if cancer cell is formed for instance, the cell it self spread acroos one diemsion that means its not valid cultvation by robot so we can digard it. ie maximum area of cancer cell should be included in this square 2054*2054 size image. advantage of croping like this are:

a) remove unnecessary background which contains no information

b) there will be no shear during resize to 96*96 augmentation like in the figure.

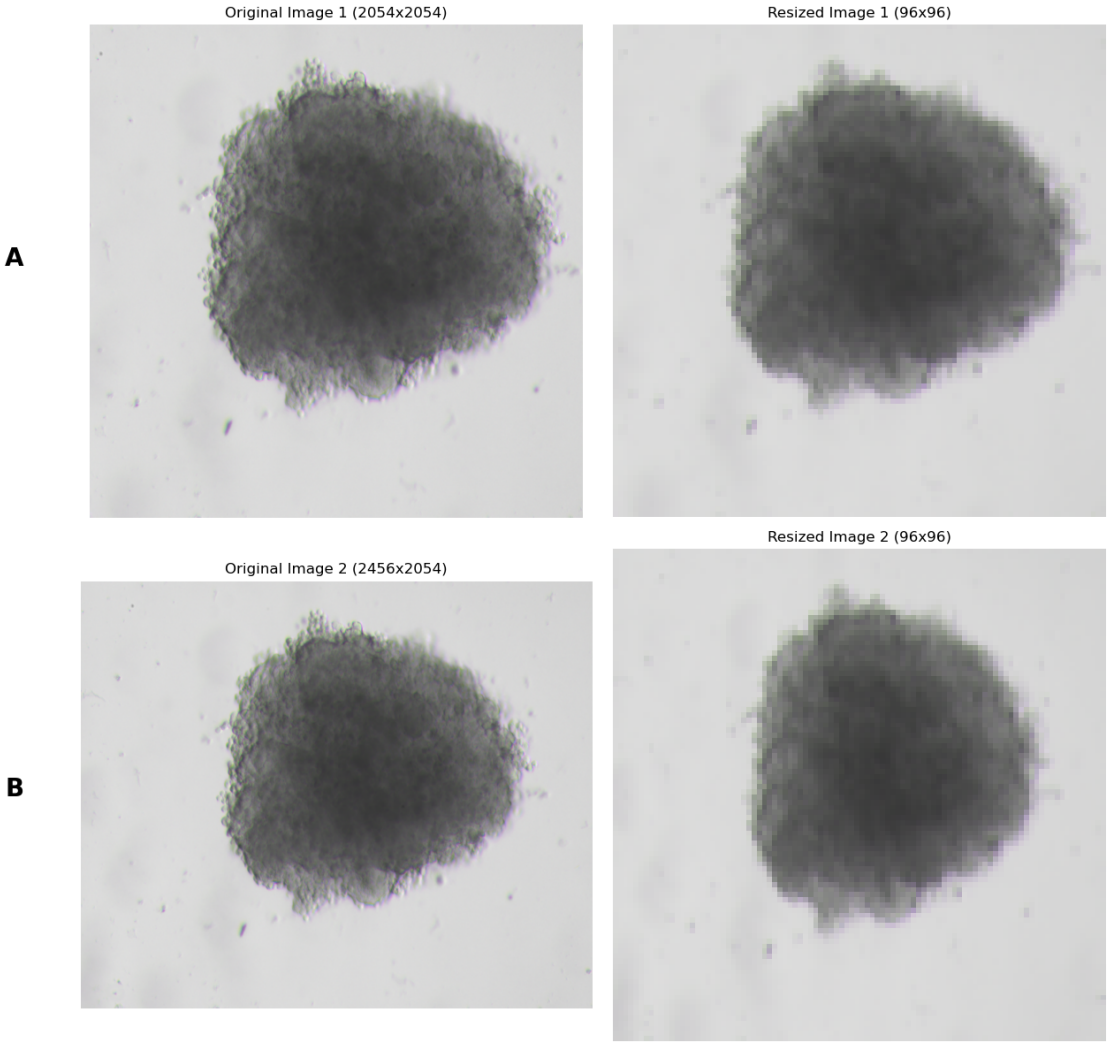

Figure 7.1: First row: croped to 2054*2054 then resized to 96*96: no shear change/elongation in one dimension happened, Second row: original image 2456*2054 then resized to 96*96: shear change/elongation in one dimension happened

c) rotation = 90 and 270, Horiflip+rotation 90, Horiflip+rotation270 this is only possible because of square image. if its other angles except multiples of 90 then images will have black part for that we need additional careful interpolation or something like that. so my point is since the image is square we could take rotations of 90 mulitiples without any additional tasks. explained in 7.2.

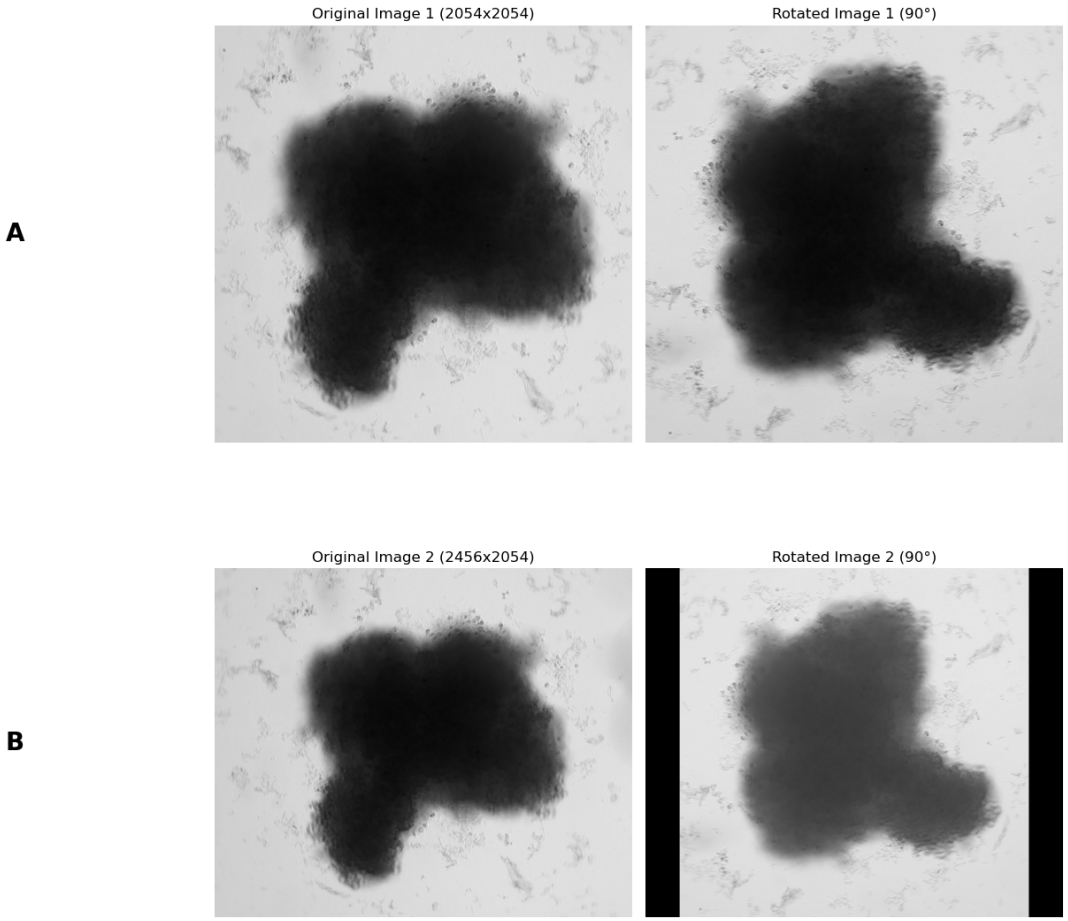

Figure 7.2: A: croped to 2054*2054 then resized to 96*96: No black cuts. B: original image 2456*2054 then resized to 96*96: black cuts happened

2. Normalize the 16-bit image to [0, 1] for the following reasons:

    a) Ideally, normalization should be done at the end after augmentations to ensure scaled input to the neural network, but in our case, we have to normalize first since the augmentation with `torch.transform.ColorJitter` didn't work without scaled data.

    b) `torch.transform.ToTensor()` didn't scale the data points to the [0, 1] range.

3. Perform data augmentations which detaily explained in the 7.2

4. Perform Z-score normalization after data augmentation for the following reasons:

    a) Pretrained models require this preprocessing.

b) It ensures that the data is still normalized even after data augmentation tweaks, allowing for effective feeding into the neural network.

5. For each original image, repeat step 2 twice to obtain two augmented images.

Visualisation of before and after preprocessing of image shown in figures 6.5 to 6.11.

7.2 Data augmentation

I started with best data augmentation pipeline that proposed by simclr [2] paper did, because it gave best performance on natural image dataset like Imagenet and CIFAR for downstream task like classification.

Which follows a sequential of:

1. Randomly crop and resize to specific smaller size . In my case I choosed to crop and resize to 96×96. i choosed this small H and W as image size to feed train our model because of two reasons: 1. computational fast, so that we can complete the pipeline in time. 2. if we can get good performance in ranking with this small image sizes (ie less pixel details compared to 2054 * 2056) that means we can improve the performance with larger image sizes. The crop of random size (uniform from 0.08 to 1.0 in area) of the original size and a random aspect ratio (default: of 3/4 to 4/3) of the original aspect ratio is made. This crop is finally resized to the original size.

2. Apply a horizontal flip, (simclr paper only used horizontal flip because it doesn't make sense to have vertical flip/rotation in natural images. This is default policy in SimCLR as it improve 1 percentage accuracy for downstream task in simclr paper) vertical flip, (Horizontal+Vertical flip), Rotation = (90, 270), Horizontal flip + Rotation = (90, 270) with the probability 50 percentage. We can have these aditional augs because it will still fall into our distribution samples. Other flip and rotation combinations are restrained due to its repeatable pattern to the above as shown in 7.3 and due to the black cut problem explained in 7.2. We set this as our default for all data augmentation.

3. Randomly change the brightness and contrast upto lower=1-0.8, upper=1+0.8 with the probability of 80 percentage. I removed saturation and hue since it doesn't have effect on gray scale images since gray scale images are nuetral and saturation and hue of a gray scale image is zero. ie when we try to adjust the value, we have essentially have no color to modify as it is achromatic and grayscale images contain only intensity information.

4. Gaussian blur and increase in sharpness. Blur augmentation is also default policy in SimCLR, because they find it helpful as it improves the performance for linear classification task by around 2 percentage. as they do we randomly sample sigma from 0.1 to 2 . We also kept the kernal size to be 10 percentage of image height/width, in our case 5. Some images are blured (Probabily due to the microscopic error). So inorder to mimic original version, I added sharpness increase. It increaases the sharpness by a factor of 2. We set this also as our default for all data augmentation. sharpeness increase what happens to the image? figure explaination? edge details increase?

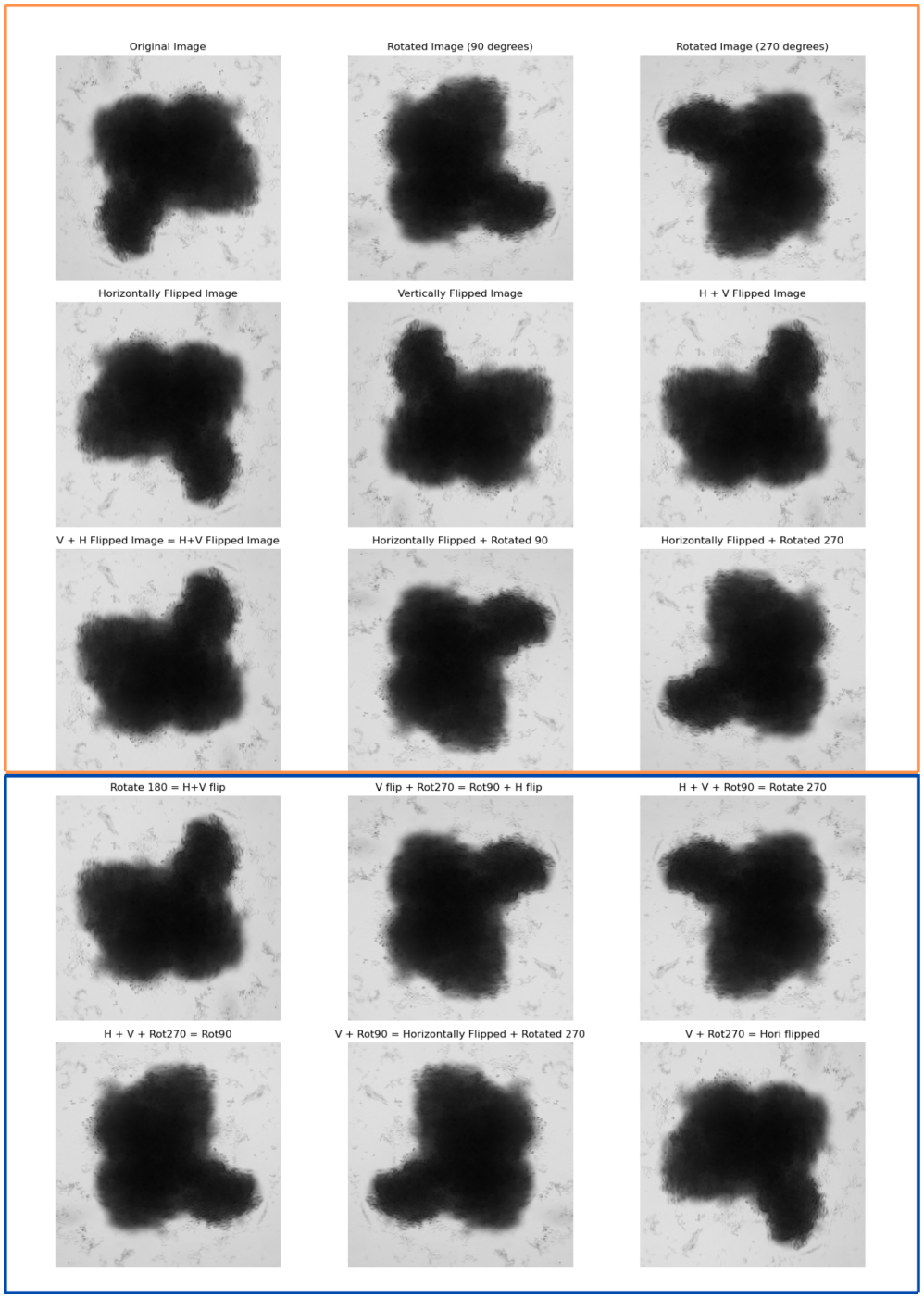

Figure 7.3: Orange box consist of implemented flips and rotations. Blue box consist of avoided flips and rotation combinations because of their repeated pattern to the orange box augmentations.

I call the above chain of data augmentation pipeline parameters as 'strong' data augmentation which illustrated in 7.6 first row.

Since there is a strong intensity change in color jitter in the above data augmentation I decided

to analyse the effect of only that color jitter. For 16-bit images with 3 channels, instead of a reduction, there was an increase in the number of unique pixel values—by a maximum of 258,757 percentage. The issue with this increase is that after data augmentation, the new pixel values are not distributed similarly to the original image. Instead, they shift to the two extremes,such as 0 or 1 which deviate significantly from the original image distribution, as shown in Figures 7.4.

16-bit three-channel image before and after data augmentation:

- Number of unique pixel values in the original image: 2111
- Number of unique pixel values in the augmented image: 5044624

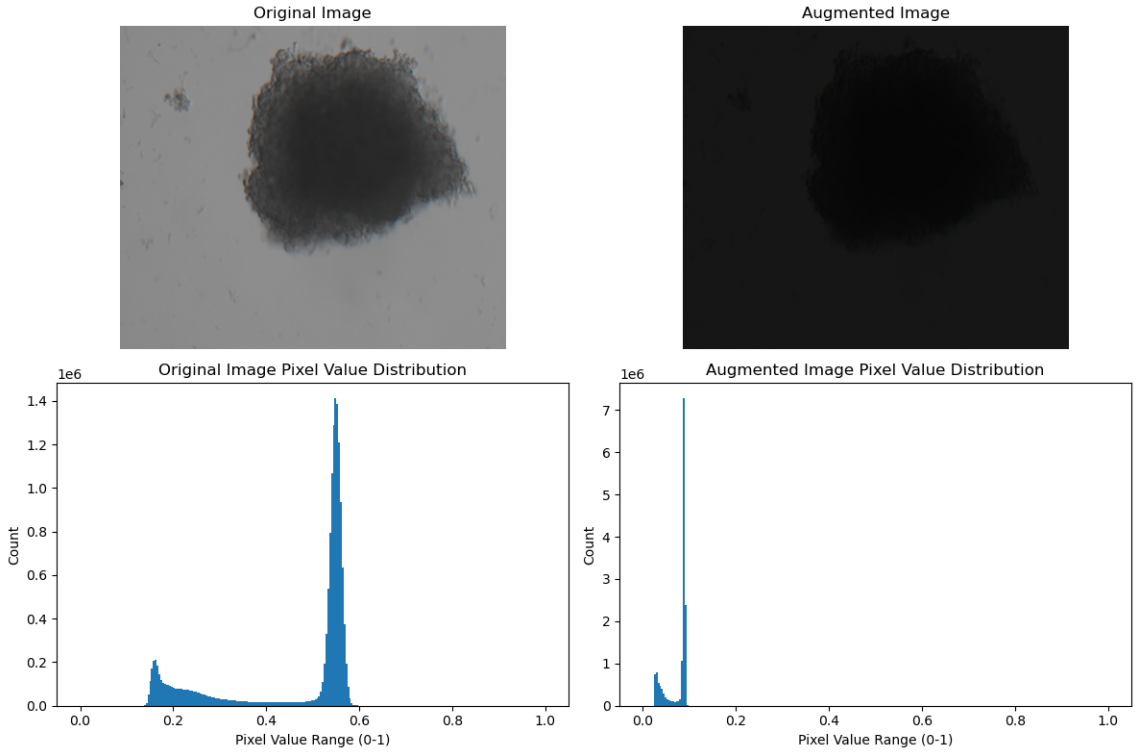

Figure 7.4: 16-bit three-channel image after 3000 epochs of random color jitter applied using PyTorch.

Comparing to three channel, single channel after 3000 epochs even with 'strong' augmentation only had 49% maximum reduction as observed in figure 7.5. It is also not ideal, as it remove the gradual spread of darker regions and debris around the cells. But compare to three channel it have less effect on getting out of original distribution. thats why we need to experiment the

downstream tasks with single channel.

Should we add the 3 layer position change issue? remove both if we are not showing comparison of sinlge and three

16-bit single-channel image before and after data augmentation:

- Number of unique pixel values in the original image: 2111
- Number of unique pixel values in the augmented image: 1058

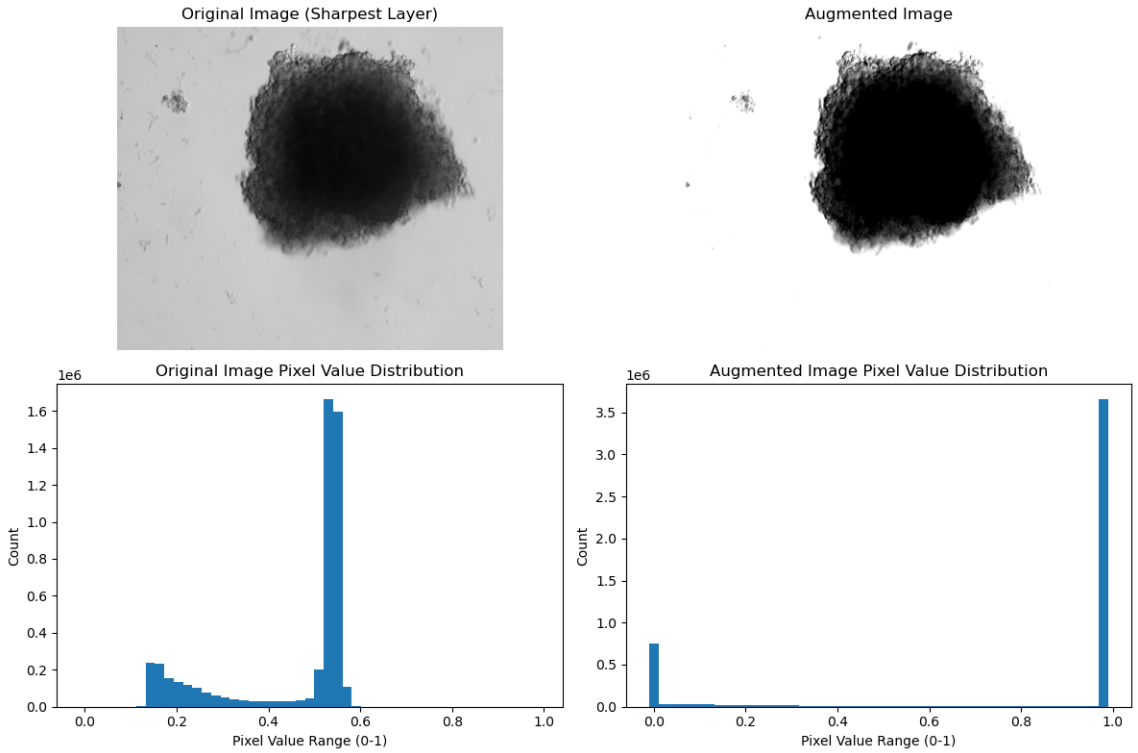

Figure 7.5: 16-bit single-channel image after 3000 epochs of random color jitter applied using PyTorch. Reduction percentage in unique pixel values: 49.88%

As we seen in the figures 7.4 and 7.5, I found that those strong color jitter augmentations transformed to distribution out of the original distribution. Hence we need to experiment with less intensity change instead of using SimCLR values.

Also original simclr random crop resizing to 96*96 is first done by random croping of random size (uniform from 0.08 to 1 in area). that means even really small crops without a cancer cell in it is train to have high cosine simililartiy to these same image smaller crop patch of dark cancer

cell from center due to the design of loss function in SimCLR. Which doesn't make sense. Hence the next data augmentation sequence designed to have random crop with scale 0.4 to 1 (so that assuming crop patches will have atleast small portion of cancer cell) and brightness and contrast variations changed to lower=1-0.2, upper=1+0.2 and lower=1-0.35, upper=1+0.35 consequently. I named this augmentation as 'sweet'. which illustrated in 7.6 second row.

In some drug screening group of images, when we look at the day 10 image, they are exploded (assuming due to the overdose). exploded in the sense, broke round small cancer cell surronding lot of debris around it. so if we apply the above random crop and resize aug: smaller crop from the round cancer cell part (without debris) have high cosine similarity with some other small crop from the round cancer cell part (without debris) of the same image. which doesn't connect anything about the debris surrounded by for exploed images. In general model will also learn to map high cosine similarity between dark area to gray area which can happen to generally all images except single dose images, meaning it doesn't learn any specific feature about certain patterns. Maybe its better to not to crop and just resize to 96*96 all the time so that model learn to map high cosine similarity by seeing the whole picture (Big picture). I understand this is not good argument because then dog and cat have same color and maybe similar shape but still learns to differentiate using strong crop in simclr. But maybe for time predictoin ranking model it helps. because model have to predict the change from day 7 to day 10. and most of the time the size/shape changes. especially to the category which are exploded, they produce debris. so maybe data augmentation seeing the whole picture is good to learn the model better for time change. With this assumption in mind, I decided to do data aug pipeline with just resizing in the begning.

Include figure to explain better

I named this augmentation as 'Resize', which illustrated in 7.6 third row.

All of the above augmentations includes contrast change. With this augmentation it is possible that model learned to have both dark cancer cell and gray cancer cell of same image have same feature map (because 2 dta aug from original image can be one with increased contrast which makes darker and one with less contrast which makes it gray) which is infact not good for our downstream task. because one of our goal is to differentiate/have diferent feature representation

for untreated images (gray color) and single dose (darker color).

Include figure to explain better

Hence I removed contrast from 'Resize' and 'Sweet' and named them as 'Resize ohne contrast' and 'Sweet ohne contrast' correspondingly. which illustrated in 7.6 third row and fourth row correspondingly.

Eventhough we know that these kind of strong augmentations is sensitive to our dataset, I was speculate that strong augmentation may have better learned latent space representations because I feel like the purpose of ths strong data augmentation explained by the authorsof simclr is not to include/increase the possible distribution that can happen in real life scenarios, but it is to learn the features that are invariant to these augmentations. ie if contrast change is there it learns to have high cosine sim for same shape. if croping happens it learns to have high cosine sim for same pixel color. if shape is different but color is same it learns to have high cosine sim. if shape is same but color is different it learns to have high cosine sim. if color is different it learns to have high cosine sim for shape something like this

as in figure when we did strong data aug background went white so it learns about only the dark matter.

Hence Its worth to experiment with this strong augmentation for downstream tasks.

add data augs without brightness too if we have time, and add our improvements to strong like sharpness rot flips

5 Different data augmentation pipe lines explained above illlustrated as nutshell in 7.6

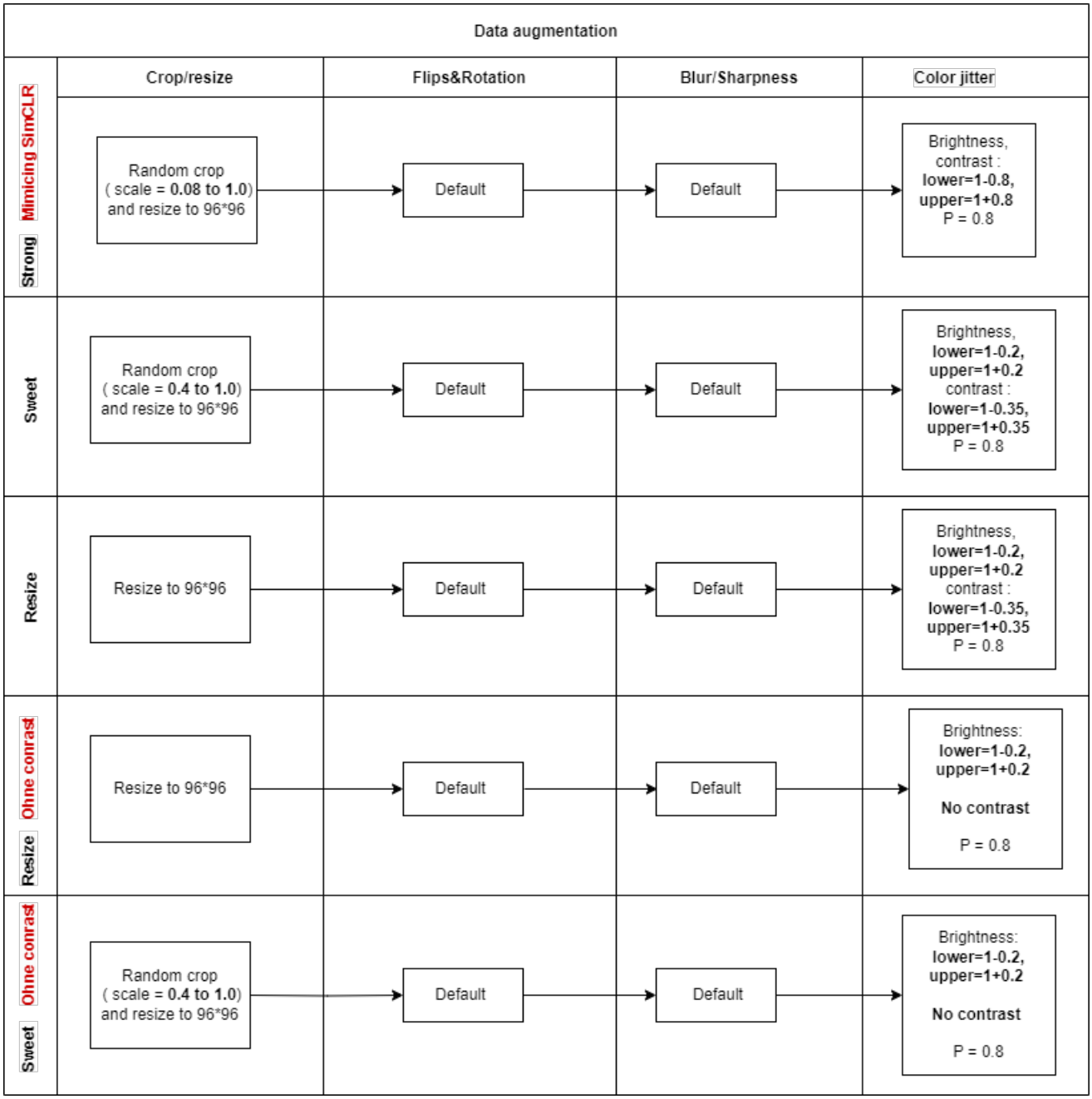

Figure 7.6: 5 Data augmentations we explored

7.3 Train SimCLR as SSL model

add explanation that tere is no pushing in the loss function

For step 1, as explained in chapter 7, SimCLR was used as the first model for self-supervised learning (SSL). Future work: Later, other models such as masked autoencoders and DINO

will be explored, depending on the available time. Why we would like to try other models? Because SimCLR demands larger batch size and more data for better performance which we don't have.

Model

The Resnet18 [6] model processes a single image to produce a latent representation of the input, aiming to cluster similar images together in a latent space.

Training

The training process follows these steps:

1. We take a batch of images with batch size N.
2. Our dataset class returns two augmented versions for each original image as explained in section 7.1 in the batch, resulting in $2N$ images as input.
3. The model produces $2N$ latent representations, independently for each augmented image.
4. For each batch, the two augmentations of the same image are treated as positive pairs, while all others are considered negative pairs.
5. We calculate the cosine similarities between the positive and negative pairs. These cosine similarities are then used as input to the loss function described below equation 7.2

The original loss function for each pair from SimCLR paper [2] is defined as:

$$\ell_{i,j} = -\log \frac{\exp\left(\operatorname{sim}\left(z_i, z_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbf{1}_{[k\neq i]} \exp\left(\operatorname{sim}\left(z_i, z_k\right)/\tau\right)} \tag{7.1}$$

which we can reformualte as:

1. Apply the logarithm: The negative log of a fraction can be separated into the difference of the logarithms:

$$\ell_{i,j} = -\left(\log\left(\exp\left(\text{sim}\left(z_i, z_j\right)/\tau\right)\right) - \log\left(\sum_{k=1}^{2N} \mathbf{1}_{[k\neq i]} \exp\left(\text{sim}\left(z_i, z_k\right)/\tau\right)\right)\right)$$

2. Simplifying the first term: The logarithm of an exponential function simplifies as follows:

$$-\log\left(\exp\left(\text{sim}\left(z_i, z_j\right)/\tau\right)\right) = -\frac{\text{sim}\left(z_i, z_j\right)}{\tau}$$

Substituting that back into the equation:

$$\ell_{i,j} = -\frac{\text{sim}\left(z_i, z_j\right)}{\tau} - \log\left(\sum_{k=1}^{2N} \mathbf{1}_{[k\neq i]} \exp\left(\text{sim}\left(z_i, z_k\right)/\tau\right)\right)$$

This gives us:

$$\ell_{i,j} = -\frac{\text{sim}\left(z_i, z_j\right)}{\tau} + \log\left[\sum_{k=1}^{2N} \mathbf{1}_{[k\neq i]} \exp\left(\text{sim}\left(z_i, z_k\right)/\tau\right)\right] \tag{7.2}$$

where $\mathbf{1}_{[k\neq i]} \in \{0,1\}$ is an indicator function evaluating to 1 iff $k \neq i$, and τ denotes a temperature parameter. The final loss is computed across all positive pairs, both (i, j) and (j, i), in a mini-batch with z_i, z_k representing negative pairs.

Equation 7.2 implemented as the loss function in our experiments.

The above standard SimCLR loss function and data augmentation combination with the ResNet18 model will be used as the initial benchmark for experiments, and in the future, the following variations will be explored.

Variation ideas:

1. Each image is treated as an RGB image with 3 channels, and two of the best-performing data augmentations, which yielded high performance for our downstream task.

2. One channel is considered as the anchor (the most sharpened layer), and two of the best-performing data augmentations, which yielded high performance for the our downstream task.

3. Since SimCLR architecture [2] allows for flexibility in model selection, and explore other pretrained models than Resnet18 suitable for medical grayscale images. For example pretrained U-Net [10] model for MRI brain images from PyTorch.

For variations 5 and 7 we need to modify the loss function since it includes more than 2 augmentations.

Variations implementations:

The two variations tried so far differ only in how they handle the image for data augmentation.

In the first variation, we take a 3-channel image and treat it like a standard RGB image, applying SimCLR-style augmentations to create two augmented versions.

In the second variation, we take a 3-channel image and compute the sharpness of each layer by calculating the magnitude of the gradient of pixel intensities in the x and y directions, which indicates edge strength and provides a measure of how sharp the transitions between pixel values are. The sharpest layer is used as the anchor, while the other two layers are treated as augmentations.

Variation 1:

Input to model (train loader dimension) :

- aug1: torch.Size([16, 3, 96, 96]) (batch size, no of channels, H, W)

- aug2: torch.Size([16, 3, 96, 96]) (batch size, no of channels, H, W)

Model output just after convolution layers: (before applying projetion head)

- torch.Size([16, 512, 1, 1]) (Batch size, standard resnet18 output dimension after avg pooling, H,W)
- This output feature will be used for further downstream task.

Model output after projection head:

- torch.Size([16, 20]) (Batch size, no of values in feature vector)
- No of values in feature vector is a variable which we can change and experiment which will give better accuracy.

The figure below shows the anchor and its two augmented versions as explained in section 7.1.

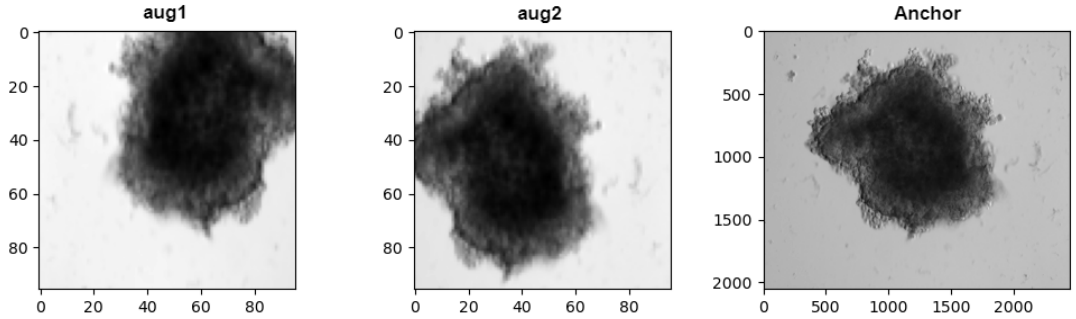

Figure 7.7: Sample 1: Anchor (the preprocessed original image) with 3 channels and its augmentations

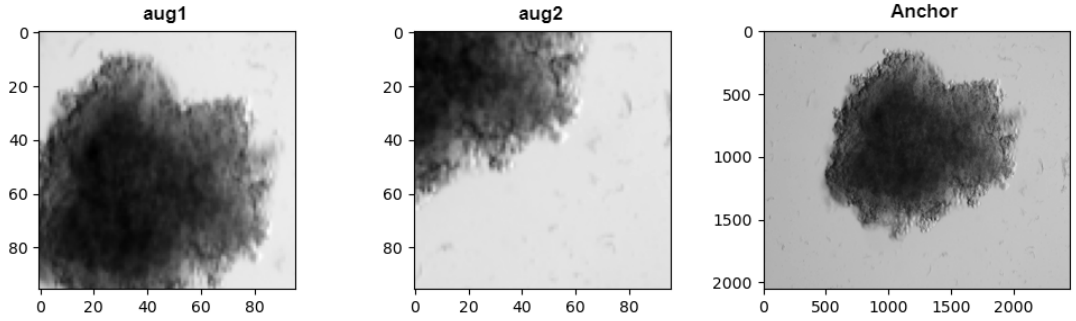

Figure 7.8: Sample 2: Anchor (the preprocessed original image) with 3 channels and its augmentations

Variation 2:

Input to model (train loader dimensions) :

- aug1: torch.Size([16, 1, 96, 96]) (batch size, no of channels, H, W)
- aug2: torch.Size([16, 1, 96, 96]) (batch size, no of channels, H, W)

Model output just after convolution layers: (before applying projetion head)

- torch.Size([16, 512, 1, 1]) (Batch size, standard resnet18 output dimension after avg pooling, H, W)
- This output feature will be used for further downstream task.

Model output after projection head:

- torch.Size([16, 20]) (Batch size, no of values in feature vector)
- No of values in feature vector is a variable which we can change and experiment which will give better accuracy.

The figure below shows the anchor and its two augmented versions as explained in section 7.1.

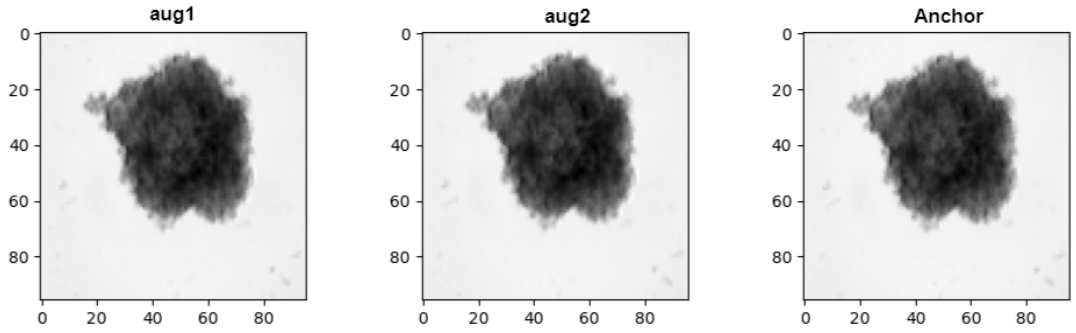

Figure 7.9: Sample 1: Anchor (the preprocessed sharpest layer amoung all 3 layers) with one channel and its augmentations

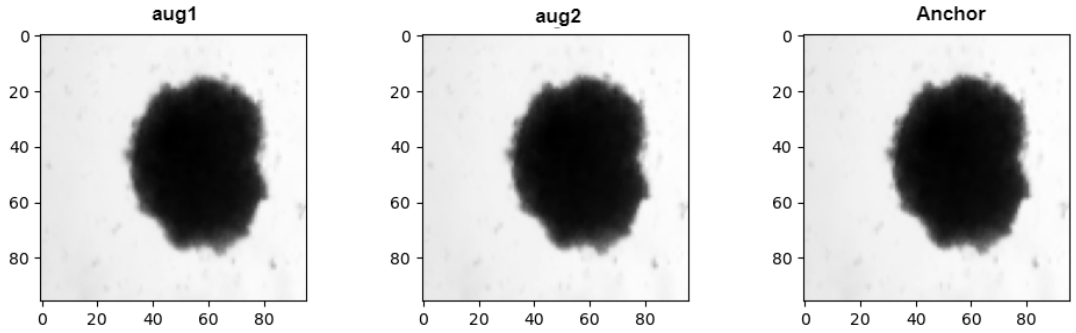

Figure 7.10: Sample 2: Anchor (the preprocessed sharpest layer amoung all 3 layers) with one channel and its augmentations

7.4 Evaluation result of SImclr

add figures of SImclr train and val and inference loss

8 Methodology for Intermediate evaluation of SimCLR model

change the name to evaluatoin to check whether simclr learned something? instead of intermediate evaluatoin?

If we do kmeans with cosine distance distance then it only compare direction of the vector thats why we need to do normal kmeans with euclidean dist to show the magnitude similarity

Final evaluation of the SimCLR model depends on the Ranking task, neverthless we can use other evaluation metrics, such as downstream task like classification, clusetring, distance measurement approach to check if simclr atleast learned to differentiate between untreated , single dose and exploded images.

8.1 Classification using Logistic Regression on the SimCLR features

1. A common approach to verify whether the SSL model has learned generalized representations is to perform Logistic Regression on the learned features. In other words, we use a single, linear layer that maps these representations to class predictions, where the three categories, 'untreated' and 'single dose', 'explod' serve as our classes. The Logistic Regression model can only perform well if the learned representations capture all the relevant features necessary for the task. Moreover, we don't need to worry much about overfitting since only a few parameters are trained. Therefore, we expect the model to perform well even with limited data.

2. Baseline comparison to original images. We classify the original images to see if simclr feature vectors are better or worse than original image to classify.

3 classes trained for 250 epochs. 1.untreated dataset: total no of images: 472 2. single dose dataset : total no of images: 103 3. exploded dataset from drug screening experiments: total no of images: 40

i didn't add any other comibnation of drug screening experiments since we are not sure whether they belong to which gp. (some of them have medium resemblence of the above dataset images but we can't be sure.)

8.1.1 Comparison of Classification Accuracy and Epochs for Different Data Augmentations

Augmentation Type	Metric	Strong	Sweet	Resize	Resize No Contrast	Sweet No Contrast
After Projection Head	Train Accuracy (%)	88	59.76	97.15	58.74	56.30
	Train Epoch	249	248	236	239	249
	Test Accuracy (%)	86.18	62.60	96.75	53.66	51.22
	Test Epoch	240	235	239	248	237
Before Projection Head	Train Accuracy (%)	100	100	100	100	100
	Train Epoch	148	18	62	14	44
	Test Accuracy (%)	100	100	100	100	100
	Test Epoch	1	1	2	2	2

Table 8.1: Performance metrics for different augmentation strategies before and after the projection head.

Table 8.2: Original Image Results

Metric	Train Accuracy (%)	Train Epoch	Test Accuracy (%)	Test Epoch
Original Image	95.12	18	94.31	15

I used some package to automate the width of the table in main.tex: change if needed

8.2 kmeans clustering

Idea is whether the learned representation from simlcr outperforms the original images in clustering the images (unsupervised manner) For that we use simple kmeans clusetring. We will

cluster them based on both euclidean distance as well as cosine distance.

Derivation of K-Means Clustering using Euclidean Distance and Mean

Objective Function The k-means algorithm aims to minimize the total squared Euclidean distance between data points and their assigned cluster centroids. The objective function is:

$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|\mathbf{x}_i - \mu_k\|^2$$

where:

- n: Number of data points,
- K: Number of clusters,
- $\mathbf{x}_i$: The i-th data point,
- μ_k: The centroid of the k-th cluster,
- r_{ik}: Binary indicator; $r_{ik} = 1$ if $\mathbf{x}_i$ belongs to cluster k, otherwise $r_{ik} = 0$.

Cluster Assignment Step

For a fixed set of centroids $\{\mu_k\}_{k=1}^{K}$, assign each data point $\mathbf{x}_i$ to the nearest centroid. This minimizes:

$$r_{ik} = \begin{cases} 1, & \text{if } k = \arg\min_j \|\mathbf{x}_i - \mu_j\|^2, \\ 0, & \text{otherwise.} \end{cases}$$

Centroid Update Step

For a fixed cluster assignment $\{r_{ik}\}$, minimize J with respect to the centroids $\{\mu_k\}$:

$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|\mathbf{x}_i - \mu_k\|^2$$

Focus on a single cluster k. The term involving μ_k is:

$$\sum_{i=1}^{n} r_{ik} \|\mathbf{x}_i - \mu_k\|^2 = \sum_{i=1}^{n} r_{ik} \left(\mathbf{x}_i^\top \mathbf{x}_i - 2\mathbf{x}_i^\top \mu_k + \mu_k^\top \mu_k \right)$$

Take the derivative with respect to μ_k and set it to zero:

$$\frac{\partial}{\partial \mu_k} \sum_{i=1}^{n} r_{ik} \|\mathbf{x}_i - \mu_k\|^2 = -2 \sum_{i=1}^{n} r_{ik} \mathbf{x}_i + 2 \sum_{i=1}^{n} r_{ik} \mu_k = 0$$

Simplify:

$$\sum_{i=1}^{n} r_{ik} \mathbf{x}_i = \sum_{i=1}^{n} r_{ik} \mu_k$$

Factor out μ_k:

$$\mu_k = \frac{\sum_{i=1}^{n} r_{ik} \mathbf{x}_i}{\sum_{i=1}^{n} r_{ik}}$$

This is the mean of the points in cluster k.

Algorithm Summary

The k-means algorithm alternates between the following two steps until convergence:

1. **Cluster Assignment Step**: Assign each point $\mathbf{x}_i$ to the nearest cluster:

$$r_{ik} = \begin{cases} 1, & \text{if } k = \arg\min_j \|\mathbf{x}_i - \mu_j\|^2, \\ 0, & \text{otherwise.} \end{cases}$$

2. **Centroid Update Step**: Update the centroid of each cluster as the mean of its assigned points:

$$\mu_k = \frac{\sum_{i=1}^{n} r_{ik}\mathbf{x}_i}{\sum_{i=1}^{n} r_{ik}}$$

This iterative process continues until the assignments r_{ik} and centroids μ_k no longer change or the change is below a threshold.

Cosine distance

Normalization

To calculate the cosine distance, we first **normalize** all data points and centroids. Suppose the normalized data points and centroids are $\mathbf{x}_i$ and $\mathbf{c}_k$, respectively, then:

$$\|\mathbf{x}_i\| = 1 \quad \text{and} \quad \|\mathbf{c}_k\| = 1.$$

This ensures all vectors are on the unit sphere. The cosine similarity between two vectors $\mathbf{x}_i$ and $\mathbf{c}_k$ is defined as:

$$\text{cosine similarity} = \frac{\mathbf{x}_i^\top \mathbf{c}_k}{\|\mathbf{x}_i\| \|\mathbf{c}_k\|}$$

Since $\|\mathbf{x}_i\| = 1$ and $\|\mathbf{c}_k\| = 1$, we substitute these values into the equation:

$$\text{cosine similarity} = \frac{\mathbf{x}_i^\top \mathbf{c}_k}{1 \times 1}$$

This simplifies to:

$$\text{cosine similarity} = \mathbf{x}_i^\top \mathbf{c}_k.$$

Thus, the **cosine distance** becomes:

$$\text{cosine distance} = 1 - \mathbf{x}_i^\top \mathbf{c}_k.$$

Relating Cosine Distance to Euclidean Distance

For normalized vectors, we derive the relationship between **Euclidean distance** and **cosine distance**. The squared Euclidean distance between a data point $\mathbf{x}_i$ and a centroid $\mathbf{c}_k$ is:

$$\|\mathbf{x}_i - \mathbf{c}_k\|^2 = \sum_j (x_{ij} - c_{kj})^2.$$

Expanding this:

$$\|\mathbf{x}_i - \mathbf{c}_k\|^2 = \|\mathbf{x}_i\|^2 + \|\mathbf{c}_k\|^2 - 2\mathbf{x}_i^\top \mathbf{c}_k.$$

Since $\|\mathbf{x}_i\| = 1$ and $\|\mathbf{c}_k\| = 1$, we get:

$$\|\mathbf{x}_i - \mathbf{c}_k\|^2 = 1 + 1 - 2\mathbf{x}_i^\top \mathbf{c}_k.$$

Simplify:

$$\|\mathbf{x}_i - \mathbf{c}_k\|^2 = 2(1 - \mathbf{x}_i^\top \mathbf{c}_k).$$

Thus, for normalized vectors, the Euclidean distance is proportional to the cosine distance:

$$\|\mathbf{x}_i - \mathbf{c}_k\|^2 = 2 \cdot \text{cosine distance}.$$

Rearranging to express the cosine distance:

$$\text{cosine distance} = \frac{\|\mathbf{x}_i - \mathbf{c}_k\|^2}{2}.$$

Objective Function

The k-means algorithm with cosine distance aims to minimize the cosine distance between data points $\mathbf{x}_i$ and their assigned cluster centroids $\mathbf{c}_k$. The objective function is:

$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \left(1 - \frac{\mathbf{x}_i^\top \mathbf{c}_k}{\|\mathbf{x}_i\| \|\mathbf{c}_k\|}\right),$$

where:

- n: Number of data points,
- K: Number of clusters,
- $\mathbf{x}_i$: i-th data point,
- $\mathbf{c}_k$: Centroid of cluster k (normalized to unit length),
- r_{ik}: Binary indicator; $r_{ik} = 1$ if $\mathbf{x}_i$ belongs to cluster k, otherwise $r_{ik} = 0$.

Objective Function in Terms of Euclidean Distance

Using the above result, the k-means objective function with cosine distance:

$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \left(1 - \mathbf{x}_i^\top \mathbf{c}_k\right)$$

can be rewritten in terms of Euclidean distance:

$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \frac{\|\mathbf{x}_i - \mathbf{c}_k\|^2}{2}.$$

Here, the factor $\frac{1}{2}$ accounts for the scaling difference.

Cluster Assignment Step

For a fixed set of centroids $\{\mathbf{c}_k\}_{k=1}^{K}$, assign each data point $\mathbf{x}_i$ to the nearest cluster based on the **cosine similarity** (or equivalently, minimize cosine distance):

$$r_{ik} = \begin{cases} 1, & \text{if } k = \arg\max_j \mathbf{x}_i^\top \mathbf{c}_j, \\ 0, & \text{otherwise.} \end{cases}$$

Centroid Update Step for Cosine Distance

For a fixed cluster assignment $\{r_{ik}\}$, minimize J with respect to the centroids $\{\mathbf{c}_k\}$:

$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \frac{\|\mathbf{x}_i - \mathbf{c}_k\|^2}{2}$$

Focus on a single cluster k. The term involving $\mathbf{c}_k$ is:

$$\sum_{i=1}^{n} r_{ik} \frac{\|\mathbf{x}_i - \mathbf{c}_k\|^2}{2} = \sum_{i=1}^{n} r_{ik} \frac{1}{2} \left(\|\mathbf{x}_i\|^2 + \|\mathbf{c}_k\|^2 - 2\mathbf{x}_i^\top \mathbf{c}_k \right)$$

Since the vectors are normalized, $\|\mathbf{x}_i\| = 1$ and $\|\mathbf{c}_k\| = 1$, we have:

$$\sum_{i=1}^{n} r_{ik} \frac{1}{2} \left(1 + 1 - 2\mathbf{x}_i^\top \mathbf{c}_k \right) = \sum_{i=1}^{n} r_{ik} \left(1 - \mathbf{x}_i^\top \mathbf{c}_k \right)$$

Now, take the derivative of the above with respect to $\mathbf{c}_k$ and set it to zero:

$$\frac{\partial}{\partial \mathbf{c}_k} \sum_{i=1}^{n} r_{ik} \left(1 - \mathbf{x}_i^\top \mathbf{c}_k\right) = \sum_{i=1}^{n} r_{ik} \mathbf{x}_i = \sum_{i=1}^{n} r_{ik} \mathbf{c}_k$$

Simplify:

$$\sum_{i=1}^{n} r_{ik} \mathbf{x}_i = \sum_{i=1}^{n} r_{ik} \mathbf{c}_k$$

Factor out $\mathbf{c}_k$:

$$\mathbf{c}_k = \frac{\sum_{i=1}^{n} r_{ik} \mathbf{x}_i}{\sum_{i=1}^{n} r_{ik}}$$

This is the mean of the normalized points in cluster k.

Centroid Update Step

For a fixed cluster assignment $\{r_{ik}\}$, update the centroids $\{\mathbf{c}_k\}$ as the **normalized mean** of all data points assigned to cluster k:

$$\mathbf{c}_k = \frac{\sum_{i=1}^{n} r_{ik} \mathbf{x}_i}{\left\| \sum_{i=1}^{n} r_{ik} \mathbf{x}_i \right\|}.$$

Algorithm Summary

The k-means algorithm with cosine distance alternates between two steps until convergence:

1. **Cluster Assignment Step**: Assign each point $\mathbf{x}_i$ to the cluster with the highest cosine

similarity:

$$r_{ik} = \begin{cases} 1, & \text{if } k = \arg\max_j \mathbf{x}_i^\top \mathbf{c}_j, \\ 0, & \text{otherwise.} \end{cases}$$

2. **Centroid Update Step**: Update the centroid of each cluster as the normalized mean of its assigned points:

$$\mathbf{c}_k = \frac{\sum_{i=1}^n r_{ik}\mathbf{x}_i}{\|\sum_{i=1}^n r_{ik}\mathbf{x}_i\|}.$$

Conclusion

By normalizing the data points and centroids, the k-means clustering objective can be expressed in terms of both cosine distance and Euclidean distance. The equivalence:

$$\|\mathbf{x}_i - \mathbf{c}_k\|^2 = 2(1 - \mathbf{x}_i^\top \mathbf{c}_k)$$

enables seamless interpretation and implementation in the algorithm.

8.2.1 Evaluation

Full dataset (unbalanced): contains all control(untreated): 472, all single dose: 103, all exploded: 40 images, all drug screen visualy similar/closer to single dose.

40 subset (Balanced to the minimum nof set in the group which is exploded. ie there is only total 40 of exploded): contains random but make sure it have exploded like controls total 40, and all exploded which is basically 40, 10 ds close to sd, 30 single dose.

Curated Full dataset (unbalanced): contains all control except controls have debris: 280, all single dose: 103, all exploded: 40 images, all drug screen visualy similar/closer to single dose: 22.

Curated 40 subset (Balanced to the minimum nof set in the group which is exploded. ie there

is only total 40 of exploded but excluded explod look alike from control): contains random but make sure it doesn't have exploded like controls total 40, and all exploded which is basically 40, 10 ds close to sd, 30 single dose.

run 100 times for different random initilisation.

Table 8.3: Dataset Composition Overview

Dataset Type	Control	Single Dose	Exploded	Drug Screen	Notes
Raw Datasets					
Full	472	103	40	–	Unbalanced
Balanced-40	40	30	40	10	Minimum set balanced
Curated Datasets					
Full	280	103	40	22	Debris-free controls
Balanced-40	40	30	40	10	No exploded-like controls

Note: Drug screen images are visually similar to single dose treatment. All balanced datasets are normalized to the exploded group size (n=40).

Table 8.4: Evaluation Results on Different Datasets and Augmentations with cosine distance

Projection Head	Augmentation Type	Full Dataset (Unbalanced)	40 Subset (Balanced)	Curated Full Dataset	Curated 40 Subset
Before	Strong	93.56	95.83	98.43	100
	Sweet	93.72	100	100	100
	Resize	93.56	98.33	98.20	100
	Resize No Contrast	92.62	99.17	91.01	99.17
	Sweet No Contrast	97.65	98.33	99.55	99.17
After	Strong	86.19	94.17	89.89	95.83
	Sweet	74.10	72.50	75.96	69.17
	Resize	90.27	94.17	88.99	94.17
	Resize No Contrast	74.1	56.67	68.31	60.83
	Sweet No Contrast	74.10	75.83	75.28	76.67

Table 8.5: Evaluation Results on Different Datasets and Augmentations with euclidean distance

Projection Head	Augmentation Type	Full Dataset (Unbalanced)	40 Subset (Balanced)	Curated Full Dataset	Curated 40 Subset
Before	Strong	93.72	99.17	99.1	100
	Sweet	98.12	93.33	99.30	100
	Resize	90.27	85	89.89	82.50
	Resize No Contrast	93.56	98.33	91.01	99.17
	Sweet No Contrast	99.06	99.17	99.78	99.17
After	Strong	92.15	95	89.89	95.83
	Sweet	74.10	71.67	73.71	68.33
	Resize	74.10	84.17	75.73	84.17
	Resize No Contrast	74.10	56.67	68.76	57.50
	Sweet No Contrast	74.10	70.83	78.88	72.50

Should I need to add precision recall? yes to confirm control exploded lokk like is the one who reduces the accuracy we need to know ds close to sd is gped correct maybe we dont need to

do precision we can check the file path. if yes then we don't necessarily need to do the sd vs others test. maybe for full dataset where accuracy lw and included control looks like explod

redo balanced resize training with simclr

Table 8.6: Evaluation Results Using Different Distance Metrics

	Metric	Full Dataset (Unbalanced)	40 Subset (Balanced)	Curated Full Dataset (Unbalanced)	Curated 40 Subset (Balanced)
Original Images	Cosine Distance	93.09	65.83	82.25	65.83
	Euclidean Distance	82.26	67.5	72.81	68.33

Include PCA 1 here? maybe in ranking. but we can add PCA 2 fugures

do sd VS others all

8.3 Direct day 7 to day 10 distance evaluation

calculate distance between day 7 untreated and its corresponding day 10 treated one. Idea is: 1. it should give same distance between day 7 and its corresponding day 10 even its changed in position/flipped. ie checking whether simlcr learned to be invariant to position error in microscope error because of manual handling/transporting. 2. it sould give same distance even if its blured/sharpened 3. it should give sma edistance even if its changed in brightness 4. main test for k means centroid approach. it should give different distance to single dose and exploded.

basic evaluations for ranking: 1. whether it learned to be invariant the position change: do flipps and calculate the cosine distance from control to treated flipped versions. (i don't expect it learn to the change in center of position, if it learns good we can say center crop have some effect maybe? but not for the edge one?) 2. shape invariant:control to all single dose should be almost same cosine distance.

it also applicable to time prediction and reconstruction ranking evaluation and also from kmeans centriod approach.

We will start with cosine distance and euclidean distance. other distcances based on time if we

Augmentation type		Strong	Sweet	Resize	No contrast Resize	No contrast sweet
Before	Flip and rotation	0.1932	0.0109	0.134	0.0120	0.0108
	Blur and sharpness	0.0061	0.0185	0.171	0.0024	0.0058
	Brightness change	0.0003	0.0037	0.184	0.0013	0.0043
After	Flip and rotation	0.3435	0.0204	0.0747	0.0113	0.0212
	Blur and sharpness	0.0108	0.0388	0.2087	0.0047	0.0133
	Brightness change	0.0005	0.0067	0.2864	0.0024	0.0078

have

we are trying to find whether simclr learns something, from simclr side cond 7 to same cond7 augs should have high cosine similarity and within all gps foe example within sd higher cosine similarity since its gping this is what we have to check but we will do this only after ranking strategy depend on time

then if have time we will do cond7 to cond10, sd stuuf

‘‘‘

Flip and Rotation	**Blur and Sharpness**	**Brightness Change**
0.0291	0.0001	5.73×10^{-8}

Table 8.7: Metrics for different augmentation types.

9 Methodology for Ranking

since we don't have ground truth labels to rank the images except control images. My strategy was to simplify the current ranking problem to only scale/order/rank images using only control, single dose and exploded images. The reason to pick these groups is that controls we know that there is no drug applied which means that there is no effect of drug at all. single dose images are the category which is clinically recommended at the moment (eventhough we don't know how much drug effected or how much it killed the cancer) and exploded are visually exploded from the original cancer cell meansing we can visually see debris around the cancer cell potentially which may potentially harm the surrounding goof cells. once if we can order those small subset of entire images, we can add the other image as inference to see where they plotted relative to control or single dose or exloded in this scale.

9.1 Ranking strategy 1: Using CAE

9.1.1 Day7 to day10 predcition

mention problem of data lack here instead in research questions?

1. **Step 1:** Create a latent space representation of all images, including untreated, clinically recommended, and drug screening images, using SimCLR. The idea is that SimCLR effectively learns efficient features of similar images that are not captured by human-interpretable metrics. We expect the SimCLR feature vectors of similar images will be closer in the latent space. In other words, feature vectors of similar images will be more linearly separable.

2. **Step 2:** Train a prediction model exclusively on the representations of untreated images from Day 7 to Day 10. (Input: Day 7 feature vector and target: Day 10 feature vector)

3. **Step 3:** Perform inference on the representations of test images, which include untreated, clinically recommended, and drug screening images.

4. **Step 4:** Perform step 2 and step 3 on images instead of simclr feature vectors for comparitive study.

Since the day 10 prediction model is trained solely on the representations of untreated images, the inference loss/metric (i.e., the difference between the predicted and actual Day 10 image representations) will be very small for untreated images. Conversely, the inference loss/metric will increase for treated images as their representations deviate from those of untreated images. This inference loss/metric will be used as the feature for the ranking/order scale, where the initial images will start with untreated images that have very small inference loss/metric, and the scale will end with images having high inference loss/metric in ascending order. Determining a reasonable inference loss/metric will be one of the research problems to tackle.

9.2 Ranking strategy 2: K means centroid approach

This strategy utilizes the after projection head vectors since simclr loss function designed that after projection head vectors have cosine similarity between similar group.

1. **Step 1:** Feed control images into k means and find the centriod of control (untreated) cluster based on both cosine distance and the euclidean distance. (we can choose the distacne metric if we have time)

2. **Step 2:** calculate the euclidean/cosine distance from this centroid to every images.

3. **Step 3:** Perform the same operation for simlcr features and on original images.

change metric to mistakes from center gp to teft and right.

Group-Wise Ranking Accuracy: Mathematical Definition and Process

below maybe wrong because I didN't added that gp order determine by calculating mean of cosine distance

Definitions

Let $G_1, G_2, G_3, \ldots, G_n$ represent n different groups of distances. The distances in each group G_i are denoted as:

$$D_i = \{d_{i1}, d_{i2}, \ldots, d_{im_i}\},$$

where m_i is the number of elements in group G_i.

Let:

$$\{D_1, D_2, \ldots, D_n\}$$

represent the collection of all groups.

After sorting all the distances across the groups into a single list, we check if the order of groups is maintained, i.e., whether all distances in G_1 are less than those in G_2, all in G_2 are less than those in G_3, and so on.

Mathematical Formula for Group-Wise Ranking Accuracy

Correct Transitions A correct transition between two groups G_i and G_j (with $i < j$) occurs if all elements of G_i are less than all elements of G_j after sorting.

This can be expressed mathematically as:

$$\text{Correct Transition}(G_i \to G_j) = \left[\forall d_{ik} \in G_i, \forall d_{jl} \in G_j : d_{ik} < d_{jl}\right], \quad \text{for } i < j.$$

Total Possible Transitions The total number of possible transitions is the number of adjacent group pairs:

$$T_{\text{total}} = n - 1.$$

Group-Wise Ranking Accuracy The ranking accuracy is the ratio of correct transitions (T_{correct}) to total possible transitions (T_{total}):

$$\text{Accuracy} = \frac{T_{\text{correct}}}{T_{\text{total}}}.$$

Step-by-Step Process

1. **Sort the Distances:** Sort all the distances from all groups into a single list while keeping track of the group each distance belongs to.

2. **Check for Correct Transitions:** For each adjacent group pair (G_i, G_j), check if the condition:

$$\forall d_{ik} \in G_i, \forall d_{jl} \in G_j : d_{ik} < d_{jl}$$

holds true.

3. **Count Correct Transitions:** If the condition holds for a group pair, increment T_{correct}.

4. **Compute Accuracy:** Finally, compute the group-wise ranking accuracy as:

$$\text{Accuracy} = \frac{T_{\text{correct}}}{T_{\text{total}}}.$$

Curated control dataset: 280 sd: full: 103 exploded full: 40

ds close to sd 10

Projection Head	Distance From	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose Mean	92.42	91.71	92.42	91.47	92.42
	Control Mean	94.55	92.65	93.84	95.73	94.55
	Explod Mean	82.23	82.23	92.65	83.65	81.52
After	Single Dose Mean	**97.39**	87.68	92.89	82.94	83.65
	Control Mean	86.97	76.78	90.76	76.30	82.70
	Explod Mean	92.18	79.62	88.86	77.96	82.23

Table 9.1: Cosine distance

Projection Head	Distance From	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose Mean	91	89.10	91.23	86.49	90.52
	Control Mean	87.68	86.73	88.15	82.94	86.73
	Explod Mean	81.99	81.52	92.65	83.18	81.52
After	Single Dose Mean	90.05	81.28	91.94	77.25	78.67
	Control Mean	82.23	76.07	84.12	76.54	74.17
	Explod Mean	90.52	78.20	94.79	75.83	81.75

Table 9.2: Euclidean distance

add figures below.

other notable metrics that achieved around 95 percentage accuracy are pearson distance by resize before explod mean with 96.69. hence we need supervised classifier for explod then its fine.

With this costumized metric, we get around 95 if there is slightest seperation for 3 gps.

after projection head: strong seperates perfectly but the problem is since the control is seperated at the middle ordering. Hence we can find the less effective in the middle,

clear seperation between sd and ex (2 gp) and max mean difference between them table:

Distance From	Single Dose Mean	Control Mean	Explod Mean
Cosine	81.28	86.02	76.78
Euclidean	83.18	78.44	80.81

Table 9.3: Your table caption here

9.3 Ranking strategy 3: Softmax approach

This strategy utilizes the before projection head vectors since its linearly seperable for downstream task classification as suggested in original simclr paper.

We will use simclr features since original images are not able to classify 100 percentage as we seen in intermediate evaluation. 1. Train classification model to classify cond7 and sd. 2. Then do inference on them and take softmax probability as metric for ranking. below table used 750 epochs, batch:8: all of these value can improve by performing regularizytion and more layers or less layers basically hyper parameter tuning. i didn't do it, since i was focused on methodolgy. Below trained for 750 epochs:batch:8

Classification	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Cond7 vs SD	97.89	99.29	98.59	98.25	100
Cond7 vs Ex	97.54	96.49	98.94	100	97.54
Ex vs SD	95.78	98.94	95.08	92.98	97.54

Table 9.4: Table description goes here.

Depend on time do ex twenty nine

this questions if strong or No contrast sweet is betterr? because previously strong was better. Now sweet performed well for this task. to understand why sweet performs better we need to see crop change and brghtness, contrast change seperatly.

When we used curated dataset percentage of gp wise accuracy went down where when we used control as the same number of images of sd accuracy is 100. it worked maybe because of class balancing.

How do we check which data aug is working? it should seperate cond7, ex, sd: 100 percentage. because if get 100 that means it is able to make gp of control and sd for sure, from ex mixing.

why are we doing this because every aug can seperate cond 7 and sd. thats obivious. because there is no other gps. so we have to add new gp as inference to see if cond7 and sd still gp together. same principle in kmeans approach.

so, with this approach from cond7 to sd classsification softmax probability score we get mixture of exploded and gray from drug screen.

if we want to avoid that we can train supervised classification to filter out the exploded from there then put them on right side of sd so that we get clean scale. for that training we can use simclr feature vectors because it gave more accuracy than the original image accuracy.

put the below table in classification and also do classification of ex fourty others because we are usign it for softmax approach

Augmentation Type	Metric	Strong	Sweet	Resize	Resize No Contrast	Sweet No Contrast
Before Projection Head	Train Accuracy (%)	97.07	96.23	96.23	97.91	97.07
	Train Epoch	30	41	128	465	485
	Test Accuracy (%)	95	91.67	93.33	93.33	95
	Test Epoch	5	5	21	12	17

Table 9.5: Performance metrics for different augmentation strategies before the projection head.

include other classification combinations in table

Why its not giving 100 because now the problem is harder. we had 2 classes, explod vs all other drug screen,where drug screen look intermediate to explod and sigle dose. Also we can improve this result by adding more hidden layers instead of just one linear layer and so more. at the moment i didn't since we want to know which data aug works for harder problem. because in the intermediate evaluation most of them gave 100 percenatage making it not evaluator for evaluating inbetween data augs. there we get that orig vs data augs.

Table 9.6: Original Image Results

Metric	Train Accuracy (%)	Train Epoch	Test Accuracy (%)	Test Epoch
Original Image	84.94	434	78.33	19

ordered images can found in the below gdrive links for both mixed and cleaned order for ds and ex. show them ordered images for 80 percentage gp wise accuracy ie only seperated 2 gps, show them its worse.

its better to not use exploded and harm people sentence from intro and data intro just say debris, because cond10 and exploded have debris. cond10 have debris even without drug so debris can happen without drug.it is probabliy because its cultivated in lab where these things happen.

9.4 Ranking strategy 4: PCA variation

- First cluster points that exceed the minimum value of the middle cluster - Third cluster points that fall below the maximum value of the middle cluster

why we choosed cluster 40 dataset because thats the one give 100 accuracy for clustering.

mistakenly this is inference i icluded ds close in sd. so when do real inference do train we used clearly differentiable curated cluster 40 dataset. after projection and original iamges couldnt make 100 percentage clustering so we didn't need to do pca.

Metric	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Cosine Distance	100	90.0	91.67	68.33	95.83
Euclidean Distance	99.17	79.17	69.17	88.00	97.50

Table 9.7: Table showing distances for different augmentation strategies.

using these we can order but in one end of the scale 2 gps will be mixed.

pca inference acc strong cosine: 90.14 sd 40 + ds close 22. pca eucli: 88.73

10 Conclusion

10.1 Final Evaluation

Ranking Strategy	Old Accuracy	New Accuracy	Accuracy Reduction
Softmax: cond7 vs sd	100	100	0
Softmax: cond7 vs Ex	100	100	0
K-means: Before resize Pearson distance using explod mean	96.69	95.50	1.19
K-means: After cosine strong using single dose mean	97.39	93.33	4.06
PCA: strong cosine distance	100	90.14	9.86
PCA: Euclidean strong	99.17	88.73	10.44

Table 10.1: Performance metrics across different ranking strategies.

How do you know which ranking strategy is better visually? inference from ds close tosd for the worked one from each strategy and look which one works better?

10.2 Future research direction

10.3 SimCLR vs Original

10.4 3 channel vs 1 channel

10.5 Unet vs Resnet

10.6 96 vs 256

10.7 Batch size 16 vs 64 vs 128 vs 256

1. Hybrid: Include human-interpretable features in the unsupervised learning features combined effect.

2. Weakly supervised DINO transformer-based approach.

3. Weakly supervised SimCLR approach.

4. Other distance-based approaches.

Appendix

Bibliography

[1] Mathilde Caron et al. *Emerging Properties in Self-Supervised Vision Transformers*. 2021.

[2] Ting Chen et al. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020.

[3] Sofia Dembski et al. "Establishing and testing a robot-based platform to enable the automated production of nanoparticles in a flexible and modular way". In: *Scientific Reports* (2023).

[4] Andre Esteva et al. "Dermatologist-level classification of skin cancer with deep neural networks". In: *Nature* (2017).

[5] Jean-Bastien Grill et al. *Bootstrap your own latent: A new approach to self-supervised Learning*. 2020.

[6] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015.

[7] Jeremy Irvin et al. *CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison*. 2019.

[8] Prannay Khosla et al. *Supervised Contrastive Learning*. 2021.

[9] Ziyu Liu et al. *Self-Supervised Learning for Time Series: Contrastive or Generative?* 2024.

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015.

[11] Xiaosong Wang et al. "ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 3462–3471.

[12] Xinyu Yang, Zhenguo Zhang, and Rongyi Cui. “TimeCLR: A self-supervised contrastive learning framework for univariate time series representation”. In: *Knowledge-Based Systems* 245 (2022), p. 108606.

[13] Yuhao Zhang et al. *Contrastive Learning of Medical Visual Representations from Paired Images and Text*. 2022.

Declaration on oath

I hereby certify that I have written my master thesis independently and have not yet submitted it for examination purposes elsewhere. All sources and aids used are listed, literal and meaningful quotations have been marked as such.

Bibin Babu, January 15th 2025

Consent to plagiarism check

I hereby agree that my submitted work may be sent to PlagScan (www.plagscan.com) in digital form for the purpose of checking for plagiarism and that it may be temporarily (max. 5 years) stored in the database maintained by PlagScan as well as personal data which are part of this work may be stored there.

Consent is voluntary. Without this consent, the plagiarism check cannot be prevented by removing all personal data and protecting the copyright requirements. Consent to the storage and use of personal data may be revoked at any time by notifying the faculty.

Bibin Babu, January 15th 2025