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.

Thank Prof. Magda for guiding me to how to think scientifically and also for the SimCLR and autoencoder prediction ranking startegy.

Thank Prof. Jan for the softmax approach idea.

Thank Dalia for the code for sharp layer calculation and thanks for the biological stuff explanation and the once in a time opportunity.

Thank Phillip for the cell viability test idea.

Thank stefan for the microscope focal length explanation.

Thank Philipp for the resize and crop augmentationc idea and general explanation for th edata augemnetation.

Thank Dad and Mom for this life. Without you guys I won't be able to write this.

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.

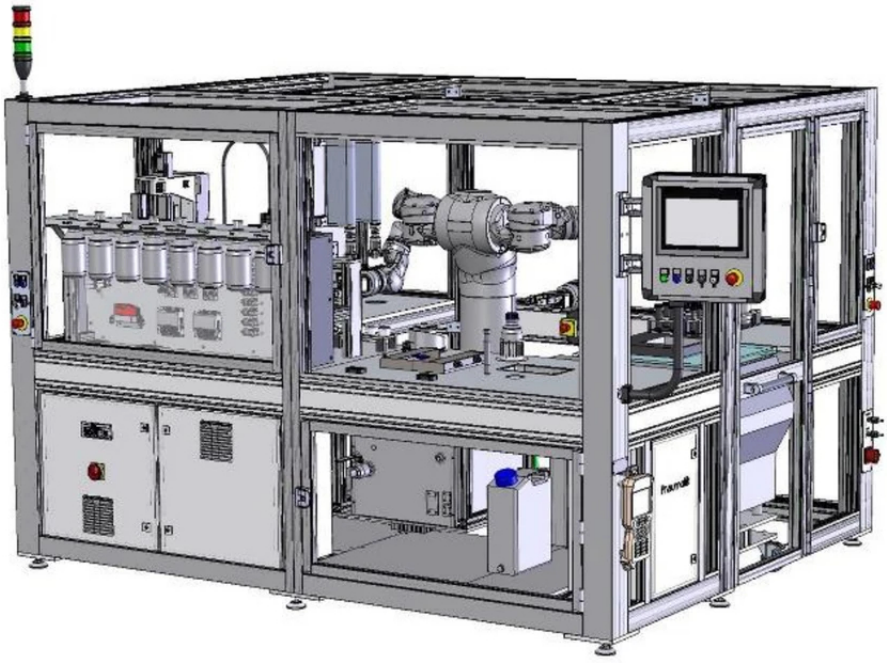

Figure 1.1: Robo platform

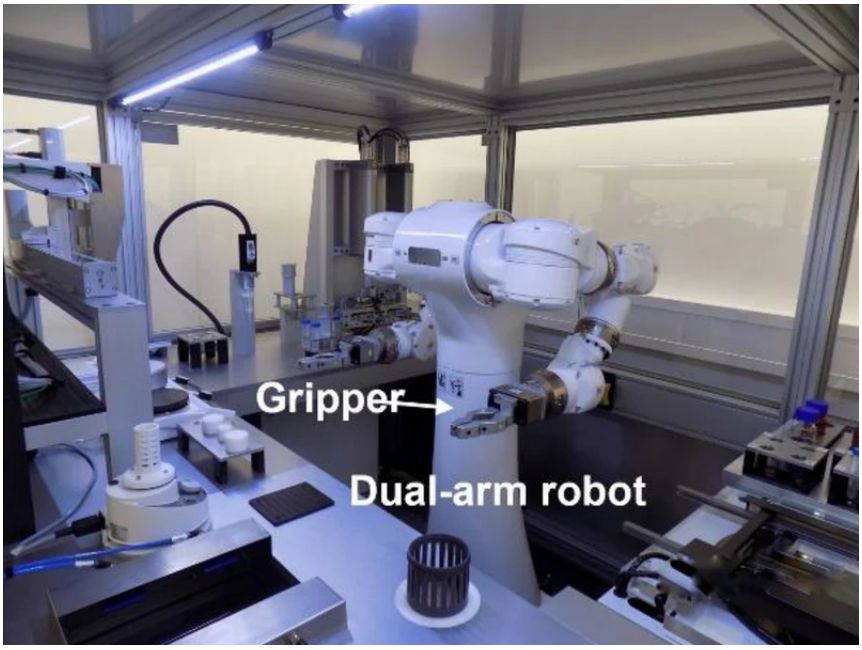

Figure 1.2: Dual-arm robot

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.

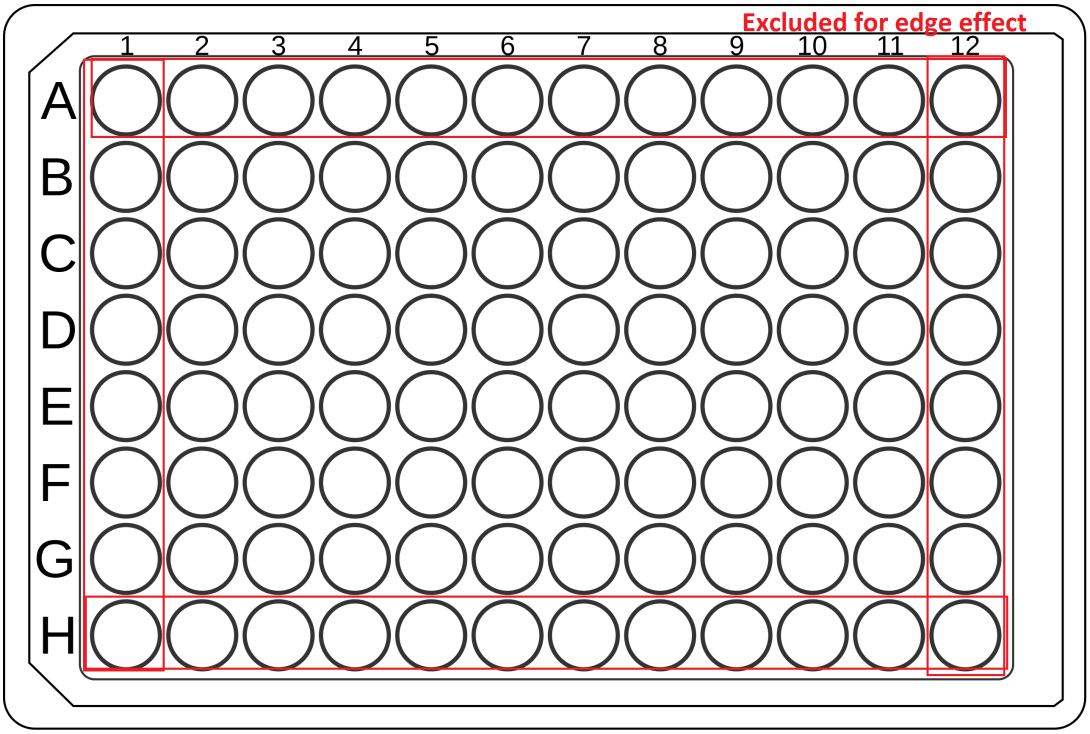

Figure 1.3: A well plate containing 96 wells where rows A, H and columns 1, 2 are excluded due to edge effects.

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

1. Control (0 percentage drug applied) For 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. Control Day 10: The untreated images show a morphological change where the center of the tumor tissue becomes darker, resembling the darkness observed when drugs are applied on Day 10. This occurs due to the natural death of tumor cells within the tissue, caused by a lack of nutrients to sustain the cancer cells.

4. Drug Screening: Different drug combinations and concentrations are used in this experimental study to evaluate drug efficacy. Some images may resemble single-dose images, while others may resemble cond10 if the drug is ineffective. Additionally, certain drugs may or may not result in the destruction of surrounding non-tumor cells in the human body, potentially causing side effects. For simplicity, we refer to this category as ”Drug Screening.” These are the images that require relative assessment through ranking.

explain here what is explod

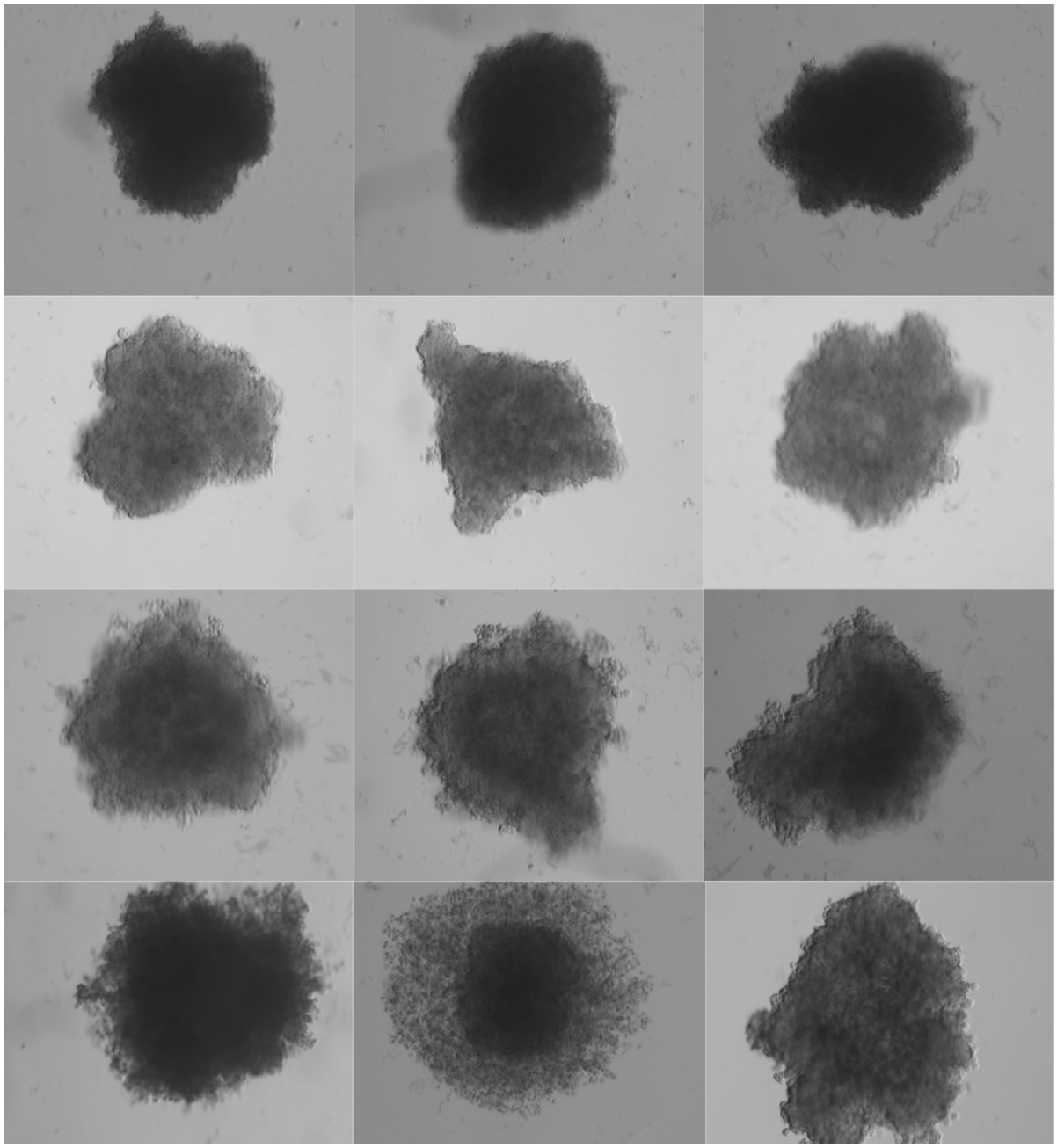

Figure 1.4: Three different types of images: Drug Screened, Single Dose, and Untreated as mentioned in section 1.1.

We apply single dose and drug screened combinations in different well plate settings as illustrated inFigure 1.5 and figure 1.6

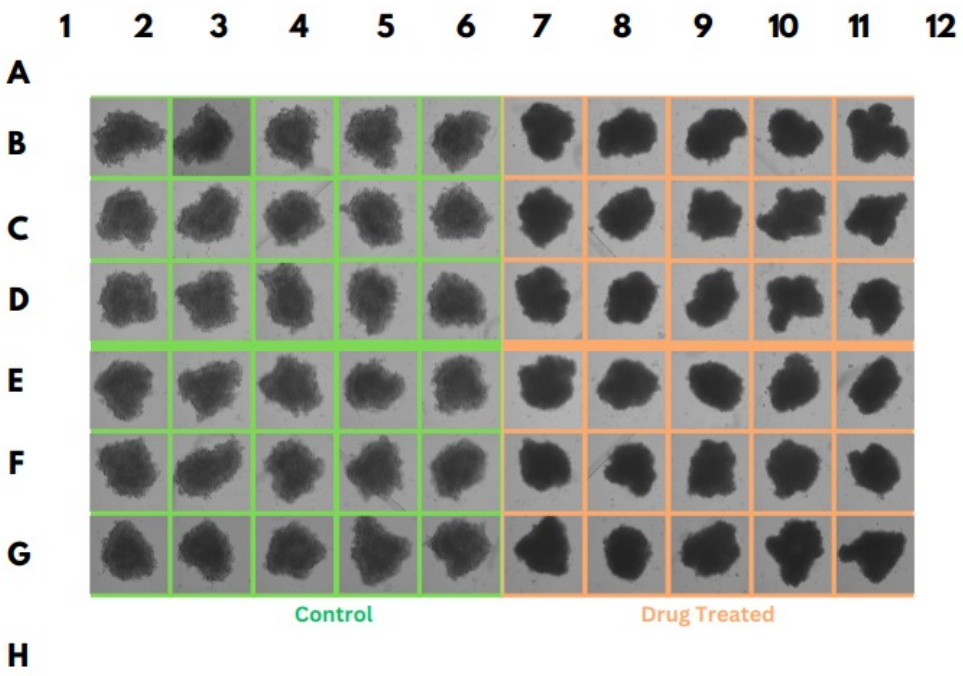

Figure 1.5: Well plate setup for the single-dose experiment where the left half remains untreated and the right half is treated with a single drug concentration. This image was taken three days after drug application.

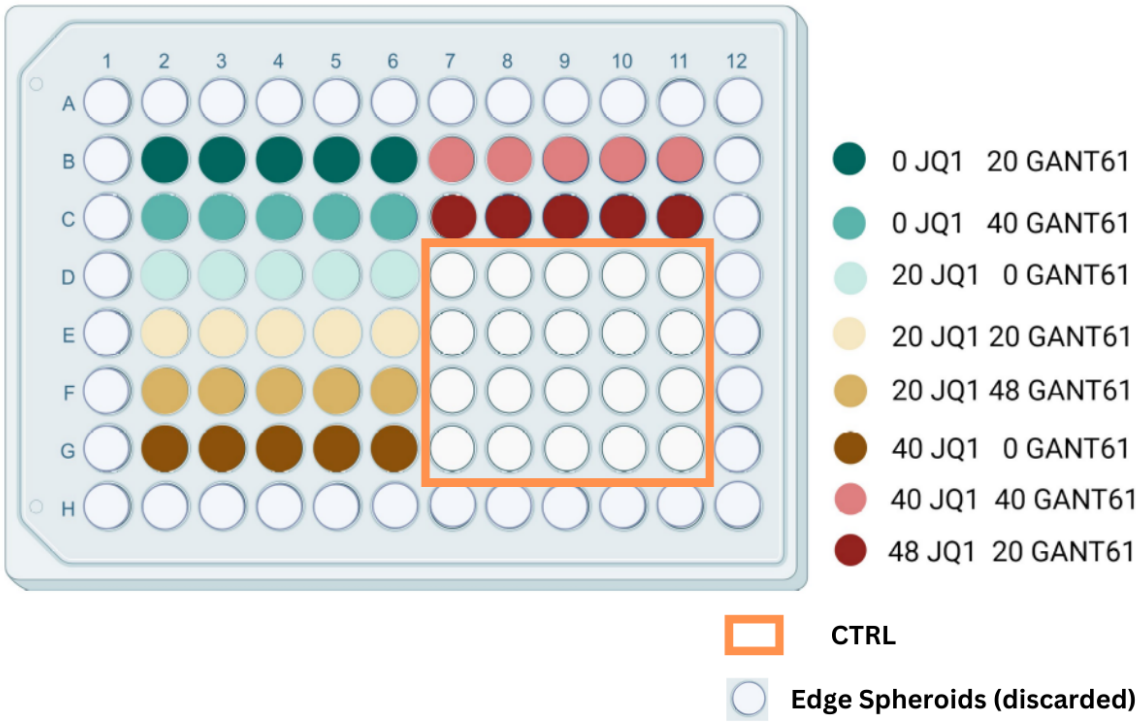

Figure 1.6: Well plate setup for the drug screening experiment where the majority of tumor tissues are treated with different combinations of drug concentrations (multi-colored wells), while some are left untreated (white wells bounded by orange box).

1.7 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.

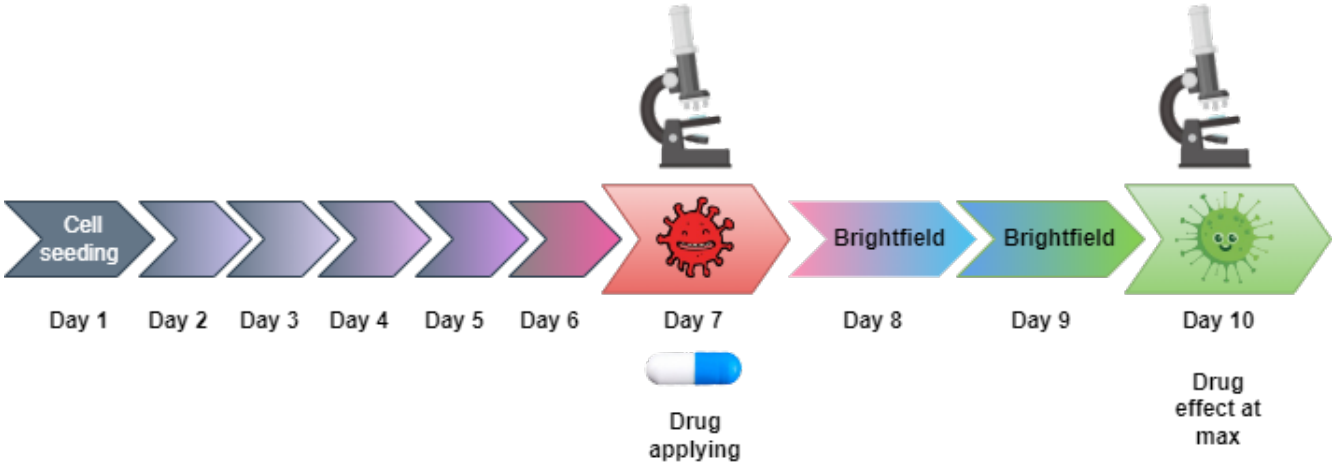

Figure 1.7: Illustrates the flow chart of time evolutoion of 3D tumor tissues.

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 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.

one more thing. You had a master thesis proposal in the beginning which actually listed the research questions as well as the methodology. Your final manuscript shall reflect it and follow it. When you diverted from the original proposal, you shall explain why you diverted.

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.

Research Questions

1. Can we learn latent features that capture the alterations induced in tumor tissue models by drug application over a period of time from bright-field microscopy images?

2. Will these features effectively establish a relative ranking assessment of drug efficacy?

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

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

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 [10] ; 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 [12] 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, [11] 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.

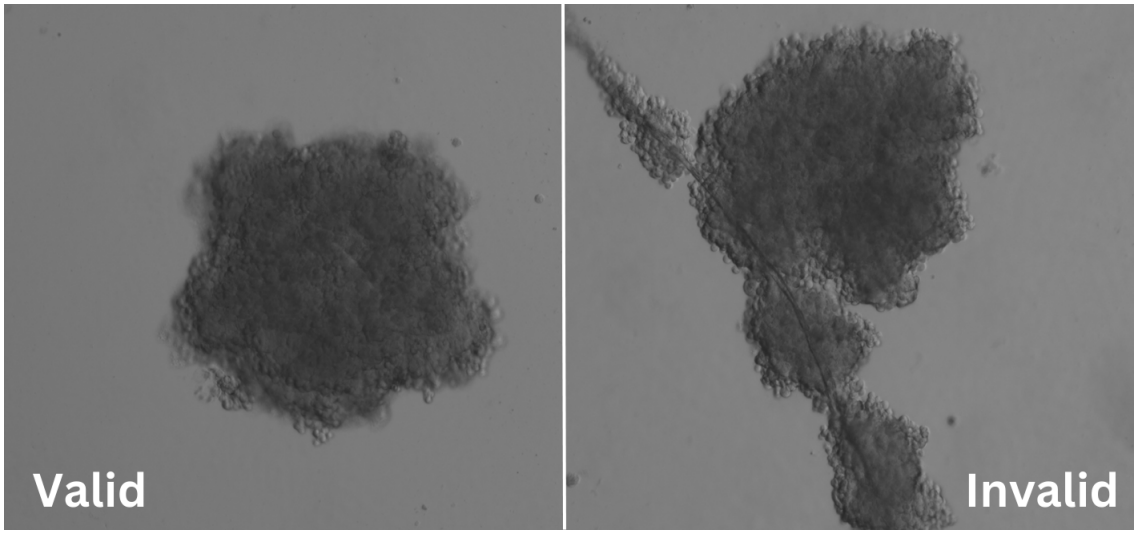

Figure 5.1: invalid vs valid

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.

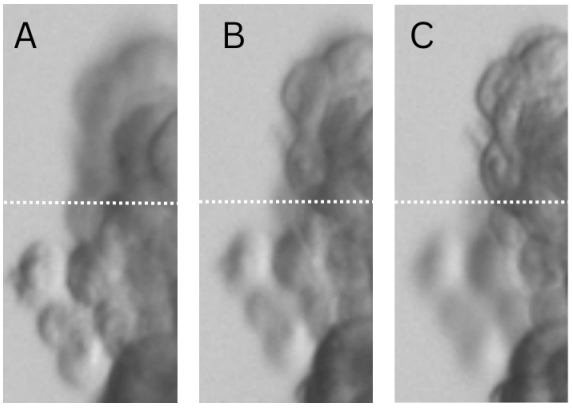

Figure 5.2: invalid vs valid

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.

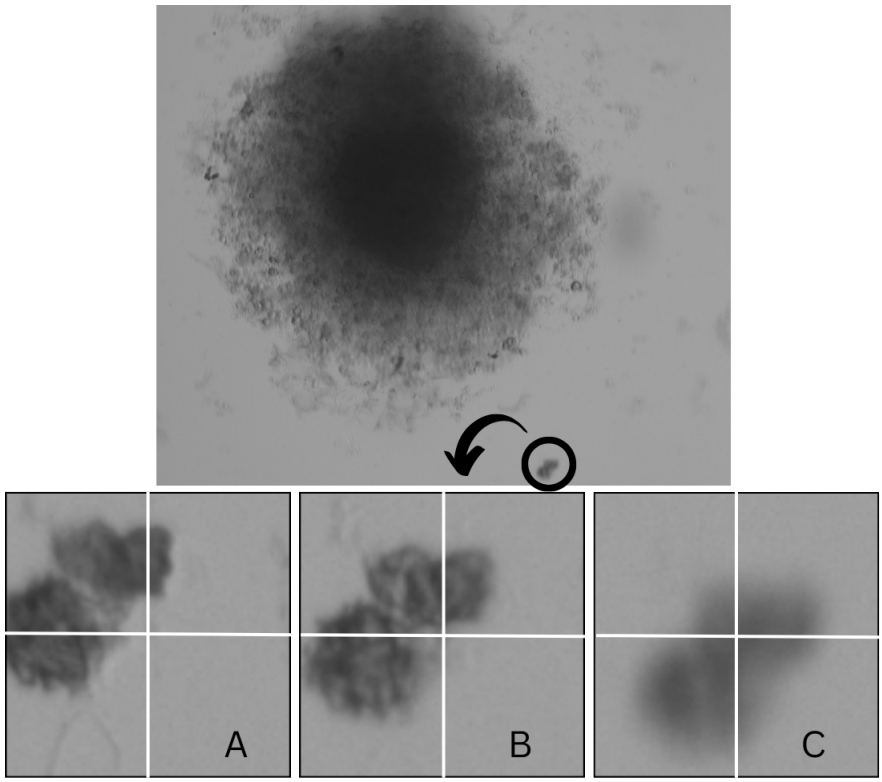

Figure 5.3: Misalighned

We observed some temporal noise/artifacts, as shown in figure 5.4 in different category of images such as untreated, cond10, single dose, explod. The presence of artifacts in the well-plates could be attributed to environmental factors such as temperature, humidity, or CO_2, or by some inconsistencies in sample handling. Such factors are likely to cause variations in cell behavior and inconsistency in the acquired images. These images were included as-is for SimCLR training, providing an opportunity for the model to learn from realistic conditions. However, I didn't artificially introduce additional noise or artifacts through data augmentation. This approach highlights a potential area for further exploration in future work.

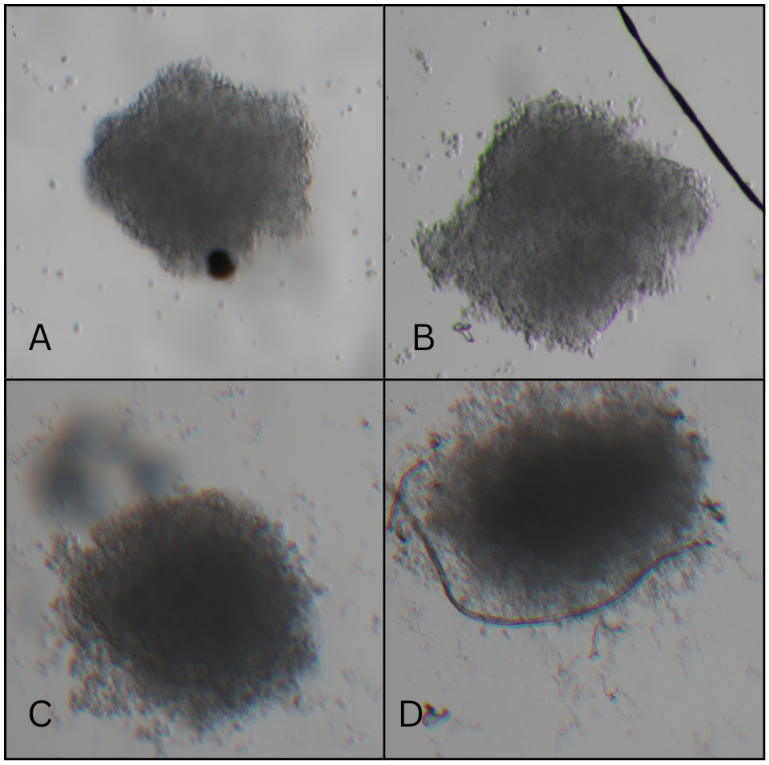

Figure 5.4: Noise

organised folders

Figure 5.5 illustrates that, even with the The same drug concentration applied to the spheroids under controlled conditions occasionally resulted in varying effects across different experiments. This variation could result from slight differences in spheroid composition, or undetected microenvironmental influences that affect drug absorption by the spheroid and cellular response. initially i thought I will be able to evaluate the effect by drug combination , but because of this difference, we need to make the ranking relative assessment based on the day 10 images neverthless its appied by same drug combo.

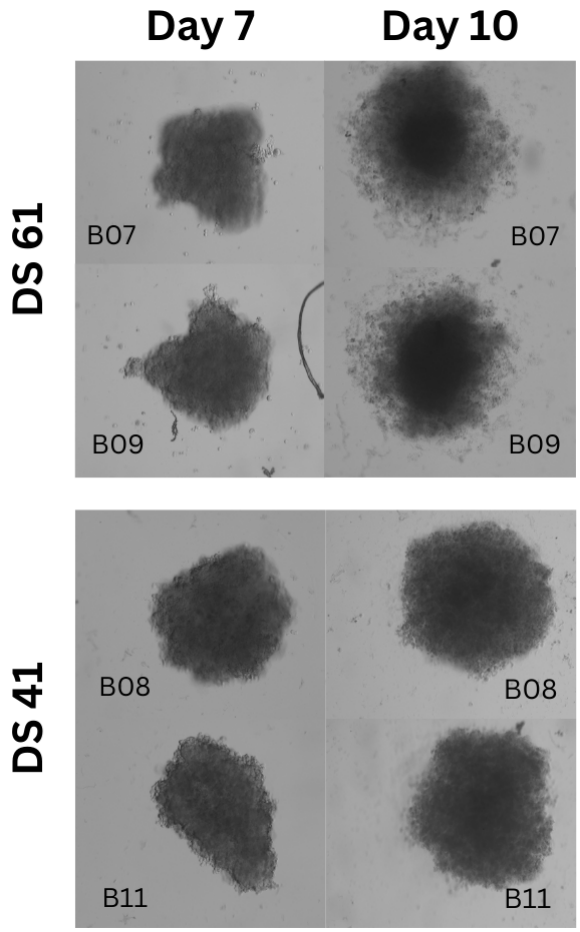

Figure 5.5: gp 8 drug applied

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 (16.05%)	103 (8.26%)	231 (18.54%)	472 (37.89%)	240 (19.26%)	1246

Table 5.1: Dataset Class Overview

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 because 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 (from 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.

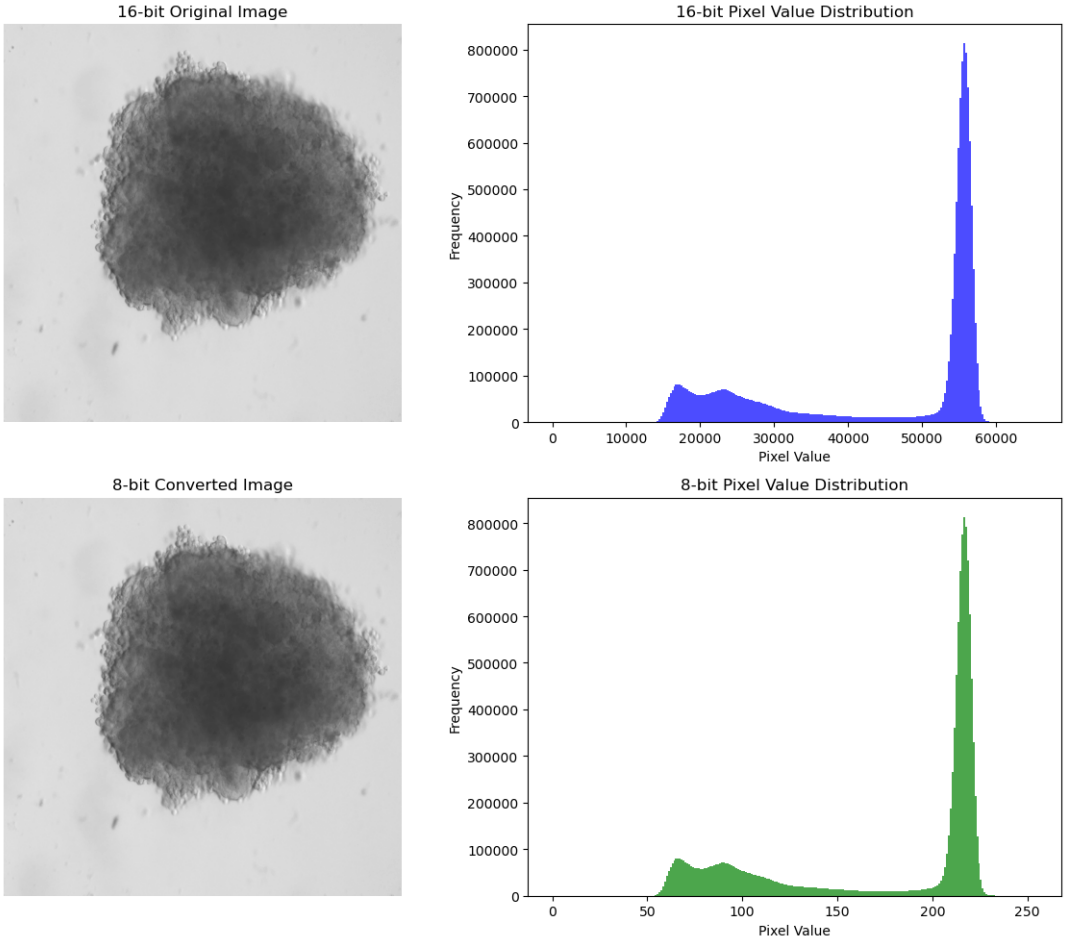

Figure 5.6: 8-bit vs 16 bit data lose comparison

5.2 Challenges

1. **Limited Dataset for Day 7 to Day 10 ranking prediction Model:** The dataset for predicting outcomes between Day 7 and Day 10 is limited, which poses a challenge for training and evaluation.

2. **Issues with Day 10 Images:**

   a) **Image Variations:** Day 10 images can exhibit variations such as flipping, blurring,

brightness changes, and position changes from the original position of tumot tissue cell in the image. The position change occurs because, when the well plate is brought outside and then placed back under the microscope for taking day 10 image, the position often shifts. This relative position of the tumor cell in the image can shift from the center to other directions. While a 'center crop' explained in in Section 7.1 approach can address this issue to some extent, it fails when the tumor cell is located at the edge of the image. To deal this issue for some extend, Applying horizontal and vertical flips, rotations by 90° and 270°, as well as combinations like horizontal flip + rotation 90° and horizontal flip + rotation 270°, can help make the model invariant to position changes.

3. **Variability in Drug Effects on Day 10:**

    a) **Effect Differences:** The same drug can have different effects on Day 10. For instance, there is a huge variation between DS 61 G6 and 41 GP6, while less variation is observed between RBTDS 4.1 and 4.2 GP3. Hence we can't assess the drug efficacy based on the drug combination instead we need to make the ranking relative assessment based on the day 10 images.

    b) **Debris Amount:** Images from different groups with significant debris were visually selected. Since the number of such images was small, no specific code was implemented for this selection process.

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

3. Softmax approach, and

4. PCA variation

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.

Section 6 - it seems to be very focused on the simCLR features. I have thought that you tried similar approaches (e.g. the clustering) over some other features, e.g. directly the original images or the ResNet features. Is it not the case? Do you have a baseline model to compare to?

So in each intermediate evaluatiions and ranking strategy we will aslo compare the resnet features to original images.

include original image comparison in the figure

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

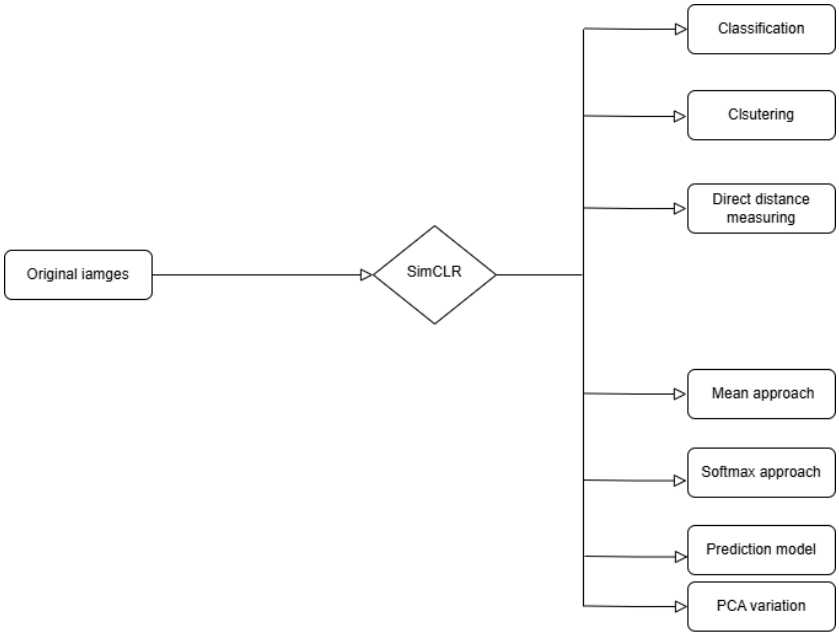

Figure 6.1: Overall framework of the thesis.

7 Methodology for SimCLR

7.1 Data preprocessing

1. **No of channel per image selection**:

   For 3 reasons, I decided to use 3 layers per image as 3 channels of the image for SimCLR training:

   a) Considering each 3 layers per image may capture depth information at 3 different focal planes of a 3D tumor tissue model give more richer information comparing to only single channel information.

   b) For ease of integration with our pretrained architecture ResNet18, which was trained on 3-channel images.

   c) The majority of the images in the dataset have exactly 3 layers, making this a practical choice.

   Images with only 1 channel will have their single channel duplicated to form 3 identical channels. For images with 5 channels, the top 3 sharpest layers are filtered and selected to reduce the image to 3 layers. This ensures consistency across the dataset while maintaining maximum texture and edge information.

   The concept of sharpness is central to selecting the most informative layers. Layers with higher sharpness are retained because they carry more texture and edge information, while layers with lower sharpness tend to have less detail and appear blurred. Sharpness

is calculated intuitively as follows: (Code is from Dalia)

a) Sharpness is related to how quickly pixel intensities change in an image. These changes are measured using gradients, which calculate how intensity changes between neighboring pixels. The gradient magnitude is computed as:

$$g_{\text{norm}} = \sqrt{g_x^2 + g_y^2}$$

where g_x and g_y are the gradients in the horizontal and vertical directions, respectively.

b) After calculating the gradients for all pixels in a layer, their magnitudes are averaged:

$$\text{Sharpness Score} = \frac{1}{N} \sum_{i=1}^{N} g_{\text{norm},i}$$

where N is the total number of pixels in the layer. A higher sharpness score corresponds to layers with more edges, transitions, and details, which are likely to be in focus.

c) Layers are ranked based on their sharpness scores, and the top 3 layers are selected. This ensures that the dataset retains layers that are in focus and have the most texture and edge information.

2. **Croping the image to make whole image from rectangular to square ie H= W**

Advantage of croping to H=W are:

a) Remove unnecessary background which contains no information

b) There will be no shear during resize to 96*96 augmentation like as shown in the figure 7.1

c) minor changes in positions of day 10 image can be reduce with this crop, but itis not a solution if the cancer cell is too edge of the image.

d) only state if you can show fig

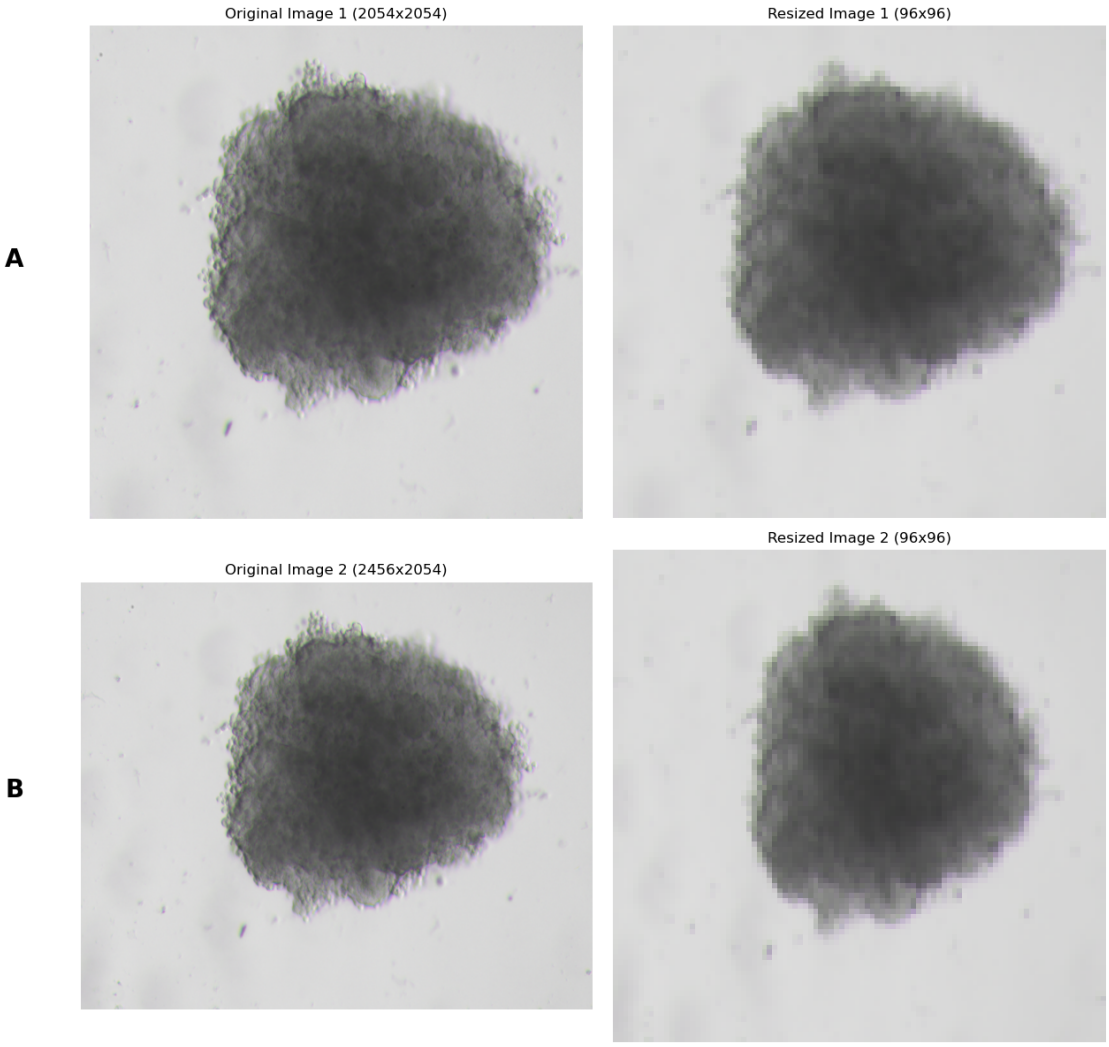

Figure 7.1: First row: croped to 2054*2054 then resized to 96*96: no shear change/elongation in one dimension happened, Second row: original image 2456*2054 then resized to 96*96: shear change/elongation in one dimension happened

e) Augmentations such as 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.

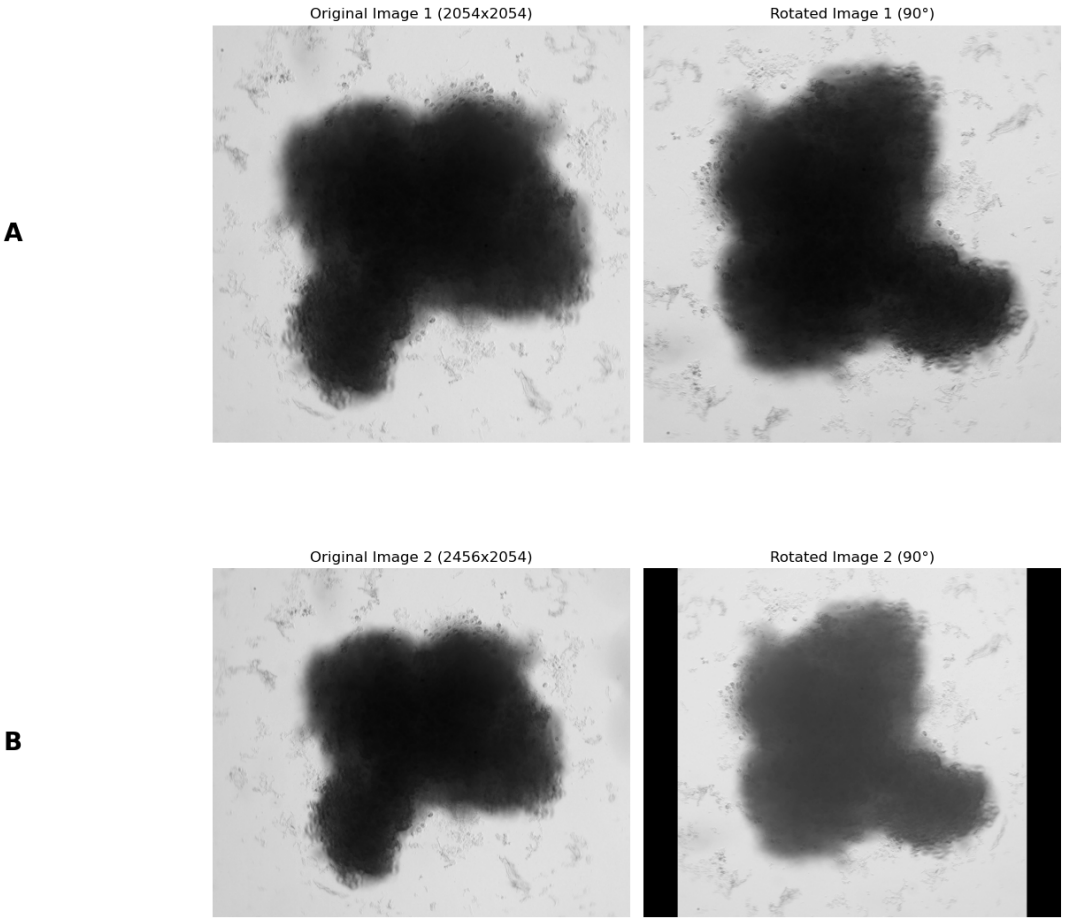

Figure 7.2: A: croped to 2054*2054 then resized to 96*96: No black cuts. B: original image 2456*2054 then resized to 96*96: black cuts happened

We cannot rely on a simple center crop for our images because some cancer cells are not centrally located within the image but are instead closer to the edges. To ensure the cancer cells are fully included in the cropped region, the solution involves identifying the boundaries of the cancer cell including the debris, obtaining the corresponding bounding box, and cropping the image, wich ensures that croped image contain the full cancer cell. The original image dimensions are 2456×2054 (Height × Width). It turns out to be H=W= 2054 ensures that the cancer cell is fully inside for all of the image using computer vision lybraries (will explain below). and any cancer cell that have more than this are either elongated in single dimension (eg. Horizontal) this indicates an invalid cultivation by the robotic system, which will be disgard in the first place through Dalias machine learning model or have have very few amount of debris which I neglected since its very small.

For implementing this procedure, the following steps were employed using the `skimage`, `cv2`, and `numpy` libraries:

a) The image is converted to grayscale by averaging across the three layers and then normalized to an 8-bit format using the `cv2.normalize` function.

b) The Otsu thresholding method (`skimage.filters.threshold_otsu`) is used to create a binary mask, followed by inversion to highlight the cancer cell region.

c) Contours are extracted from the binary mask using `cv2.findContours`. The largest contour, corresponding to the cancer cell or debris, is identified by sorting the contours based on area.

d) A bounding rectangle is determined around the largest contour using `cv2.boundingRect`. The rectangle is then adjusted to ensure the crop is centered around the detected region and fits within the 2054×2054 dimensions. The image is then cropped ensuring that the cancer cell and debris are fully included in the crop. Figure 7.3 demonstrates the whole procedure.

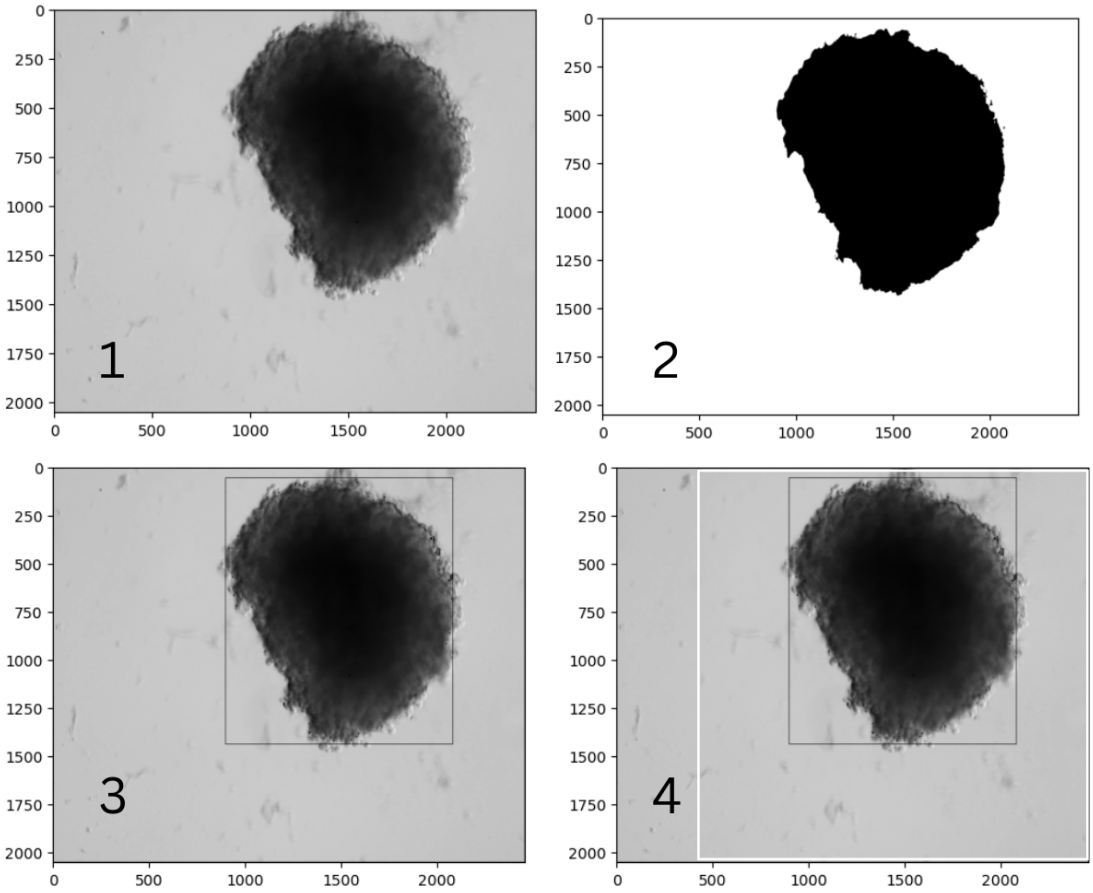

Figure 7.3: crop

3. **Normalize the 16-bit image to [0, 1]**

4. **Perform data augmentations** which detaily explained in the 7.2

5. **Perform Z-score normalization** after data augmentation since:

   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.

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

7.2 Data augmentation

figure add data aug for sweet , resize, ohne figure 7.7 why is anchor the last image and not the first in the left?

I started with best data augmentation pipeline that proposed by simclr [2] paper did, because it gave best performance on natural image dataset such as 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 may 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 7.5 and due to the black cut problem explained in 7.2. We set this as our default for all data augmentation. Figure 7.4 shows the original and corresponding blured image.

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.

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 downstream 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. Figure 7.4 shows the original and corresponding sharpened image

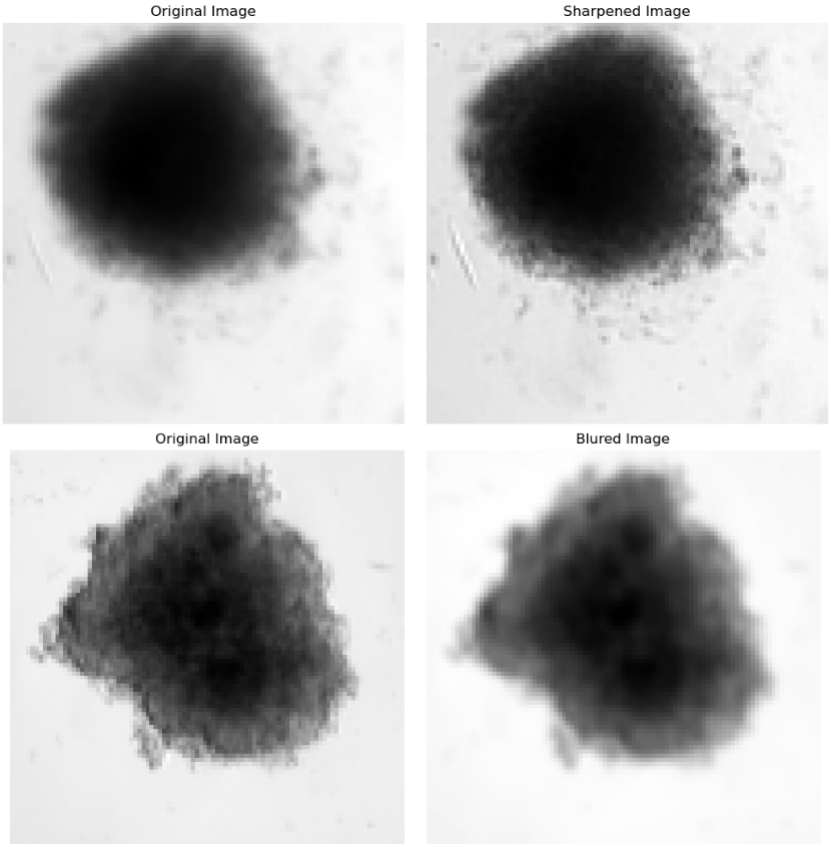

Figure 7.4: Blur and sharped. explain what happens when its blured or sarpened. edge details

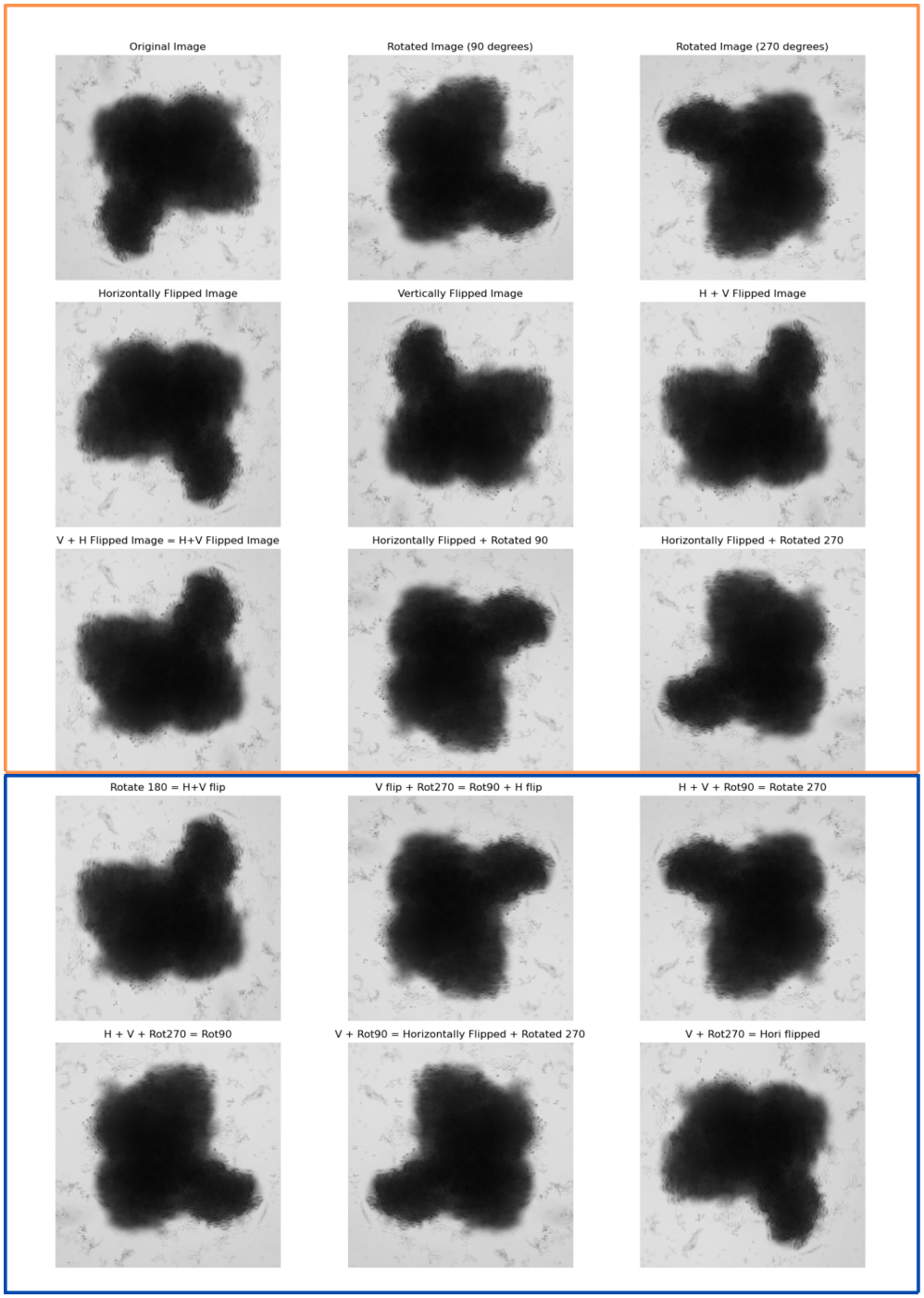

Figure 7.5: Orange box consist of implemented flips and rotations. Blue box consist of avoided flips and rotation combinations because of their repeated pattern to the orange box augmentations.

I call the above chain of data augmentation pipeline parameters as 'strong' data augmentation which illustrated in 7.11 first row.

Since there is a strong intensity change in color jitter explain whats color jitter or directly use

brightness contrast word 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 7.6.

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

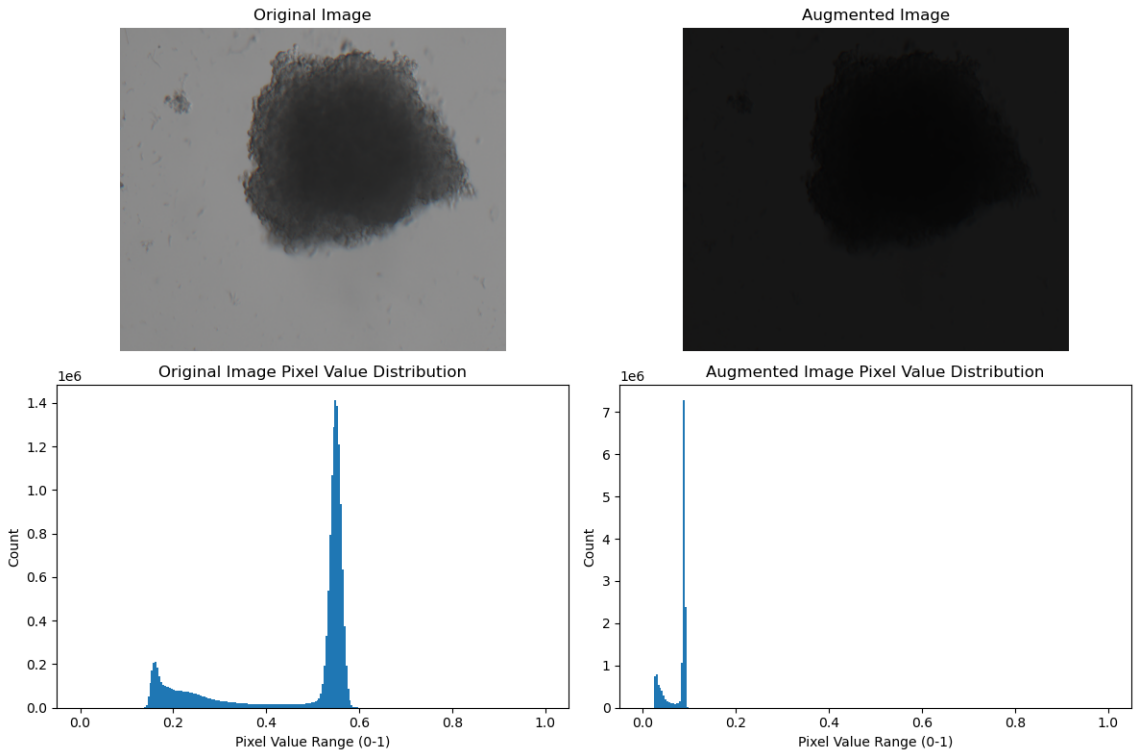

Figure 7.6: 16-bit three-channel image after 3000 epochs of random color jitter applied using PyTorch.

As we seen in the figures 7.6, I found that those strong color jitter augmentations transformed to distribution out of the original distribution. Hence its a good idea to experiment with less intensity change instead of using SimCLR best performed data aug parameters.

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 especially when we have commom as background inner well for every of our images (or natural images atleast cars are only seen in roads, so relating cars or bikes to road 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 reduced to lower=1-0.2, upper=1+0.2 and lower=1-0.35, upper=1+0.35 consequently as showed in figures 7.7 and 7.8. I named this augmentation as 'sweet'. which illustrated in 7.11 second row.

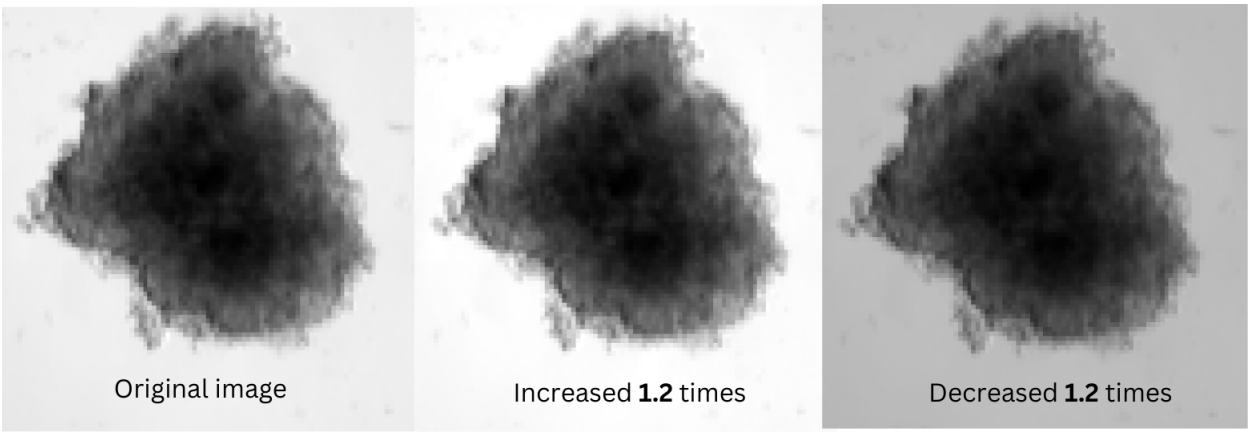

Figure 7.7: contrast increased or decreased

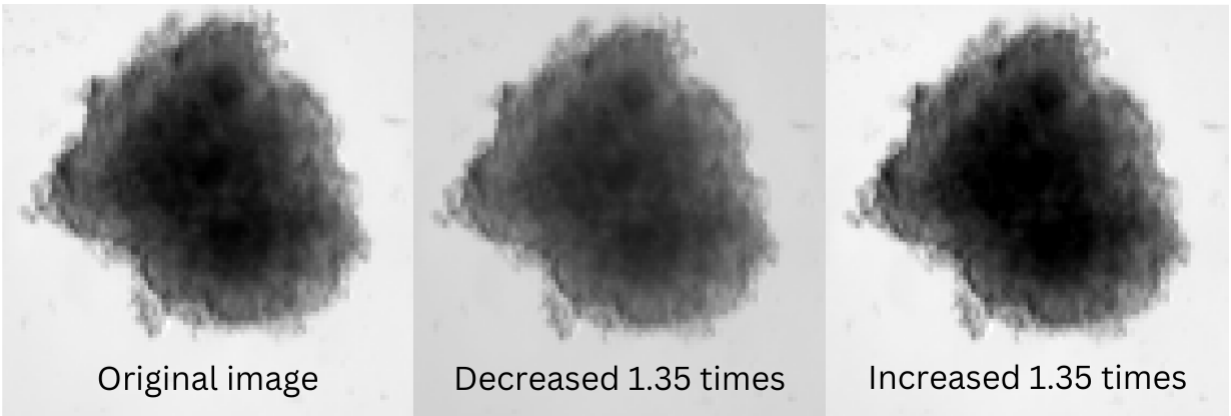

Figure 7.8: Brightness increased or decreased

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, round small cancer cell surronding lot of

debris around it as we seen in figure 1.4. 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 like in figure 7.9. 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 or certain debris patterns of explod images. 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.

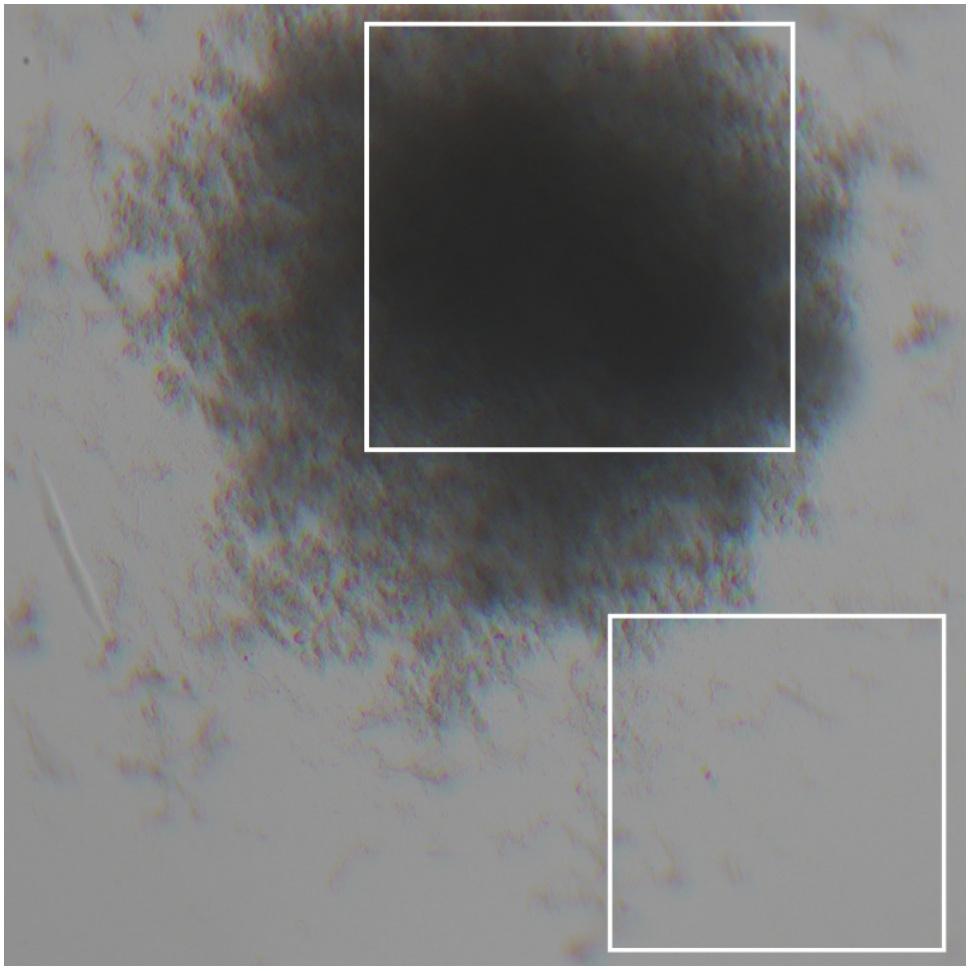

Figure 7.9: 5 Data augmentations we explored

I named this augmentation as 'Resize', which illustrated in 7.11 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 data aug from original image can be one with increased contrast which makes darker and one with less contrast which makes it gray as in the fugure 7.10) which is infact not good for our downstream task. because one of our goal is to differentiate/have different feature representation for untreated images (gray color) and single dose (darker color).

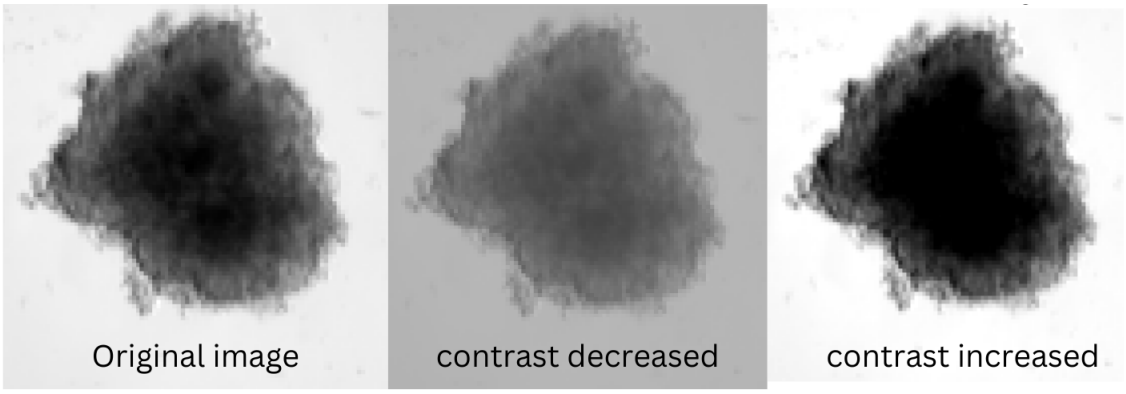

Figure 7.10: contrast increased or decreased

Hence I removed contrast from 'Resize' and 'Sweet' and named them as 'Resize ohne contrast' and 'Sweet ohne contrast' correspondingly. which illustrated in 7.11 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

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

5 Different data augmentation pipe lines explained above illlustrated as nutshell in 7.11

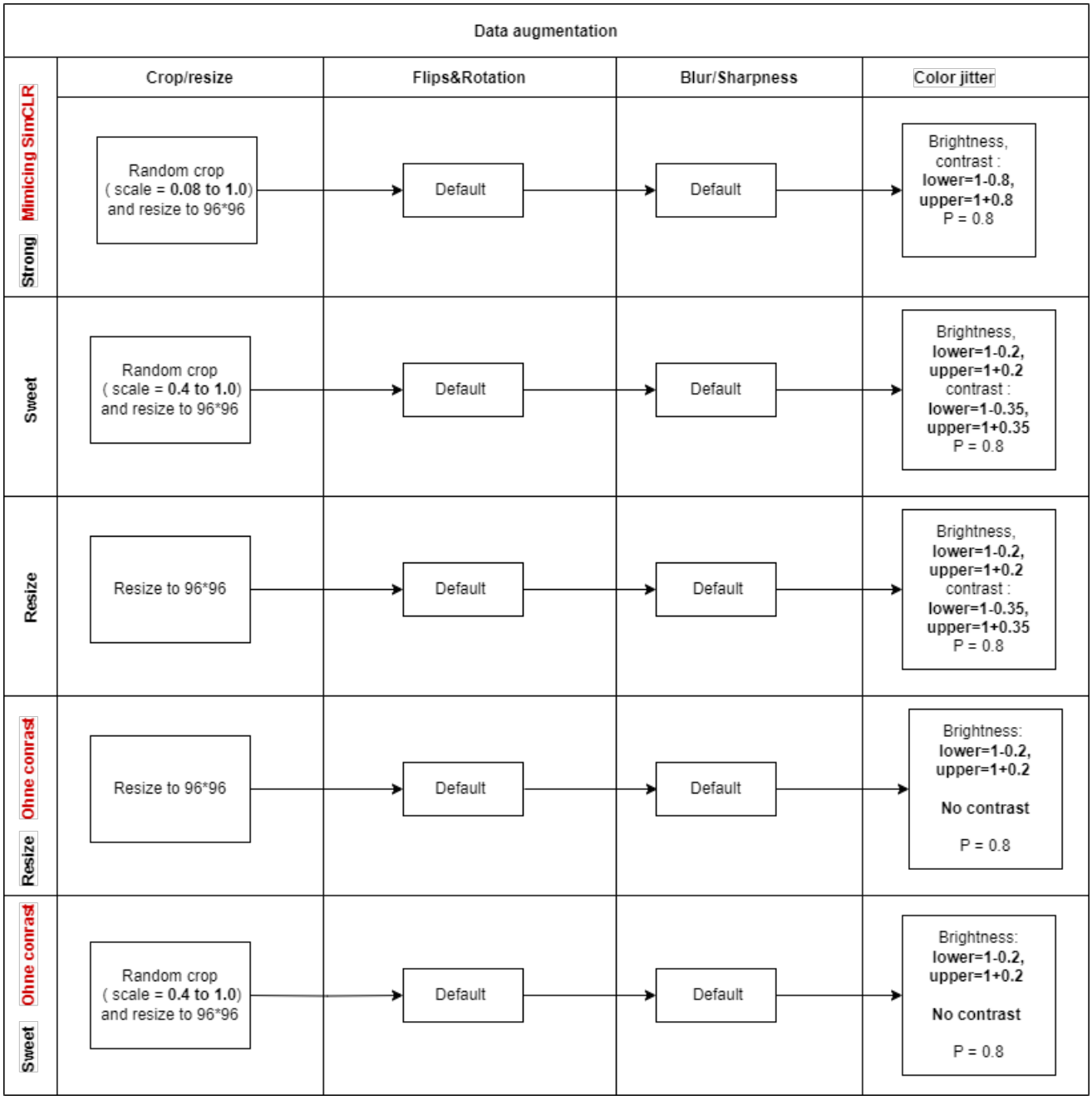

Figure 7.11: 5 Data augmentations we explored

7.3 Train SimCLR as SSL model

As explained in chapter 7, SimCLR was used as the model for self-supervised learning (SSL).

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:

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

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.

Intituivly given a set of $\{z_k\}$ with the batch size 2N, that includes a positive pair of examples z_i and z_j, the loss function aims to identify z_j in $\{z_k\}_{k\neq i}$ for a given z_i.

Above loss function we can reformualte as:

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

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

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

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 gives us:

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

Equation 7.2 is implemented as the loss function in our experiments.

The SimCLR framework was originally implemented in TensorFlow by the authors. In this work, I adopted the PyTorch implementation of SimCLR and utilized the parameters provided in the PyTorch SimCLR tutorial add reference, as they demonstrated optimal performance for the STL10 dataset. The STL10 dataset is still natural image dataset, with an image resolution of 96×96, matches the dimensionality of the our images to feeded into the model. I used the following parameters for training from them: a learning rate of 5×10^{-4}, a temperature of 0.07, a weight decay of 1×10^{-4}, a cosine learning rate schedule with a minimum learning rate of $\frac{\text{lr}}{50}$ and T_{max} set to the maximum number of epochs where T_{max} refers to the maximum number of iterations or epochs for which the cosine learning rate schedule is defined. It determines the period of the cosine function used to anneal the learning rate, the ADAM optimizer, and 4 hidden layers with a hidden dimension of 128 in the projection head. Additionally I used max

epoch as 245 and Batch size as 64. I didn't finetuned specifically for downstream ranking task because we don't know the ground truth label. I could have been fine tune for the intermediate evaluations, but my focus was on completing the pipeline.

The following explains how we process a 3-channel image as if it were a standard RGB image and apply augmentations to generate two augmented versions.

Input to model (train loader dimension) :

- aug1: torch.Size([64, 3, 96, 96]) (batch size, no of channels, H, W)
- aug2: torch.Size([64, 3, 96, 96]) (batch size, no of channels, H, W)

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

- torch.Size([64, 512, 1, 1]) (Batch size, standard resnet18 output dimension after avg pooling, H,W)

Model output after projection head:

- torch.Size([64, 20]) (Batch size, no of values in feature vector)

Both output feature before and after projection head will be used for further downstream task to check the performance.

7.4 Evaluation of SimCLR training

Top-1 Accuracy:

Top-1 accuracy measures how often the loss function correctly identifies z_j in $\{z_k\}_{k\neq i}$ for a given z_i. Specifically, it evaluates how often the model ranks z_j and z_i as having the highest cosine similarity in a given set $\{z_k\}$, where the batch size is $2N$ while the set includes a positive pair of examples z_i and z_j. If the positive pair z_j and z_i is ranked first (i.e., has the highest cosine

similarity), it is counted as correct. Top-1 accuracy is calculated as the mean of these correct counts over all samples in the batch.

Purpose: Indicates how often the model correctly identifies the positive example as the most similar, reflecting the model's precision at a fine-grained level.

Top-5 Accuracy:

Top-5 accuracy measures how often the loss function identifies z_j in $\{z_k\}_{k\neq i}$ for a given z_i within the top 5 ranked pairs. Specifically, it evaluates whether z_j is ranked among the 5 highest cosine similarity scores in $\{z_k\}$, where the batch size is $2N$, and the set includes a positive pair z_i and z_j. If the positive pair z_j and z_i is ranked within the top 5, it is counted as correct. Top-5 accuracy is the mean of these correct counts over all samples in the batch.

Purpose: Evaluates the model's ability to identify the true positive within a broader range of candidates, providing a softer measure of precision compared to Top-1 accuracy.

Mean Position: For each pair, the ranking is calculated based on its cosine similarity relative to all other z_k in $\{z_k\}_{k\neq i}$. Calculate the ranking position of the positive pair z_j and z_i across all sample pairs in the batch. The mean position is the average of these ranking positions.

Purpose: Provides a quantitative metric for how far down the ranked list the positive pairs typically appears, offering insight into the quality of the model's ranking performance.

While training and validation images are applied by data augmentation pipelines such as 'strong' , 'sweet', 'resize', 'sweet no contrast', 'resize no contrast' inference images are only resized to 96*96 before the training as data augmentation. that means while calculating the ranking position or top 1 accuracy or top 5 accurcay or loss, model do inference on pairs of images which are always full image without any cropping.

If we compare the strong aug vs others we can see loss or top accuracies or mean position for training and validation dataset are better for others than strong. This is because 'strong' have to learn harder than other augmentations since 2 different augs of same image in strong are completely different. For example one is highly increased contrasted and other is highly decreased contrast. or another example: one have very small portion of background other have

small portion of cancer which is just pure black. which doesn't have any certain pattern or feature relation of same image. These pure small portion of black can be also be present in the batch as from another image as aug so it have to learn to map high cosine sim for these two different augs. Specifically Top one accuracy struugles to get close to 100 comparing to others.

Another interesting thing to notice is: top accuracy and mean position for inference dataset. From the beginning of the epoch inference top accuracies or mean position are the close to 100 or 1 respectively. This is because since inference image augs are just resized and not applied any kind of brightness or contrast or blur/sharp or flip/rotation, its just original images but with reduced size 96*96. so while ranking or calculating the similarity between pairs model sees the full picture to determine the similarity. and as we are gonna seen in section **??** if its full image without any additional data augs other than reduced to size 96*96 cosine similarity between augs of same image is higher than any other data aug pipelines which includes crop(strong, sweet etc). in another words for model its so easy. this leads to the question whether model learn some effective features for downstream tasks.

Thats also refelects why 'Resize' and 'resize no contrast' data aug pipelines have better performance in terms of training and val loss or top accuracies or mean position than data aug pipeline which includes cropping.

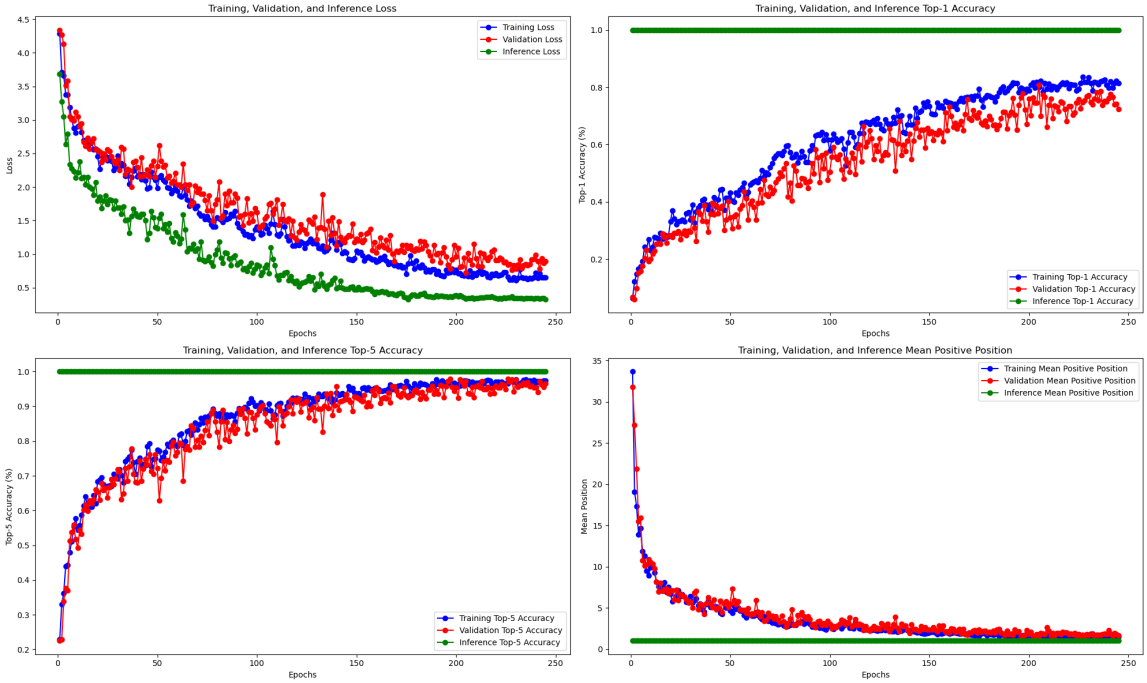

Figure 7.12: strong

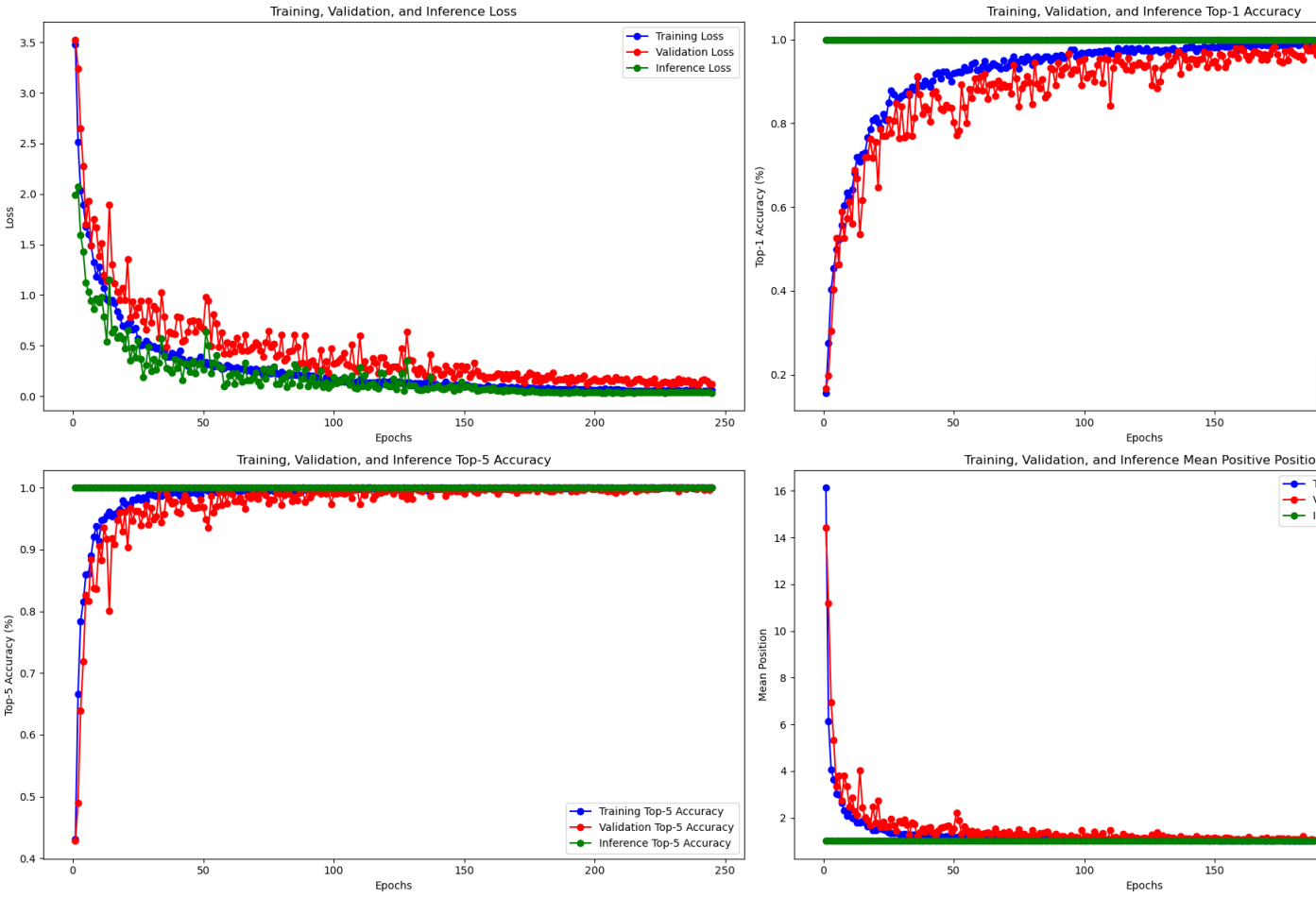

Figure 7.13: sweet

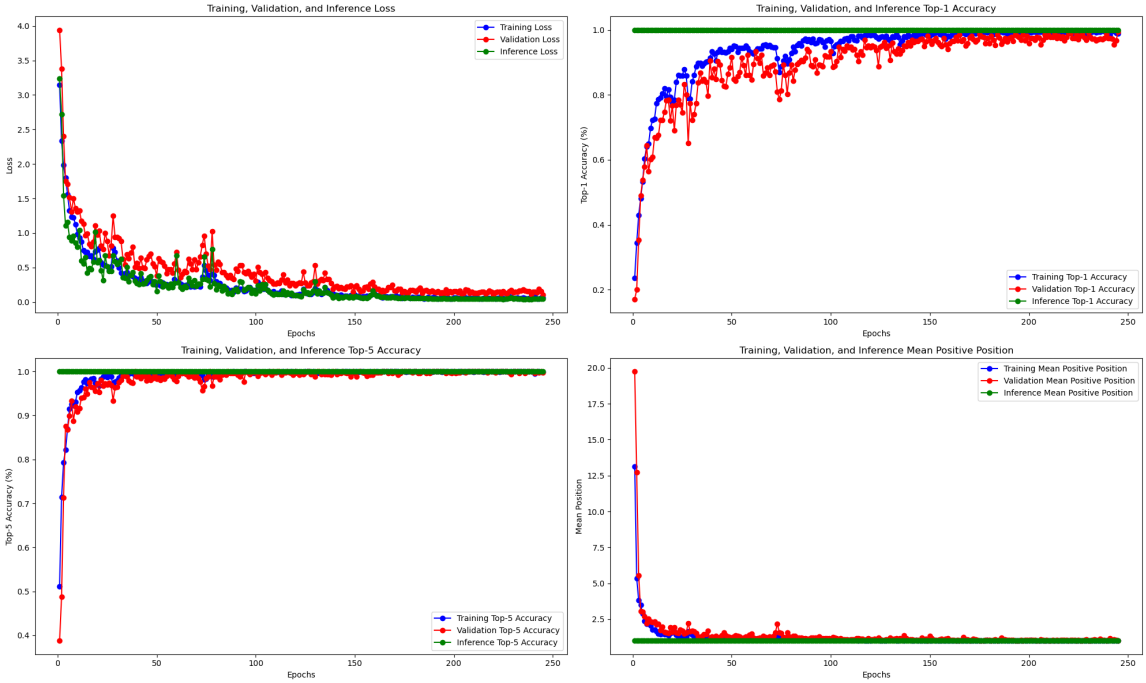

Figure 7.14: sweet no contra

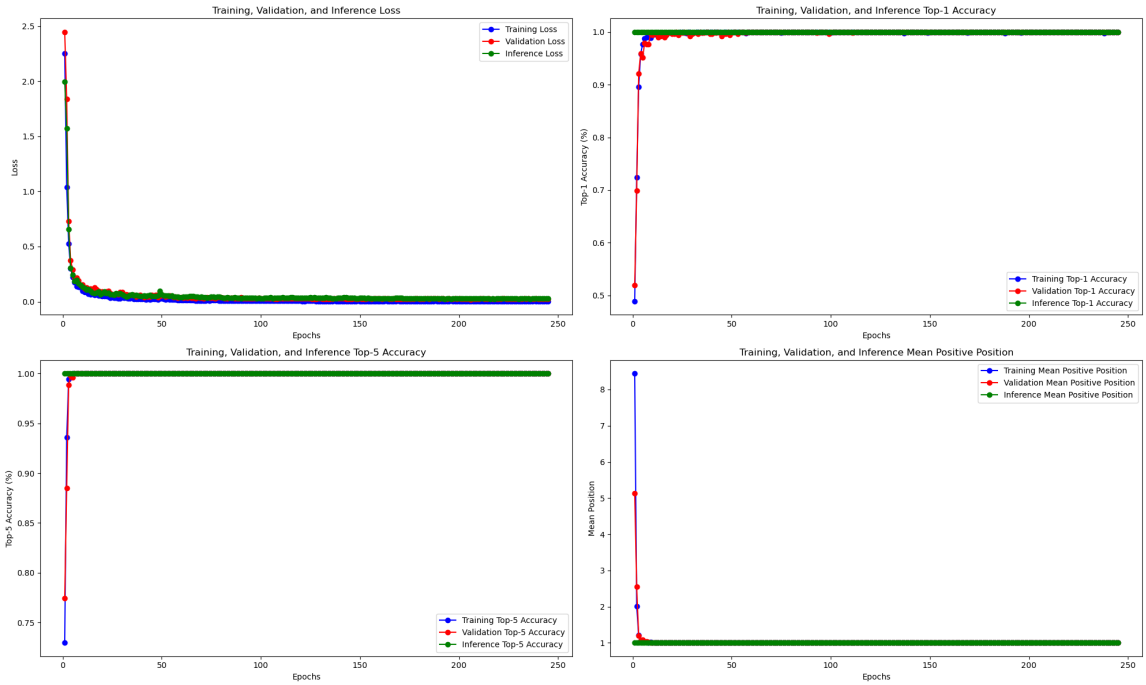

Figure 7.15: Resize no contra

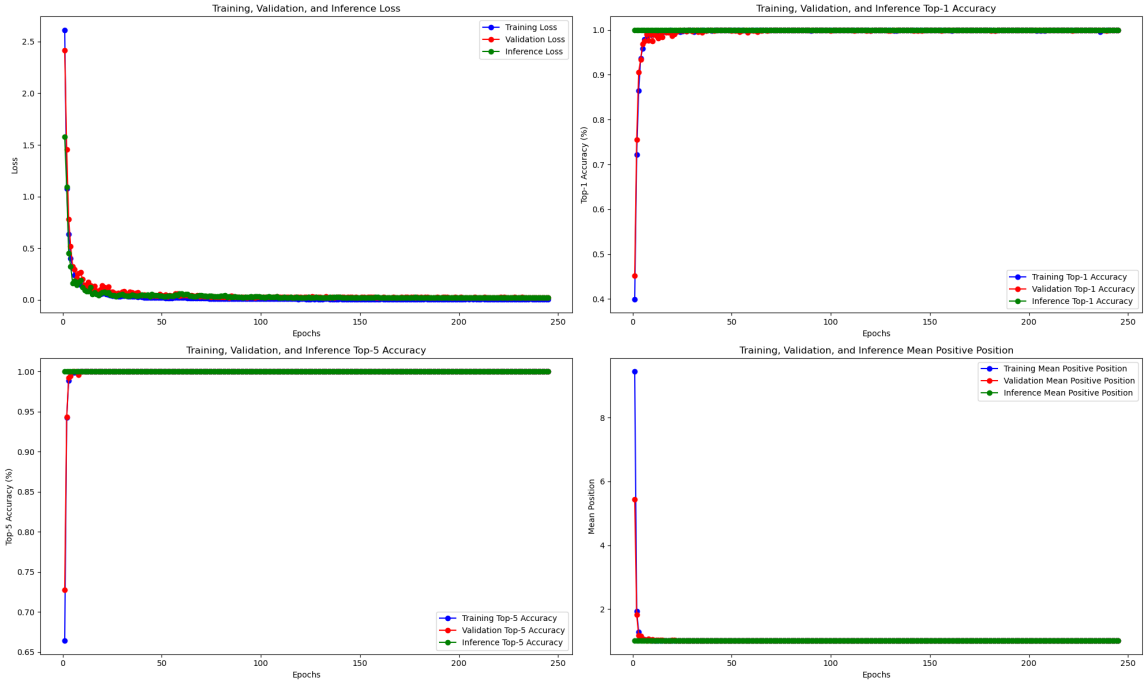

Figure 7.16: Resize

add tsne i prefer pca plot for the cluster vs original vs befroe and after

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 and clusetring to check if simclr atleast learned to differentiate between untreated , single dose and exploded images.

Distance measurement approach is used to whats the performance of simclr feats derived from 5 different aug pipelines in terms of cosine distance between 2 augs derived from same image.

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.

The logistic regression model is implemented as a single linear layer, where the input dimension corresponds to the feature dimension of feature spits out from simclr, and the output dimension corresponds to the number of classes. Specifically, the model uses a feature dimension of 512 if the feature is before projection head or 20 if the feature is after projection head and outputs predictions for 3 classes. The mathematical representation of the model is as follows:

$$\hat{y} = xW^T + b$$

where:

- $x \in \mathbb{R}^{N \times d}$ is the input feature vector, where $d = 512$ if the feature is extracted before the projection head, or $d = 20$ if the feature is extracted after the projection head. - $W \in \mathbb{R}^{3 \times d}$ represents the learnable weights. - $b \in \mathbb{R}^3$ is the bias term, - $\hat{y} \in \mathbb{R}^{N \times 3}$ are the logits representing the unnormalized class scores for N samples.

PyTorch's CrossEntropyLoss combines the softmax operation with the log loss computation for numerical stability, which is why we pass the logits (unnormalized outputs) directly to the loss function.

This model is trained to minimize the cross-entropy loss for multi-class classification. 3 classes trained for 250 epochs. The dataset was divided into a training set (80%) and a validation set (20%). Batch size = 8. loss function = cross entropy loss. a learning rate of 5×10^{-4}.

The learning rate scheduler used is the `MultiStepLR`, which reduces the learning rate at specific milestones during training. In this case, the learning rate is reduced by a factor of 0.1 at the epochs corresponding to 60% and 80% of the total training epochs. These milestones are

defined as:

$$\text{milestones} = [\text{int}(T_{\text{max}} \times 0.6), \text{int}(T_{\text{max}} \times 0.8)]$$

where T_{max} represents the total number of training epochs. The `gamma` parameter specifies the factor by which the learning rate is multiplied at each milestone, which in this case is 0.1.

Table 8.1: Dataset Summary from Drug Screening Experiments

Dataset	**Total Number of Images**
Untreated Dataset	472
Single Dose Dataset	103
Exploded Dataset from Drug Screening Experiments	40

I didn't add any other comibnation of drug screening experiments since we are not sure whether they belong to which group. Some of them have medium resemblence of the above 3 classes but we can't be sure.

8.1.1 Comparison of Classification Accuracy and Epochs for Different Data Augmentations

As explained in Data augmentation section we use 'strong', 'sweet', 'resize', 'sweet no contrast', 'resize no contrast' simclr features to compare their performance inbetween and also against original images which one classify these 3 classes better.

We calculate the train accuracy and the test accuracy for each data aug pipelines as well as for original images.

Train epoch in the table: The number of epochs required to reach the best training accuracy. Test epoch in the table: The number of epochs required to reach the best test accuracy.

how do we turn raw images to orig images to feats To use original images directly first we resized to 96*96 since we did that for simclr so that it could be fair comparison. Then we normalised each by dividing it by 65535 (16 bit). Each image flatten into vectors of ([1,96*96*3]where 96 =H=W and 3 = no of channels) suitable for logistic regression model. and we use same parameters that we used for simclr feats classification lke learning rate , batch size etc for fair

comparison.

Table 8.1.1 - are you saying that you classification accuracy is 100? That seems far too good to be true. Why do the train runs have different number of epochs? And what do you mean by ”Test Epoch”?!

general comment to tables with result - after giving the tables you need to write a text which helps us to interpret it. What are the most important number we shall look at? What shall we take out from the table as a message. For example something like (please do not use this directly. I have no clue if this is the message you want us to take away from the table) ”In Table 8.1. we show that when using the features Before Projection Head all augmentation methods reached 100 train and test accuracy. The Resize approach needed the fewest epochs.”

Augmentation Type	Metric	Strong	Sweet	Resize	Resize No Contrast	Sweet No Contrast
After Projection Head	Train Accuracy (%)	66.67	47.76	89.63	63.41	54.67
	Train Epoch	246	249	250	250	245
	Test Accuracy (%)	68.29	55.28	90.24	63.41	56.10
	Validation Epoch	228	246	223	202	244
Before Projection Head	Train Accuracy (%)	100	100	100	100	100
	Train Epoch	190	3	13	12	21
	Validation Accuracy (%)	100	100	100	100	100
	Validation Epoch	1	1	1	2	2

Table 8.2: Performance metrics for different augmentation strategies before and after the projection head.

Table 8.3: Original Image Results

Metric	Train Accuracy (%)	Train Epoch	Validation Accuracy (%)	Validation Epoch
Original Image	99.39	249	99.19	17

Train loss vs Val loss . Train accurcay vs val accuracy

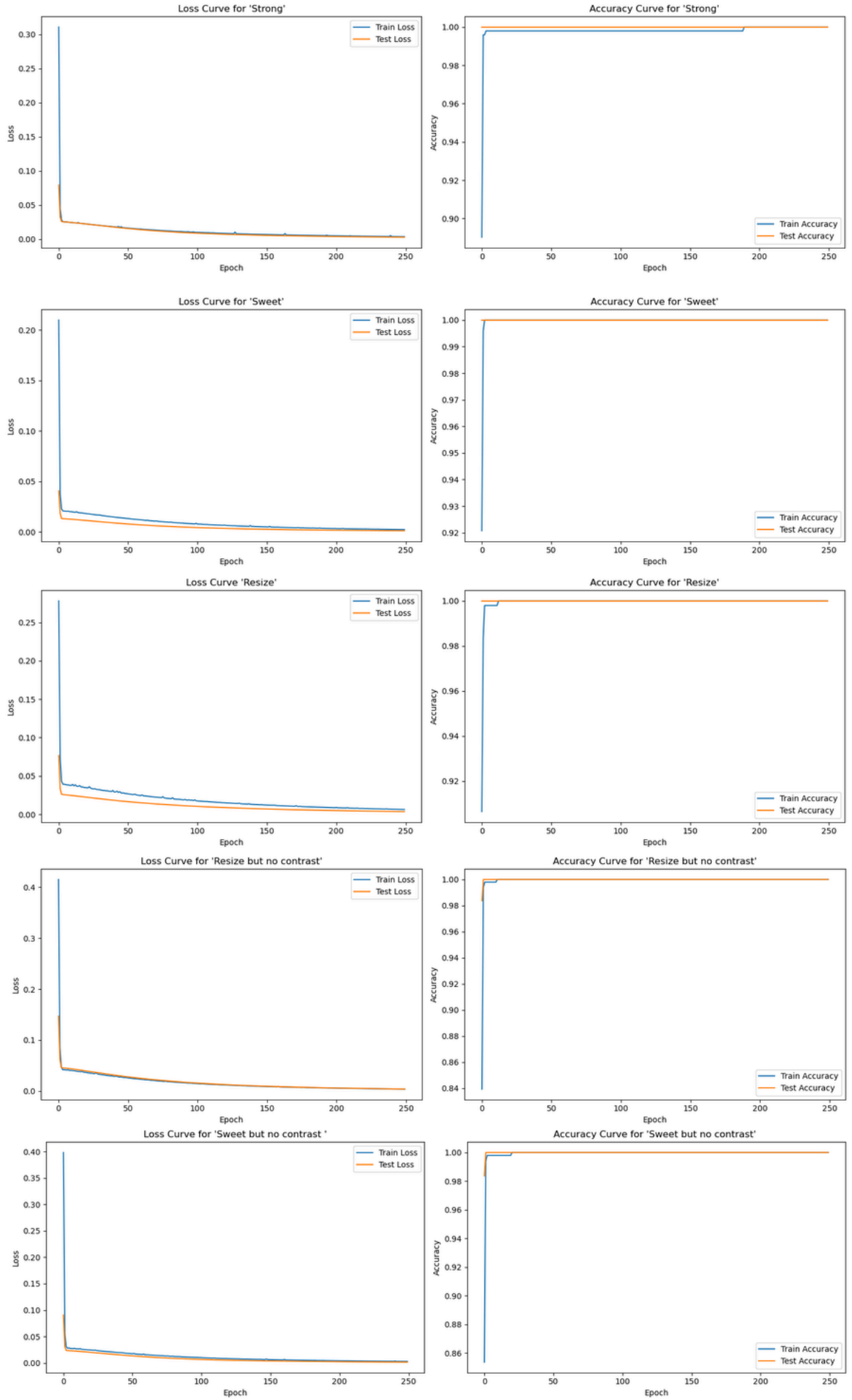

Figure 8.1: before

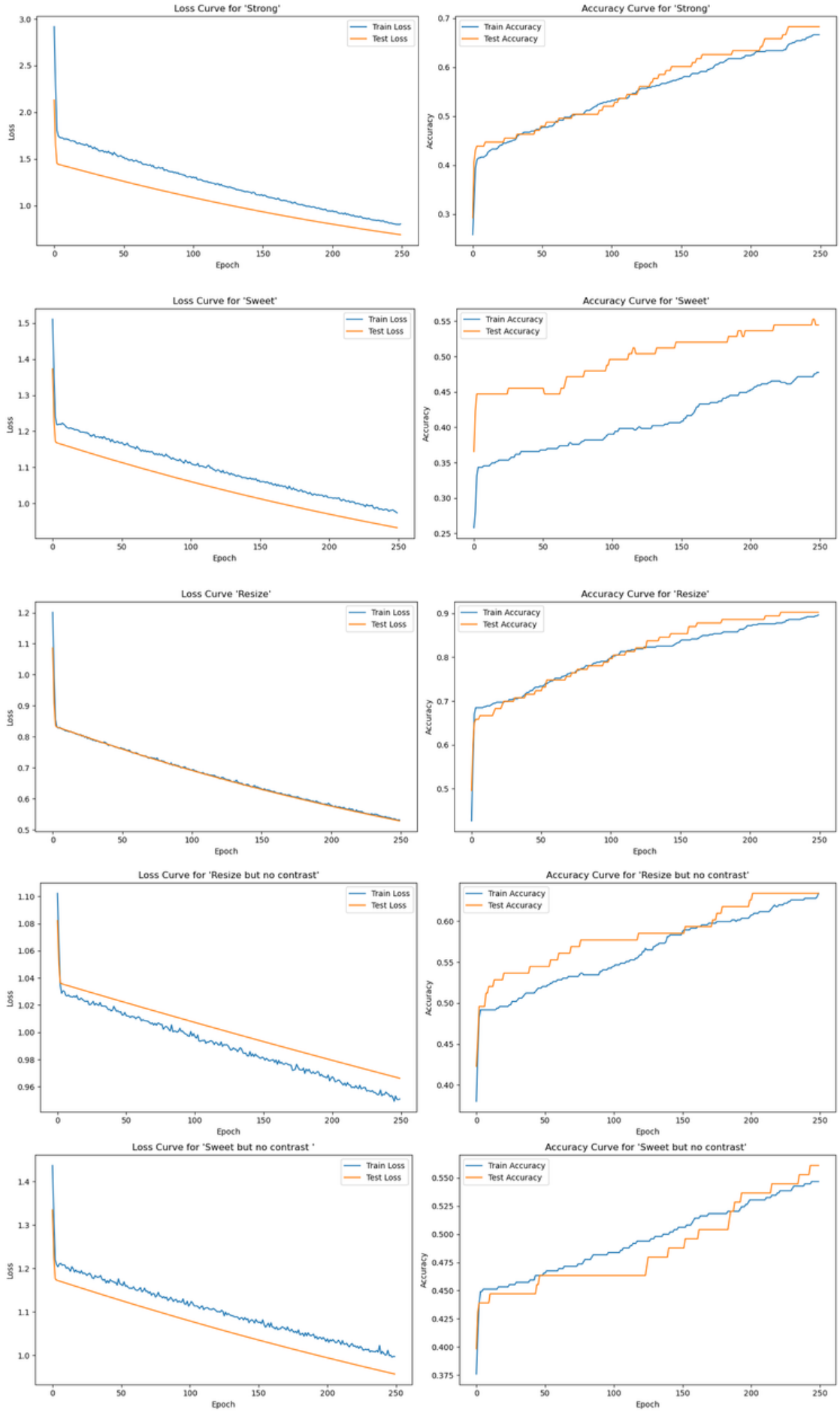

Figure 8.2: after

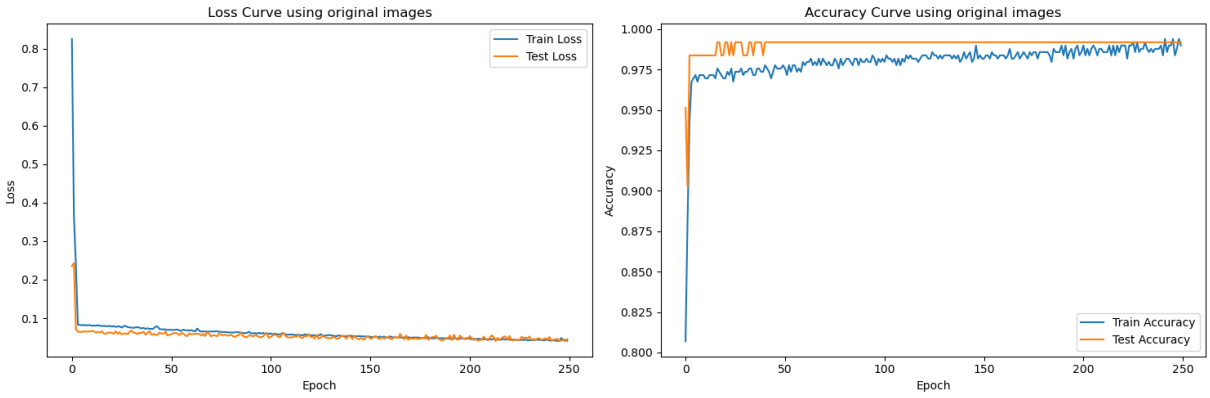

Figure 8.3: orig

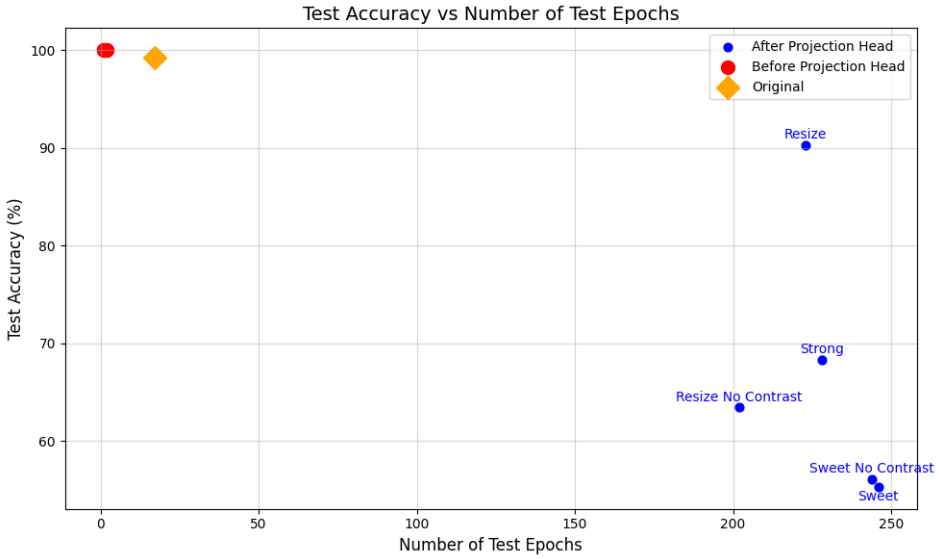

Figure 8.4: before

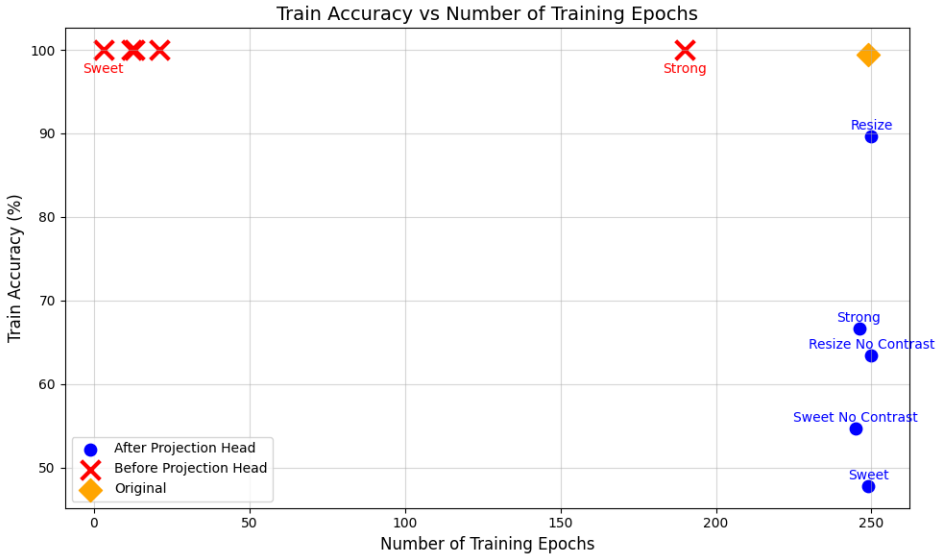

Figure 8.5: before

8.1.2 Inference

I choose my self 22 images that have close resemblence in my eyes to single dose images from drug screen for inference. (I repeat these are not biology expert labeled drug screened image. so any inference result is irrelavant in the sense to actual ground truth)

For doing inferecne I choosed before head projection features since thats the one gave 100 perentage accuracy while train and validation.

Type	Strong	Sweet	Resize	Resize no Contrast	Sweet no Contrast
Before Projection Head	90.91	100.00	100.00	100.00	95.45
Original Images				68.18	

Table 8.4: Performance Metrics Before and After the Projection Head

From the above table we can see that during the inference except 'strong' and 'sweet no contrast' all other augs still able to reach 100 percentage accuracy to classify the ds close to sd images to sd based on visual resemblence. Using raw Original images features it got 68.18 percentage accuracy while Simclr before projectoin head could classify atleast 90 percentage.

table 8.2 can you somehow bring this into table 8.1 e.g. as another column? Would be easier to read. The first column "Augmentation Type" could be made smaller if you wrap the text so that it is on multiple lines.

In Table 8.1. we show that when using the features Before Projection Head, all augmentation methods reached 100 percent train and test accuracy. It is clear that the features extracted before the projection head perform far better than those extracted after the projection head, as seen in the original similar paper. Also, the number of epochs required to reach the best accuracy is lower for the features extracted before the projection head than those extracted after the projection head.

While original images also classify with 99 percent accuracy, they perform better than the features we use after projection. Also, their almost indistinguishable performance from the features extracted before the projection head leads to the conclusion that we cannot make an informed decision based on this classification task.

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.

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.

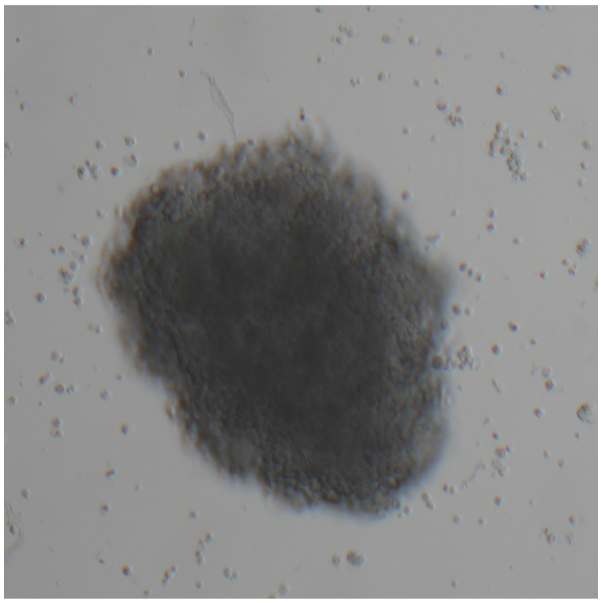

Figure 8.6: orig

Table 8.5: Summary of Datasets

Dataset	Control (C)	Single Dose (SD)	Exploded (E)	SD close for inference
Full Dataset (Unbalanced)	472	103	40	all 22
40 Subset (Balanced)	(20 with debris + 20 without debris)	40	40	20
Curated Full Dataset	280	103	40	all 22
Curated 40 Subset (Balanced)	40 (all without debris)	40	40	20

run 400 times for different random initilisation. Note: Drug screen images are visually similar to single dose treatment. All balanced datasets are normalized to the exploded group size (n=40). the curated full data set doesn't give any infromation ? if no remove the column

Table 8.6: 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
	Strong	**83.90**	**100**	**100**	**100**
	Sweet	**99.18**	**100**	**100**	**100**
Before	Resize	65.7	97.5	97.87	100
	Resize No Contrast	69.92	90	86.76	100
	Sweet No Contrast	**98.21**	**99.17**	**99.52**	**99.17**
	Strong	72.68	95.83	74.00	100
	Sweet	47.32	74.17	55.08	78.33
After	Resize	60.32	92.50	74.23	100
	Resize No Contrast	44.72	64.17	47.28	66.67
	Sweet No Contrast	52.20	86.67	58.16	89.17

cosine distance:

The results show a clear trend of superior clustering performance before the projection head compared to after the projection head.

Before:

data augmentatoin ‘sweet’ performs the best over other data augmentations and achieved consistant accuracy over all different data set types.

since sweet performed 100 percentage accurcay across cured and uncured dataset its evidant that the problem for having less accracy is nothing to do with exploeded look like images in control instead its weight unbalance class problem.

comparing ’sweet’ and ’sweet no contrast’ its evident that without contrast slightly hurts the performance.

comparing ’sweet’ and ’strong’ they both achieved 100 percenatage accuracy ie (same performance)over all different data set types except the full dataset. for full dataset sweet performed far better than strong , sweeet even achieved 99.18 where strong reached only 83.90.

comparing all data augmentations ’resize’ and ’reesize no contrast performed the least giving the insight that cropping is essential to get better cluster performance using cosine distance as distance metric.

as a breif conclusion its pretty clear that cropping 0.35 to 1 percentage of whole image is crucial especially cropping have huge effect for cluster performance using cosine distance as distacne metric.

One thing that need to keep in mind that all above explanations or achievements by different data augs are corresponds to clustering performance doesn’t mean that it could work for rankin strategies too.

Table 8.7: 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	**88.45**	**100**	**100**	**100**
	Sweet	**98.21**	**91.67**	**99.29**	**95.83**
	Resize	70.40	78.33	66.66	88.33
	Resize No Contrast	68.62	90.00	86.28	100
	Sweet No Contrast	**99.02**	**97.50**	**99.76**	**99.17**
After	Strong	**75.45**	**100**	**99.53**	**100**
	Sweet	51.54	74.17	60.52	79.17
	Resize	43.25	83.33	53.90	87.5
	Resize No Contrast	47.8	63.33	47.04	65.83
	Sweet No Contrast	51.87	85.0	57.68	88.33

Euclidean distance: Just like cosine distance, with eulcidean distacne as distacne metric also established a superior performance trend for before the projection head compared to after the projection head. Just like cosine distance, inside the before projection head, data augs with cropping have far better performance than without cropping dataaugs such as resize and resize no contrast.

Eventhough 'strong' achieved 100 percentage accuracy across all dataset types except the full dataset unbalanced. when it comes to full dataset, sweet and sweet nocontrast pperforms better than all others. and among all data augs we can see sweet no contrast' maintain the cosistancey in performance across all dataset types with more than 97 percentage accuracy and also perfromed better than just 'sweet' across all dataset.

interesting thing to notice here is the after projection head 'strong' it achieved higher clusterig performance 99 and above across all dataset types except the full dataset. which is a good indication that we should also consider the after projection head features for ranking task.

Table 8.8: Evaluation Results Using Different Distance Metrics for original images

	Metric	Full Dataset (Unbalanced)	40 Subset (Balanced)	Curated Full Dataset (Unbalanced)	Curated 40 Subset (Balanced)
Original Images	Cosine Distance	62.76	72.5	58.15	77..50
	Euclidean Distance	55.28	76.67	53.66	77.50

Original images: comparing original images feats to simclr feats we can see that entire data augs for before head feats outperformed entirly for both cosine distacne and euclidean distacne. while comparing original image to after projection head the performance is comparable except for the strong data aug where after head still have better perfromance.

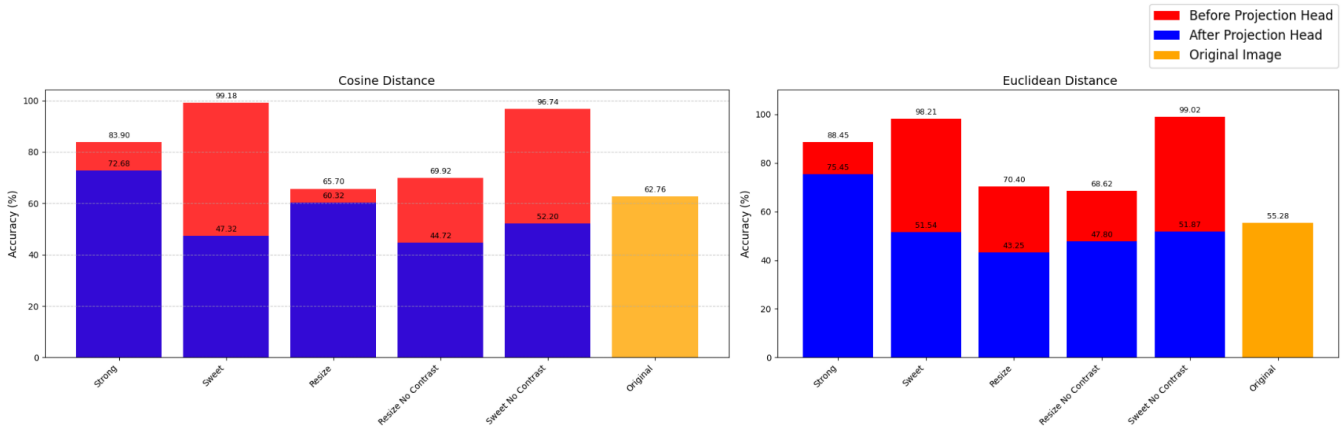

Figure 8.7: orig

Include PCA 1 here? maybe in ranking. but we can add PCA 2 fugures. remove inference data from table.

do sd VS others all? yes depend on time

Inference:

Since before projection head for cosine distance gives far better performance, i do inference on them. I choose 22 images that have close resemblence in my eyes to single dose images from drug screen for inference. We name those images as 'sd close'(I repeat these are not biology expert labeled drug screened image. so any inference result is irrelavant in the sense to actual ground truth)

we used all 22 sd close images for full dataset along with the 3 classes that describe in the table. But for balanced datasets except control and exploded we changed the dataset setup described in the table as follows for keeping the dataset balancing: 20 sd and 20 sd close images. so that it will stil be 40 as other classes.

so during 3 class clustering 'sweet' (99.18) perfromed better than sweet no contrast (98.21) for all full dataset unbalanced. and in others sweet achieved 100 percentage accuracy whihc makes clear that for 3 class problem sweet have higher perfromacne comparing to other data augs. but during inference sweet accuracy reduced to 95.76 with -3.42 change in reduction while sweet no contrast reduced to 97.33 with -0.88 reduction change. and sweet no contrast kept the highest accuracy during inference. figure 8.8 shows that with sweet no contrast data

aug it classified all 103 sd images correctly and during inference it clustered all sd close image images in the cluster of sd images.

Table 8.9: Evaluation Results on COSINE

Type	Augmentation	Full Dataset (Unbalanced)	Uncured Balanced	Curated Full Dataset	Curated Balanced
	Strong	85.09 (+1.19)	100 (0)	98.43 (-1.57)	100 (0)
	Sweet	95.76 (-3.42)	100 (0)	100 (0)	100 (0)
Before Projection Head	Resize	67.03 (+1.33)	97.50 (0)	98.42 (+0.55)	100 (0)
	Resize No Contrast	71.89 (+1.97)	90.83 (+0.83)	87.42 (+0.66)	100 (0)
	Sweet No Contrast	**97.33 (-0.88)**	**98.33 (-0.84)**	**99.55 (+0.03)**	**99.17 (0)**

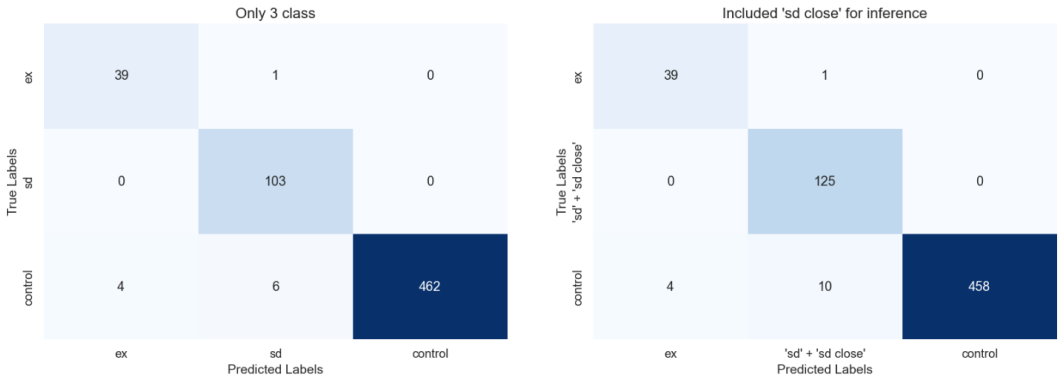

Figure 8.8: orig

How do you know in inference dsclose was the one confused? maybe confusion matrix? sweet no contrast before cosine

Table 8.10: Evaluation Results on Euclidean

Type	Augmentation	Full Dataset (Unbalanced)	Uncured Balanced	Curated Full Dataset	Curated Balanced
	Strong	84.61 (-3.84)	100 (0)	99.10 (-0.9)	100 (0)
	Sweet	98.27 (+0.06)	91.67(0)	99.32(+0.03)	95.83 (0)
Before Projection Head	Resize	72.06 (+1.66)	79.17 (+0.84)	68.54 (+1.88)	86.67(-1.66)
	Resize No Contrast	71.11 (+2.49)	100 (+10)	87.42 (1.14)	100 (0)
	Sweet No Contrast	**99.06 (+.04)**	**97.50 (0)**	**99.78 (+.02)**	**99.17 (0)**
After Projection Head	Strong	75.51 (+0.06)	92.5 (-7.5)	72.58 (-26.95)	95 (-5)

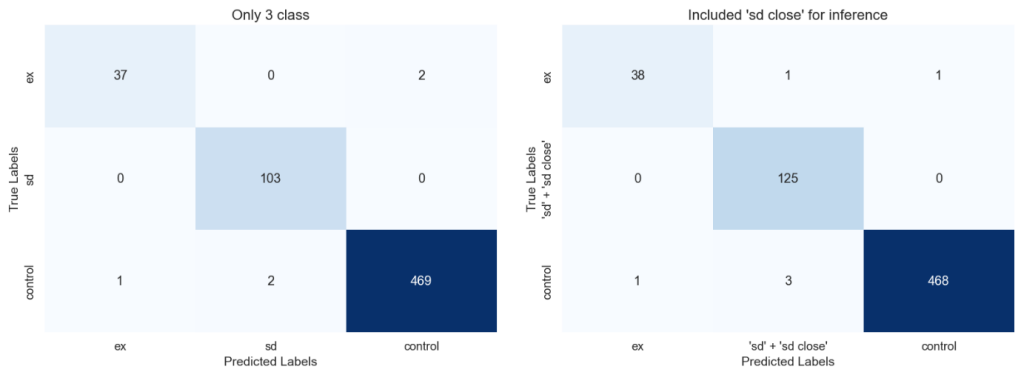

Figure 8.9: orig

For euclidean distance, during 3 class 'sweet no contrast' have he highest performance with 99.02 accuracy without any misclassification for sd images as we seen in figure 8.9. and during inference it clustered all sd close image and sd images together in the cluster of sd images corrrectly as we seen in figure 8.9.

From this we can conclude that 'sweet no contrast' data aug before projection head is the best data aug for clustering task for both cosine distance and euclidean distance as distance metric. which basically means that sweet cropping of 0.35 to 1 (less intense cropping is essential for clustering performance.)

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 Day7 to day10 predcition using CAE

mention problem of data lack here instead in research questions? Reason why control doesn't work as anomaly normal because when it turns to day 10 it contains darker + gray so it learns to predict both dark and gray. but when it comes to single dose as anomaly normal it learns only contains dark in day 10 which have no similarity to control day10 or explod day10, and also when it comes to explod in day 10 it have explod thats not in both / no similarity in day 10 of single dose and control

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 using convolutional autoencoder. (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. 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.

4. **Step 4:** so in the above methods we train first solely on untreated /control images then we did inference on all images just like classical anomaly detection approach. with that same idea/concept, but now we considered other gps as normal and will try to find the deviation from that transition. that is we train solyliy on day 7 untreated to day 10 sd images then we do the inference on all images so that inference loss will be how much it deviated from the single dose images. repeating the same concept, we train solely on day 7 untreated to day 10 ex images then we do the inference on all images so that inference loss will be how much it deviated from the exploded

5. Perform the above steps on original image features instead of simclr feature vectors for comparitive study.

Lack of dataset problem for day 7 to day 10 images: Due to environmetal factors sometimes we may get images of day 7 but we won't able to aquaire day 10 images or sometimes we may have day 10 images but not corresponding day 7 images.

Hence for the images for prediction from day 7 to day 10, we only have 130 control images of pair transitions. 29 images of pair for single dose and 40 image pairs for explod.

original images

data preperation: I decided to train with day 7 untreated to day 10 untretaed images. normalised the images by dividing them by 65535. reduced the size to 96*96. Added data augmentations such as horizontal/vertical flips and rotations and random brightness/sharpness/blur as explained in the sweet augmentation section except contrast and cropping. We can't change contrast as I explained in the augmentation chapter.and also i didn't use cropping because of the idea that inorder to learn whole change or transformation to day 10, it needs to see the whole day 7 image which is already explained in the data augmentation resize chapter. i made sure that 1. augmentation to be coupled that means the same data augmenattion type parameter value used for both day 7 and corresponding day 10 images. 2. flips or rotation will be unique to each image with random brightness/sharpness/blur change. that means in the dataset original image can have different augs with the same parameter value of brightness/sharpness/blur since its random but it won't have the same geomatrical transformation.

The augmentation helps to increase the possible sample distribution that could happen in real life by that we are trying to find a solution for data defficiency and also helps to prevent overfitting. also it helps to be invariant of the brightness/position/blur/sharp changes that can happen due to the microscope mishandling.

training parameters and archiecture:

Convolutional Autoencoder Architecture: We developed an autoencoder architecture with an encoder-decoder structure, utilizing convolutional layers, batch normalization, dropout, and activation functions to learn feature representations and predict day 10 image from day 7. The number of feature maps is progressively increased and then symmetrically decreased to balance feature extraction and construction of day 10 same size image as day 7.

Encoder The encoder consists of three convolutional layers with a kernel size of 3×3 and `same` padding to maintain spatial resolution.

- Input channels begin with 3, progressively increasing to 16, 32, and 64 to extract deeper hierarchical features.

- Each convolutional layer is followed by `BatchNorm2d` for stable learning, `ReLU` activation for non-linearity, and dropout with probabilities of 0.2, 0.3, and 0.4, respectively, to mitigate overfitting.

- Downsampling is performed using `MaxPool2d` with a kernel size of 2×2, stride of 2, and padding to reduce spatial dimensions at each step.

Decoder The decoder mirrors the encoder, reconstructing the input from the learned features.

- It begins with 64 channels and symmetrically decreases the number of channels to 32, 16, and finally 3 to match the input dimensions.

- Each convolutional layer retains a kernel size of 3×3 with `same` padding, followed by `BatchNorm2d`, `ReLU` activation, and dropout with probabilities of 0.3 and 0.2, respectively.

- Upsampling is performed using `Upsample(scale_factor=2, mode='nearest')` to restore spatial dimensions step by step.

- The final layer includes a `Sigmoid` activation function to ensure output pixel values are in the range $[0, 1]$, making the model suitable for day 10 image construction.

The overall structure employs progressive channel expansion in the encoder ($3 \rightarrow 16 \rightarrow 32 \rightarrow 64$) to capture detailed feature hierarchies and symmetric channel reduction in the decoder ($64 \rightarrow 32 \rightarrow 16 \rightarrow 3$) for day 10 image construction. training parameters: bacth size = 32. learning rate = 0.001. optimizer = adam. loss = mse. epochs = 500.

Result: unfortunately, the model didn't learn the features of transition to day 10 well. the loss didn't decrease in the 500 epochs as we see in the figure. figure reflects that model struggles to predict the day 10 images from day 7 image, the predicted image is more resemblence visually to day 7 than day 10. For the model its hard to learn probably because in most cases the day 10 image cancer cell is totally changed to different shape or size or color pixel intensity. Also it

could be the position change due to the microscope handling position. (We used different flip and rotation augmentation but it maynot be effective as we think it is.) since the same problem we may have with the day 7 to day 10 single dose or explod images, I decided to move to simclr features to see if it can learn the transition better than the original images.

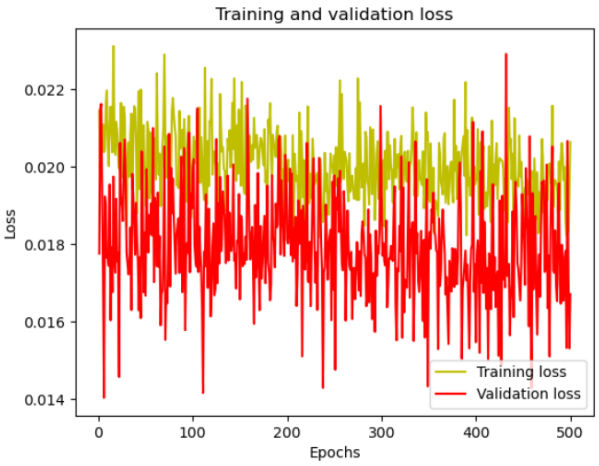

Figure 9.1: train vs test loss

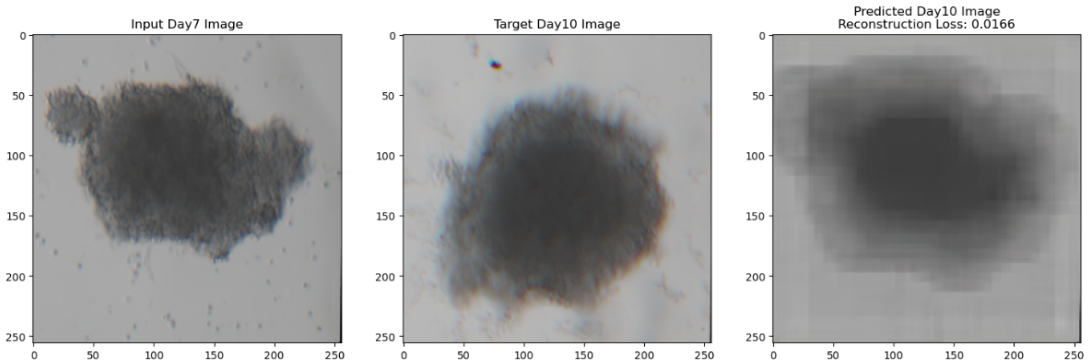

Figure 9.2: train vs test loss

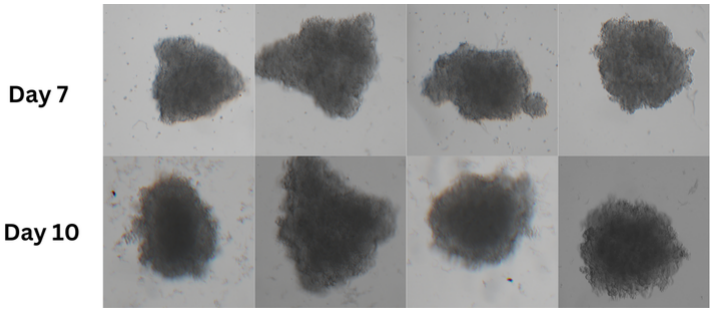

Figure 9.3: last row shows misposition of the cancer cell. in the day 7 it was above now. and in day 10 its below.

Simclr feats

Set up of data processing same as above original setup. frist I started with the pair of day 7 untreated to predict the day 10 untreated images setup.normalised images by dividing them by 65535. resized to 96*96. then I increased the dataset of all gps 130 of untreated. by using same data augmentation technics that I did to original image setup as explained in the above section but with made sure that flips or rotation will be unique to each image with random brightness/sharpness/blur change within a range as above. then I feed to simclr model to get features after and before projection head to train the CAE model.

Training setup: first to ensure the simclr features are scaled I did featurewise minmax scaling. Then I feed those into FeaturePredictor model which is a fully connected neural network. It consists of a series of 12 linear layers, where the input size reduces from 512 to 8 and then symmetrically increases back to 512 to match the original feature dimensions for before projection head features. For after projection head features,consists of a series of 6 linear layers, where the input size reduces from 20 to 4 and then symmetrically increases back to 20.

Each intermediate layer is followed by:

Batch normalization (BatchNorm1d): Normalizes activations to stabilize training and accelerate convergence. ReLU activation: Introduces non-linearity to model complex feature relationships. Dropout (Dropout): Applied with probabilities ranging from 0.2 to 0.4 to prevent overfitting, with deeper layers using higher dropout rates for regularization. The final layer is a linear transformation to ensure the output feature size matches the day 10 feature vectore dimensions which is 512 for before and 20 for after.

batch size = 32. loss = mse loss. optimizer = adam optimizer. learning rate = 0.0001. no of epochs = 2000. used cross validation with 3 fold and early stopp with the parameters patience = 500 and delta = 0.0001. training data and validation data divided as 80: 20 ratio.

As we see in the figure below, for the training of day 7 to day 10 untreated images and single dose, the train loss and validation loss decreased gradually in the 2000 epochs. But for the category of cosnidering exploded as normal ie training day 7 to day 10 exploed images we can see overfitting behaviour in the loss. This is probably due to the fact that for training day 7 to

day 10 untreated images we have 130 images correspondingly but for day 7 to day 10 exploded images we have only 40 images as original show sd only have 29 original images also shows overfitting. This is the problem of data deficiency. We have tried to mitigate this problem by increasing the dataset size by data augmentation but neverthless for the exploded images training didn't quite worked out.

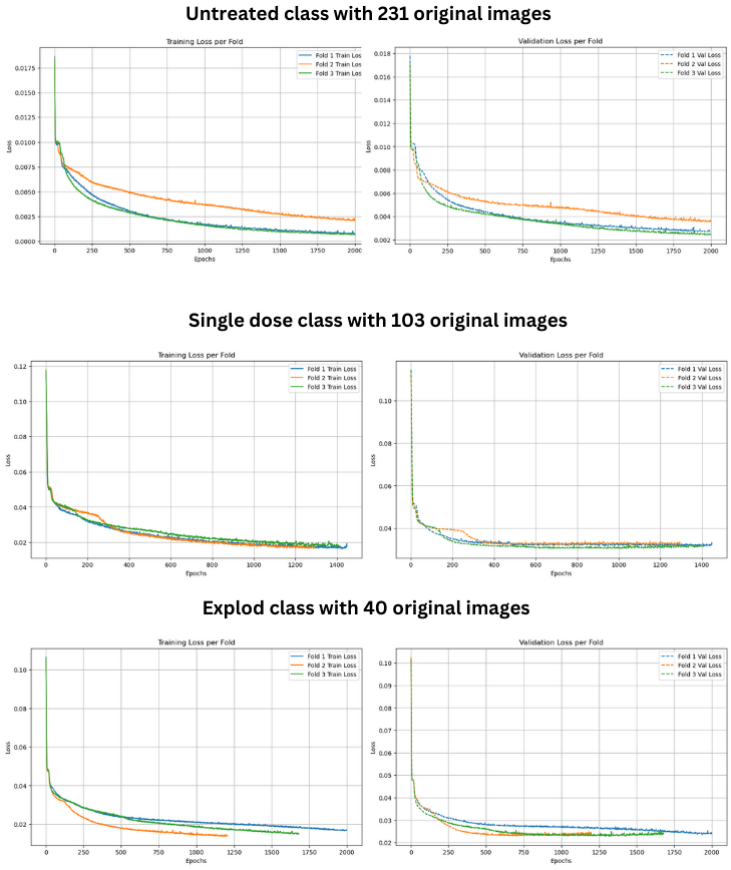

Figure 9.4: Wrong heading crct image upload Ohne contrast Resize before projection head data aug: Explod dataset plot in the figure shows overfitting

Do inference loss for both before and after projection head feats. used following metrics as inference loss between target feature vector and predicted feature vector.: mse, cosine distance, L2 (euclidean distance), L1 distance, Pearson correlation, dot product,jaccard similarity, hamming distance.

Reason to choose cosine distance is basically because the loss function of simclr is designed to make the cosine distance between similar images to be close to 1. the reason to choose other metrics are the metrics mainly used in medical images as explained in literature review.

results:

As explained before, if the cae trained solely on the transition of untreated images when we do the inerence metric calculation between predicted feature vector and target feature vector for these untreated images should be less than other classes except pearson, dot product and jaccard similarity. since for those metrics similartiy increases as the value of them increases as explained in metric chapter. we met that expectation as we can see the figure 9.5. and some of the metric are able to make a clear seperation of the training class to others ie atleat one class is have clear seperation from other classes as we seen in figure. and some of the metric couldn't make a clear seperation for atleast one class ie all of the classes are mixed as we seen in the figure for dot product and jaccard similarity. We observed this same trend for both before and after projection head features and for all transition category. Ideal situation that we would like to have is all of the classes are seperated from each other. so that when we do inference on drug screen images there will be look alike smooth transition from class to other.

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	34.01	67.22	66.55	66.39	-
	Control	54.03	68.10	53.15	66.29	-
	Explod	64.22	75.59	67.58	67.48	-
After	Single Dose	62.15	65.41	61.38	51.81	-
	Control	35.42	38.16	33.82	35.26	-
	Explod	48.40	64.32	61.01	36.40	-

Table 9.1: Cosine distance

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	80.71	66.60	66.39	66.13	-
	Control	48.45	67.11	55.79	66.03	-
	Explod	68.77	74.35	64.74	69.70	-
After	Single Dose	64.12	68.87	64.01	47.57	-
	Control	35.68	36.61	33.66	35.32	-
	Explod	51.50	56.41	37.80	44.98	-

Table 9.2: Euclidean distance

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	80.71	66.60	66.39	66.13	-
	Control	48.45	67.11	55.79	66.03	-
	Explod	68.77	74.35	64.74	69.70	-
After	Single Dose	64.12	68.87	64.01	47.57	-
	Control	35.68	36.61	43.80	35.32	-
	Explod	51.50	56.41	61.48	44.98	-

Table 9.3: MSE distance

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	80.61	66.60	66.39	65.05	-
	Control	50.98	66.75	51.34	66.03	-
	Explod	63.91	68.20	62.56	64.79	-
After	Single Dose	62.36	66.13	57.91	42.24	-
	Control	35.52	37.44	33.61	34.64	-
	Explod	48.71	52.12	59.10	41.68	-

Table 9.4: L1 distance

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	34.21	66.39	66.55	66.13	-
	Control	56.72	68.51	60.34	66.24	-
	Explod	63.96	67.01	65.67	65.51	-
After	Single Dose	48.50	64.68	48.24	50.52	-
	Control	47.78	43.02	44.26	34.49	-
	Explod	45.04	67.11	59.26	47.21	-

Table 9.5: Pearson correlation

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	65.99	34.13	33.40	34.23	-
	Control	42.09	49.28	34.90	40.12	-
	Explod	35.32	35.32	34.85	44.93	-
After	Single Dose	36.81	46.38	47.00	39.71	-
	Control	33.30	38.99	34.23	34.28	-
	Explod	34.18	45.45	33.87	36.19	-

Table 9.6: Dot product

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	28.43	66.65	48.55	62.72	-
	Control	56.36	68.46	61.89	68.67	-
	Explod	61.38	66.18	53.83	67.32	-
After	Single Dose	41.78	51.50	34.33	37.59	-
	Control	42.14	37.18	34.33	33.09	-
	Explod	39.14	50.78	42.55	35.47	-

Table 9.7: Jaccard similarity

Projection Head	Normal as	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose	78.58	66.60	62.67	59.20	-
	Control	37.28	60.24	48.60	64.84	-
	Explod	62.82	72.60	56.36	64.06	-
After	Single Dose	41.73	46.74	35.94	48.55	-
	Control	45.76	42.97	34.33	33.92	-
	Explod	45.24	55.17	43.07	43.07	-

Table 9.8: Hamming distance

range of cosine distacnes = 0-2 range of euclidean distacnes = working range of mse working

range of L1 distacne is working

range of pearson correlation coefficient: -1 to 1 working. 1 linearly correlated 0 no corelation, -1 inversly corelated. range of dot product working. range of jaccard : working 0 to 1 range of hamming distance is working.

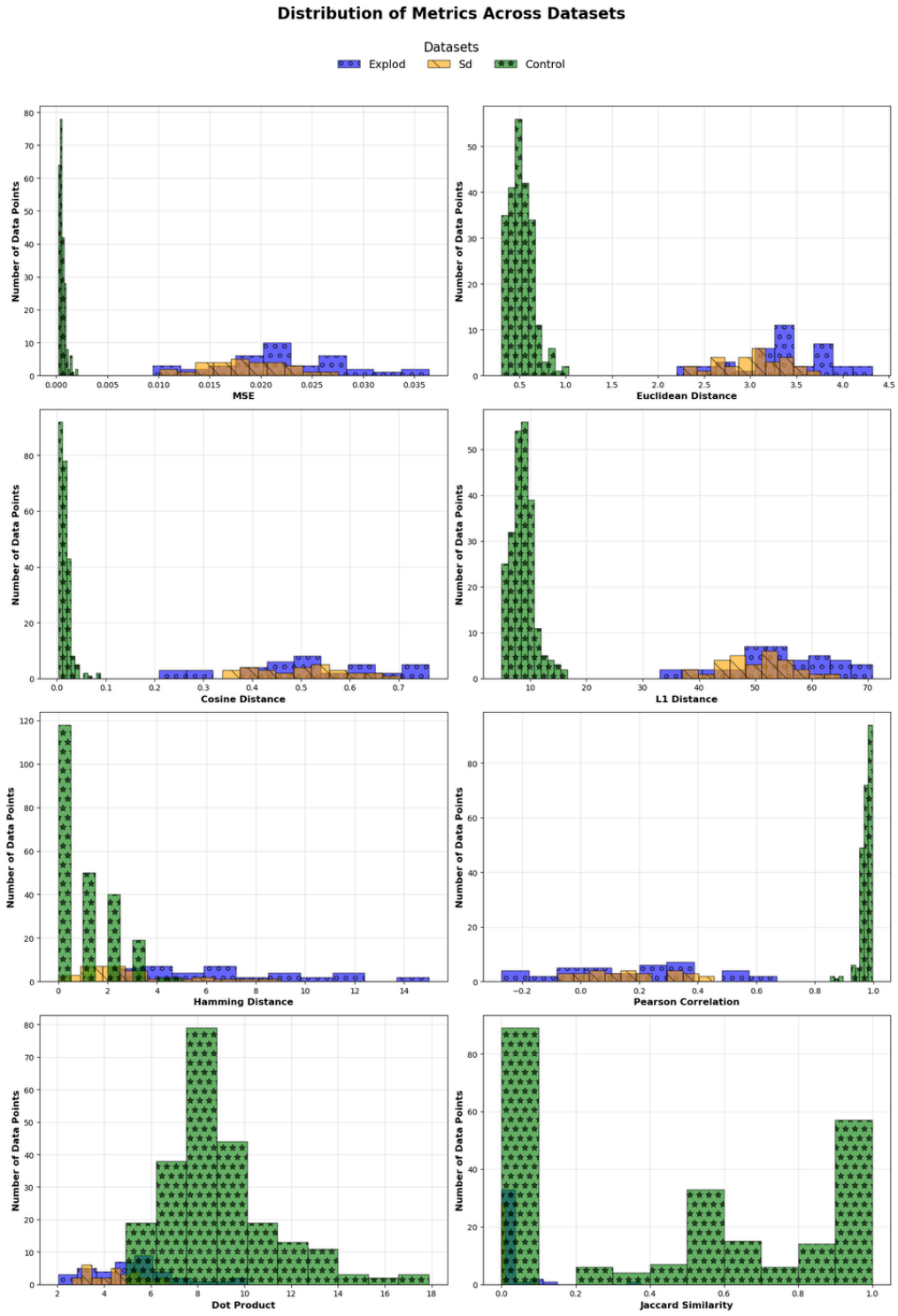

Figure 9.5: last row shows misposition of the cancer cell. in the day 7 it was above now. and in day 10 its below.

Since we couldn't seperate those 3 classes well that means we won't have good ranking of drug screnn images since sd close images and explod images will be mixed while ranking. this calls the need for other ranking strategy such as Softmax approach, K means centroid approach,

PCA approach. where for K means centroid , PCA approach small no of data will not effect that much for the performance of ranking. since there is no training required.

9.2 Ranking strategy 2: Mean approach

1. **Step 1:** Feed control images simclr features into k means and find the centriod of control (untreated) cluster based on both cosine distance and the euclidean distance using kmeans clustering.

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

3. **Step 3:** Rank the images based on the distance from the centroid of control simclr features.

4. **Step 4:** Perform the above ranking procedure from centroid of single dose image simclr features to other image features and from centroid of exploded image simclr features to other image features correspondingly.

5. **Step 5:** Perform the same operation on original images.

Ranking customized Accuracy Calculation

Accuracy formulation: Goal is to check whether we have clear seperation of each class by the inference metric while using as it as a scale. Accuracy below formulated to check whether the first and third classes are well-separated from the middle. there by it also penalizes if first and third classes are overlaped each other. If its clearly seperated from the middile means all 3 classes are well seperated.

Detailed explanation of the accuracy calculation:

Once we obtain the inference metric for each individual feature vector in each class, the next

steps involve calculating the mean inference metric for each class, identifying the middle class based on the mean values, and proceeding with the remaining computations as described below.

Task 1: Mean of Each Class

The mean inference metric for each class C_i is calculated as:

$$\mu_i = \frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} M(\mathbf{x}, \mathbf{c})$$

where $|C_i|$ represents the number of points in class C_i, and $M(\mathbf{x}, \mathbf{c})$ is the inference metric between a point $\mathbf{x}$ and the centroid $\mathbf{c}$.

Task 2: Middle Class Based on Mean

To determine the middle class, sort the mean values μ_1, μ_2, μ_3 in ascending order:

$$\mu_{\text{sorted}} = \{\mu_{\text{min}}, \mu_{\text{mid}}, \mu_{\text{max}}\}$$

The middle class is the class corresponding to μ_{mid}.

Task 3: Minimum and Maximum of Middle Class

For the middle class (denoted as C_{mid}), compute the minimum and maximum inference metric values:

$$\text{middle\_class\_min} = \min_{\mathbf{x} \in C_{\text{mid}}} M(\mathbf{x}, \mathbf{c})$$

$$\text{middle\_class\_max} = \max_{\mathbf{x} \in C_{\text{mid}}} M(\mathbf{x}, \mathbf{c})$$

Task 4: Error and Accuracy Calculation

1. **Error Calculation**: Count the number of points in the first class C_1 that exceed the middle

class's minimum, and the number of points in the third class C_3 that are less than the middle class's maximum. The total error is given by:

$$\text{error}_1 = |\{\mathbf{x} \in C_1 : M(\mathbf{x}, \mathbf{c}) > \text{middle\_class\_min}\}|$$

$$\text{error}_3 = |\{\mathbf{x} \in C_3 : M(\mathbf{x}, \mathbf{c}) < \text{middle\_class\_max}\}|$$

$$\text{total\_errors} = \text{error}_1 + \text{error}_3$$

2. **Accuracy Calculation**: Subtract the total errors from the total number of points across all classes to compute the number of non-errors. Divide this by the total number of points to calculate the accuracy:

$$\text{accuracy} = \frac{\text{total\_points} - \text{total\_errors}}{\text{total\_points}}$$

where:

$$\text{total\_points} = |C_1| + |C_2| + |C_3|$$

Summary of Steps 1. Compute μ_1, μ_2, μ_3. 2. Identify the middle class using μ_{mid}. 3. Compute middle_class_min and middle_class_max. 4. Count the errors error_1 and error_3. 5. Calculate accuracy using the formula above.

Dataset we used : As explained in the table, we used all of the control images (472), single dose image (103) and exploded images (40).

ds close to sd 10 worked 12 didn't worked

Projection Head	Distance From	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose Mean	93.33	93.50	48.23	92.52	93.82
	Control Mean	78.86	87.64	91.49	85.53	92.20
	Explod Mean	83.25	25.53	82.74	29.43	82.76
After	Single Dose Mean	**95.12**	85.69	28.29	77.72	83.74
	Control Mean	71.54	76.75	82.76	14.47	20.16
	Explod Mean	85.37	80.00	37.72	78.21	80.81

Table 9.9: Cosine distance

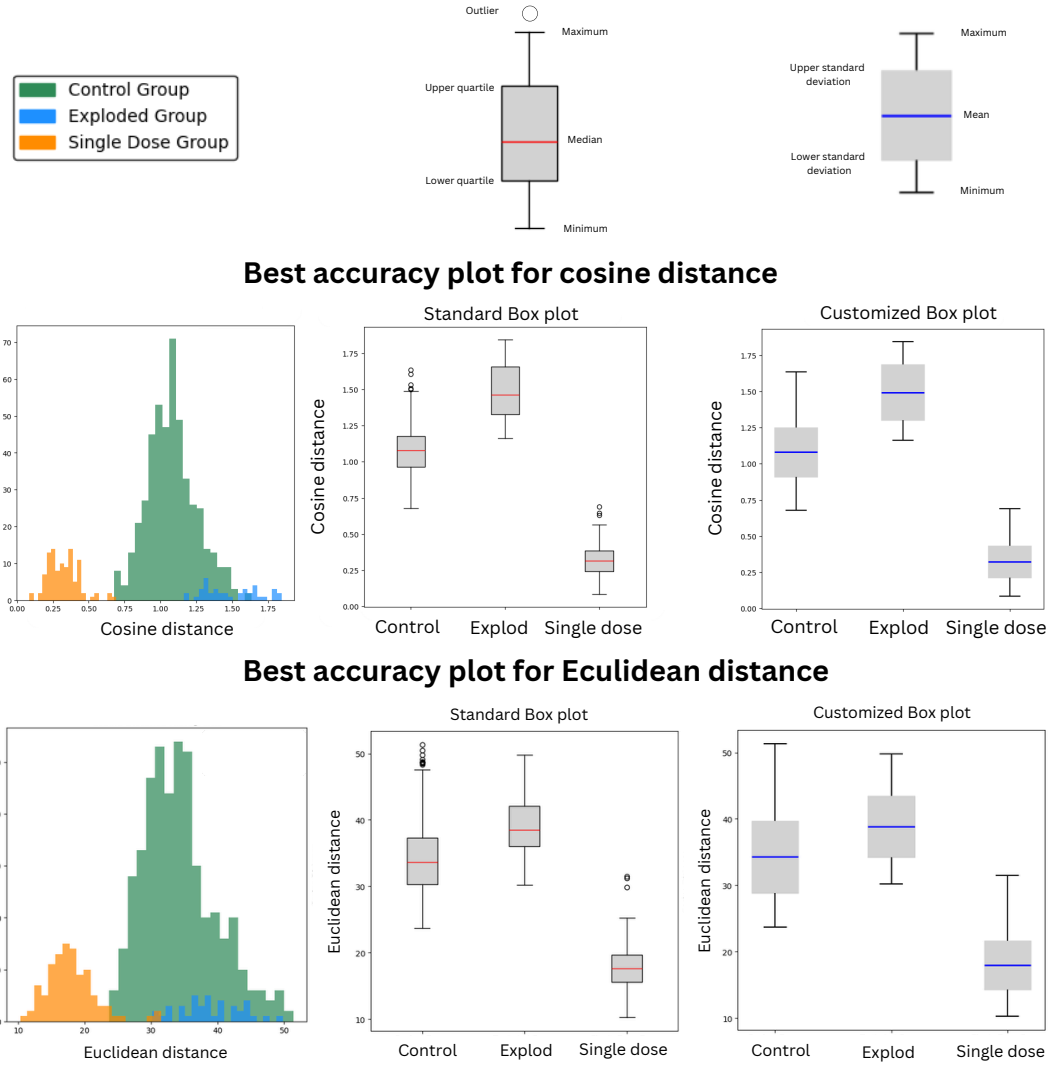

Figure 9.6: train vs test loss

add figures below.

Projection Head	Distance From	Strong	Sweet	Resize	No Contrast Resize	No Contrast Sweet
Before	Single Dose Mean	**92.52**	90.41	31.71	83.58	91.38
	Control Mean	69.11	37.24	76.75	39.84	57.89
	Explod Mean	82.44	82.60	67.97	23.90	82.28
After	Single Dose Mean	90.57	80.33	36.42	76.75	21.63
	Control Mean	55.70	76.75	76.75	76.75	77.07
	Explod Mean	83.09	77.40	55.93	19.84	81.63

Table 9.10: Euclidean distance

As you can see in the above table, cosine distance of each after projection head strong simclr feature from the single dose mean have the most seperation of classes with accuracy of 95.12. It doesn't seperate three classes fully but its something to hope for.

With the euclidean distance also each before projection head strong simclr feature from the single dose mean have the most seperation of classes with accuracy of 92.52. It clearly seperates one class from other but still far awy from the result of 3 class seperation that we hoped for.

By observing the plot of all inference metric scales the general trend of the accuracy metric is that if its achieved 90 percentage means it seperates one class from other while other 2 classes are still be mixed. And if it starts to increase from 95 percentage accuracy, it means it seperates those mixed 2 classes. 100 percentage means it seperates all classes clearly without any overlap.

For other inference metrics instead of kmeans clustering, I direcly calculated mean vector of each class to get the centroid vector. and treat it as the centroid of each class and calculated the accuracy as we did before. None of them achieved 90 percentage accuracy. Hence for this mean ranking approach cosine distacne as inference metric is the better choice for the ranking.

from both of the Cosine and euclidan table its evident that distance from single dose centroid using strong simclr fature is the better choice for the ranking comparing to others since it all achieves 90 percentage accuracy.

All of the above inference metric scaling I did it on original image vector and as you can see below perfromacne of the ranking accuracy using cosine and euclidean distacne were quite low comparing to the simclr feature vectors. Other inference metrics were also very low. Whcih

basically tells us for this ranking strategy also simclr features are better than original image vectors.

Distance From	Single Dose Mean	Control Mean	Explod Mean
Cosine	17.56	46.50	28.45
Euclidean	24.227	50.08	34.47

Table 9.11: 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. Steps:

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.

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 9.12: 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 9.13: 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.

ordered images can found in the below gdrive links for both mixed and cleaned order for ds

Table 9.14: Original Image Results

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

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 9.15: 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.

Conclusions - yes, what are they? What is the outcome. Important - it is fine to say that despite your best efforts the methods do not seem to deliver useful results. Perhaps they do not beat the baseline, this is very common in research and is perfectly fine, correct and useful to admit. Or they actually seem to perform randomly and are not reliable. This may be due to not enough data and is also ok to say, admit and point out. Or they do something but you are not actually sure, how to use for the final objective or ranking. This is also ok and you shall explain why you are not convinced. Perhaps the ranking is simply a badly formulated problem and it should not be approached this way at all. This is also ok to say and be open about it. Simply, even if it ”does not work” it is still VERY useful if we understand what you have done and why you think it does not work. 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?

References - clean these. Correct references shall have not only the names and year but also

the publication venue (journal, conference, arxiv... etc.)

10.2 Future research direction

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

2. Weakly supervised DINO transformer-based approach. 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.

3. Weakly supervised SimCLR approach.

4. Other distance-based approaches.

5. kl density estimation probability

6. Image size for simclr 96 vs 256 vs 512 as well as original images 96 vs 256 vs 512 vs even original size 2054?

7. other architectures. Resnet vs unet in Simclr

8. 3 channel vs 1 channel (mis alignment probelm or compuatoinal faster?)

9. Batch size 16 vs 64 vs 128 vs 256

10. after projectoin add l2 norm so that it will exactly like the loss fn gives good cosine sim

Appendix

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:

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

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.

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] 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.

[11] 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.

[12] 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