Degree Project in Data Science
Second cycle, 30 credits

Unsupervised Deep Learning to Study Microscopy Images

Leveraging Adaptable Variational Autoencoders for Microscopy-Based Cell Fate Analysis

ANNA KOVÁCS

Unsupervised Deep Learning to Study Microscopy Images

Leveraging Adaptable Variational Autoencoders for Microscopy-Based Cell Fate Analysis

Anna Kovács

Master's Programme, ICT Innovation
Industrial Supervisor: Juliette Griffie
Entry Year's Supervisor: Dániel Varga
Exit Year's Supervisor: Shirin Tahmasebinotarki
Examiner: Amir Hossein Payberah
School of Electrical Engineering and Computer Science (EECS)
Host company: SciLifeLab
Swedish title: Oövervakad Djupinlärning för att Studera Mikroskåpbilder
Swedish subtitle: Utnyttjande av Anpassningsbara Variational Autoencoders för Mikroskopibaserad Analys av Cellöde

SciLifeLab

Abstract

This thesis explores Variational Autoencoders (VAEs) for unsupervised learning on biomedical microscopy images, focusing on Multiple Sclerosis (MS) and lung cancer data. A wide range of VAE architectures were evaluated to identify optimal depth configurations that balance reconstruction quality and latent space regularization.
To reduce manual tuning, an adaptable VAE was developed using a layer interpolation formula that calculates model depth based on image resolution. This model was validated on unseen 80×80 images and compared against fixed 2- and 4-layer variants. Results show that the interpolated 3-layer design achieves the optimal balance between reconstruction fidelity and latent space usage, avoiding overfitting and collapse.
The proposed adaptable framework generalizes well across resolutions, offering a scalable and robust solution for microscopy-based medical image analysis.

Keywords

Microscopy, Machine Learning, Image Analysis, Data Analysis, Bioinformatics, Variational Autoencoder

Sammanfattning

Detta examensarbete undersöker användningen av Variational Autoencoders (VAE:er) för oövervakad inlärning på biomedicinska mikroskopibilder, med fokus på data relaterad till multipel skleros (MS) och lungcancer. Ett brett urval av VAE-arkitekturer utvärderades för att identifiera optimala nätverksdjup som balanserar rekonstruktionskvalitet och regularisering av den latenta representationen.
För att minska behovet av manuell justering utvecklades en anpassningsbar VAE som använder en interpolationsformel för att beräkna nätverksdjupet baserat på bildupplösning. Modellen validerades på tidigare osedda 80×80-bilder och jämfördes med fasta varianter med 2 och 4 lager. Resultaten visar att den interpolerade 3-lagersmodellen uppnår en optimal balans mellan rekonstruktionsfidelitet och användning av latent utrymme, samtidigt som överanpassning och kollaps undviks.
Det föreslagna anpassningsbara ramverket generaliserar väl över olika upplösningar och erbjuder en skalbar och robust lösning för mikroskopibaserad medicinsk bildanalys.

Keywords

Mikroskopi, Maskininlärning, Bildanalys, Dataanalys, Bioinformatik, Variationsautoencoder

Acknowledgments

I would like to express my sincere gratitude to my supervisors and examiner. Inês Cunha and Dániel Varga provided invaluable professional guidance throughout my research. I am also deeply thankful to Amir Hossein Payberah, who consistently answered my questions with great competence and care.

I would also like to acknowledge Emma Latron, whose work I had the opportunity to learn from and which gave me a new perspective on multiple sclerosis research. Her advice was highly valuable, and I could count on her support on countless occasions.

Finally, I would like to thank my family and friends, who stood by me despite the distance and supported me throughout the entire process. I owe a special thanks to my mother, who set me on this path and whose influence has shaped my greatest goal: to use my knowledge to help improve people's lives in the future.

Contents

1 Introduction

This chapter introduces the research area of unsupervised representation learning for high-resolution biomedical microscopy. It motivates the need for automated analysis of complex, high-dimensional image data and describes how Variational Autoencoders (VAEs) can address this real-life challenge. Section 1.1 provides background on microscopy imaging and deep generative models. Section 1.2 formulates the research problem and states the guiding research questions. Section 1.3 describes the overall purpose of this thesis, while Section 1.4 details the specific goals and objectives. Section 1.5 summarizes the research methodology. Section 1.6 presents the delimitation of this work. In Section 1.7 we discuss the ethical and sustainable aspects of the thesis. Finally, Section 1.7 presents the structure of the thesis.

1.1 Background

Medical imaging is a key approach of modern diagnostics, offering a non-invasive and increasingly detailed perspective into disease detection, diagnosis, and monitoring. However, traditional imaging techniques such as Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) often identify structural changes only once a disease has already progressed to an advanced stage. On the other hand, cellular-level changes frequently occur much earlier in the lifecycle of the disease, offering a potential time window for earlier detection. Microscopy, especially when applied to live-cell environments, provides high-resolution access to this critical level of biological detail. Yet, the complexity of microscopy data and its hardly recognizable features make manual analysis impractical and standard computational pipelines insufficient. To overcome this difficulty, automated, data-driven methods are therefore essential to extract meaningful patterns from large microscopy datasets. In the compression of high-resolution images into lower-dimensional latent spaces, unsupervised learning techniques, such as VAEs, show promise, since they preserve biologically relevant features. By learning a compact encoding of each image, VAEs can be used for downstream tasks (e.g., anomaly detection, cell fate prediction, early disease recognition) without relying on manual features. Moreover, an adaptable model that dynamically scales based on different image resolutions would enhance cross-platform reproducibility and reduce the need for manual reconfiguration when applying the same pipeline to new biological datasets.

1.2 Problem

Despite advances in imaging and initial experiments in deep learning, there is still a gap between raw microscopic data and information that can be used in practice. Specifically:

1. Manual or semi-automated methods often depend on engineered features, limiting their generalizability across different resolutions or imaging domains.
2. Noise, staining heterogeneity, and imaging artifacts can mask subtle patterns related to disease that can only be detected through a robust diagnostic and normalization process.
3. Existing VAE architectures are typically designed for a fixed input size. This property prevents the adaptation to images of different resolutions without manual adjustment of network depth and hyperparameters.

Without an integrated approach that can:

- Compress high-resolution microscopy images into a lower-dimensional, informative representation,
- Denoise and normalize variable imaging conditions,
- Adapt dynamically to datasets of unseen resolutions,

researchers lack a scalable, generalizable pipeline for large-scale, cross-platform microscopy analysis. To overcome these bounderies, this thesis addresses the following research questions:

1. Can we build a Variational Autoencoder (VAE) that effectively reduces the dimensionality of microscopy data?
2. Can we make this VAE adaptable to other biological datasets?

1.3 Purpose

The primary goal of this thesis is to develop an adaptable, generalizable, and dynamic unsupervised framework that leverages VAEs to learn compact, robust representations of microscopy images, while providing good-quality reconstructions. By encoding each image into a latent space that preserves essential morphological and molecular features, the proposed approach aims to:

- Reconstruct the original images with well-quality, capturing the primary features,
- Provide a model architecture that dynamically scales to different input sizes without manual reconfiguration.

This kind of adaptability is critical for studies where the input includes diverse biomedical images across several domains.

1.4 Goals

To address the research questions and fulfill the stated purpose, this work aspires the following objectives:

1. Construct and compare multiple VAE architectures of varying depths and downsampling strategies to identify configurations that balance reconstruction fidelity and latentspace regularization on high resolution MS microscopy images.
2. Systematically assess the impact of latent dimensionality (e.g., $d_z = 2$ vs. $d_z = 20$) on disentanglement and reconstruction quality, selecting a model that achieves the optimal trade-off.
3. Adapt the selected VAE for low-resolution lung cancer cell imaging by reducing network complexity (fewer convolutional layers) while maintaining an informative latent encoding.
4. Implement and validate a layer-interpolation formula that determines the appropriate number of layers based on input image size, creating an "adaptable VAE" capable of handling arbitrary microscopy resolutions without manual tuning.

1.5 Research Methodology

In the first phase, six candidate VAE architectures—differing by downsampling strategy (MaxPooling vs. strided convolution + BatchNormalization) and depth (2, 3, or 4 layers)—are implemented in Python using PyTorch. All models are trained for 4,000 epochs on memory B-cell immunofluorescence images (two channels, 140×140 pixels) from MS patients and healthy controls. Reconstruction loss (mean-squared error with sum reduction) and KL divergence are saved per epoch to calculate the total VAE loss:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}}.$$

Each architecture is evaluated at latent dimensions $d_z = 2$ and $d_z = 20$ to assess trade-offs between compression and expressivity. The final model (a 4-layer MaxPooling VAE with $d_z = 2$) is then tested across β values.
In the second phase, the selected VAE is retrained on 20×20-pixel lung cancer cell images, reducing architecture depth to match the smaller input size. Two-layer and three-layer variants (e.g., VAE_MP2_64, VAE_MP3_128) are compared again fixing $d_z = 2$. Performance, measured via reconstruction loss, latent KL, and qualitative visual assessment, is used to select the simplest model that preserves critical cellular features.
Finally in phase 3, a layer interpolation formula can be derived based on all previous experiments, mapping input resolution to the optimal number of convolutional blocks. This formula is validated on resized lung cancer images that unseen to confirm that the "adaptable VAE" (using interpolated depth) achieves comparable reconstruction fidelity to manually designed architectures, while eliminating the need for dataset-specific re-engineering.

1.6 Delimitation

This study used two specific fluorescence-based microscopy datasets: FACS-sorted memory B-cell imaging for Multiple Sclerosis and FRET-labeled lung cancer cells. Other imaging modalities (e.g., brightfield, phase contrast, histology) are not evaluated. Only VAEs are considered; traditional machine learning pipelines, alternative generative models such as Generative Adversarial Networks (GANs) or normalizing flows are out of scope. Although the adaptable VAE is designed for a range of image sizes, performance on extremely high-resolution ($> 512 \times 512$) or multi-channel (> 2 channels) inputs has not been assessed. All experiments use PyTorch. This work emphasizes reconstruction fidelity and latent-space compactness; downstream tasks (e.g., classification, clustering) are not implemented here.

1.7 Social Aspects

1.7.1 Social Relevance

Nowadays, the convergence of biomedical imaging and artificial intelligence (AI) shows significant impacts on how we interpret cellular-level data for healthcare applications. Traditionally, high-resolution microscopy required expert interpretation, using manual effort, subjective variability, and scalability limitations. However, with the integration of deep learning techniques, such as Variational Autoencoders (VAEs), there is more capacity to analyze complex, underlying cellular patterns at a scale and speed previously unattainable [17, 5].
This thesis contributes to that frontier by designing a VAE framework capable of adapting to various biomedical datasets, enabling broader applicability across different disease domains, such as Multiple Sclerosis and lung cancer. This model lowers the barrier to adopting AI in clinical and research settings by reducing the dependence on manual reconfiguration. In the long term, these tools can support earlier disease detection, assist in treatment stratification, and potentially help make real-time decisions during clinical workflows [19].

1.7.2 Sustainability

There is a cost of the rapid acceleration of AI in medical imaging. Training deep neural networks, particularly on high-resolution microscopy data, consumes significant energy. Nowadays, the environmental footprint of AI technologies has become a significant concern, especially in the field of medical imaging where large datasets and complex models are common [25].
It is essential to evaluate model design through the lens of computational sustainability. This work proposes an adaptable VAE that adjusts its architecture to match the resolution of the input data. This dynamic approach reduces redundant computation and avoids overfitting models to low-complexity tasks. Using an adaptable structure leads to lower training times and reduced hard-

ware demands, making the solution more scalable to resource-constrained settings such as smaller labs or hospitals without access to high-performance workstations. Furthermore, another aspect can be improving the efficiency and quality of early disease detection. It can indirectly minimize the carbon and material costs of prolonged, late-stage treatment plans and repeated imaging procedures. In this way, the proposed methodology serves computational sustainability and supports more efficient resource usage across the medical pipeline [15].

1.7.3 Ethics

Deploying machine learning (ML) models in the biomedical field presents several ethical challenges. First, the opacity of deep learning systems makes clinical interpretability a concern. Even though the proposed framework operates unsupervised, the decision it informs may influence high-stakes diagnosis. If the internal representations are misunderstood or misused, there is a risk of unintended bias or overreliance in automated pipelines [28].
Furthermore, generalizability across datasets raises ethical concerns around fairness. Different group domains, across age, ethnicity, or disease subtype, may exhibit subtle imaging differences. Thus, a model trained on one population may underperform or misrepresent patterns in another. This approach is essential when considering the translation of a model validation on one dataset to another, as done in this project.
Finally, when there is increasing automation in biomedical interpretation, the issue of accountability appears. If a model assists in a diagnostic decision incorrectly, it remains unclear whether the responsibility lies with the clinician, the developer, or the institution deploying the system. This work focuses on research application and proof-of-concept architectures, but we still have to pay attention to the ethical imperative aspects. Transparency, fairness, and cautious deployment must accompany technical innovation in AI-driven healthcare tools [6].

1.8 Structure of the Thesis

- Chapter 1: Introduction, including background, problem statement, research questions, purpose, goals, methodology, delimitations, and thesis structure.
- Chapter 2: Background, collecting all the necessary knowledge for a deeper understanding and literature review on generative models—specifically VAEs—and prior applications in biomedical microscopy.
- Chapter 3: Experimental design and implementation details for VAE architectures across MS and lung cancer datasets; derivation of the adaptable layer-interpolation formula.
- Chapter 4: Quantitative and qualitative results, including reconstruction errors, KL losses and visual comparisons across datasets.

- Chapter 5: Conclusions, limitations, and future directions, such as extending the adaptable VAE to additional modalities and integrating downstream predictive models.

2 Background and Related Work

This chapter deepens the motivations and objectives outlined in chapter 1 by first grounding our work in the theory of unsupervised representation learning with the introduction of the general autoencoder and its probabilistic extension, the Variational Autoencoder (VAE). We then survey how these models have been adapted to key medical imaging modalities, such as X-ray, CT, or MRI. In section 2.2, we highlight recent advances tailored to microscopy and dermatology. By identifying both the strengths of these approaches and the remaining challenges, we set the stage for our own VAE-based framework. chapter 3 will present the detailed network architecture, loss formulations and training strategy; chapter 4 will describe the datasets and experimental design; and chapter 5 will report our empirical findings and chart directions for future research.

2.1 Background

Integrating Machine Learning (ML) and Deep Learning (DL) techniques has revolutionized several fields, including medical imaging and bioinformatics. The use of DL enhances the analysis of complex datasets, improving diagnostic accuracy and facilitating personalized medicine. This section provides an overview of the background and related work in the application of autoencoders, especially Variational Autoencoders (VAEs), in medical image analysis. Image analysis is crucial to make important decisions and predictions, and it is one of the key steps in recent studies.

Furthermore, this section elaborates on the application of ML in lung cancer and Multiple Sclerosis (ML), emphasizing the potential of combining image analysis with ML models to improve disease diagnosis and prediction.

2.1.1 Autoencoders

Autoencoders are a class of unsupervised neural networks that learn to encode high-dimensional data into a compact latent representation and then decode that representation to reconstruct the original input. Their main focus is on capturing the most important features of the data while reducing noise and redundancy. These models can be useful in domains like medical image analysis, where images often suffer from variability and artifacts, because autoencoders can be used for image compression, anomaly detection, dimensionality reduction, and denoising.

2.1.1.1 General Autoencoders

The main concept includes two main components:

- Encoder: A function $f_\theta(.)$ that maps the input data x to a latent representation z:
$$\mathbf{z} = f_\theta(\mathbf{x})$$
- Decoder: A function $g_\phi(.)$ that reconstructs the input from the latent code:
$$\hat{\mathbf{x}} = g_\phi(\mathbf{z})$$

During the model's training process, the goal is to minimize the reconstruction error. This is typically measured by a loss function such as mean squared error (MSE) or cross-entropy loss. It can be written as:

$$\mathcal{L}_{\text{rec}} = \|\mathbf{x} - \hat{\mathbf{x}}\|^2,$$

Which encourages the output $\hat{\mathbf{x}}$ to be as similar as possible to the input $\mathbf{x}$.

2.1.1.2 Types of Autoencoders

Over the years, several variants of autoencoders have been developed to solve specific challenges.

One of the earliest types was introduced by Vincent et al. (2008)[29]. This model is the Denoising Autoencoder (DAE), which is designed to reconstruct the original input from a corrupted version, so it learns robust representations that are resilient to noise. By training the model to denoise corrupted inputs, DAEs capture essential structures in the data, making them effective for tasks such as image denoising and feature extraction.

Rifai et al. (2011)[24] proposed the Contractive Autoencoders (CAEs). These models add a regularization term to the loss function that penalizes the sensitivity of the encoded representations to small changes in the input. It means that the contractive penalty encourages the model to learn features that are invariant to small input variations, thereby capturing the underlying structure of the data.

In 2013, Kingma and Welling (2013)[14] developed Variational Autoencoders (VAEs), which introduce a probabilistic framework to the autoencoder architecture. The main concept is that VAEs assume that the latent variables follow a prior distribution, usually a standard normal distribution, and aim to learn a posterior distribution that approximates this prior. The model is trained by maximizing the evidence lower bound (ELBO), which balances the reconstruction accuracy and the divergence between the learned posterior and the prior. This probabilistic formulation allows VAEs to generate new data samples and has been widely applied in generative modeling tasks.

As another important variant of autoencoders, Adversarial Autoencoders (AAEs) were introduced by Makhzani et al. (2015)[18]. These models combine the autoencoder architecture with adversarial training. In this framework,

the encoder's output is regularized by matching the aggregated posterior of the latent variables to a specified prior distribution using a discriminator network. This approach allows AAEs to perform variational inference and generate data samples, making them suitable for applications in generative modeling and semi-supervised learning.
The diversity of autoencoders has a huge impact on developing the field of unsupervised learning, because models are able to learn meaningful and robust representations from complex data.

2.1.1.3 Variational Autoencoders (VAEs)

In this degree project, VAEs are one of the key components of the architecture. Using this model makes it possible to use meaningful latent representations, which is really important in the case of classification.

VAEs, introduced by Kingma and Welling in 2013[14], are generative models that combine principles from variational inference and neural networks. The main difference between the traditional autoencoders and the VAEs is that traditional ones deterministically map inputs to latent representations; however, VAEs learn a probabilistic distribution over the latent space, enabling the generation of new data samples.

In order to gain a deeper insight into its mathematical background, in a VAE, the encoder maps an input x to a latent variable z, characterized by a mean μ and a standard deviation σ. The decoder then reconstructs x from z. The model is trained to maximize the Evidence Lower Bound (ELBO), which consists of two terms:

1. Reconstruction Loss: Measures how well the decoder can reconstruct the input from the latent variable.
2. Kullback-Leibler (KL) Divergence: Regulates the learned latent distribution to be close to a prior distribution, typically a standard normal distribution.

ELBO is expressed as:

$$\mathcal{L} = E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})) \tag{2.1}$$

Here, $q_\phi(z \mid x)$ is the approximate posterior distribution, $p_\theta(x \mid z)$ is the likelihood of the data given the latent variable, and $p(z)$ is the prior distribution over the latent variables.
A key to VAEs is the reparametrization trick, which enables efficient backpropagation during training. Instead of sampling z directly from $q_\phi(z \mid x)$, the model expresses z as:

$$\mathbf{z} = \mu + \sigma \odot \epsilon \tag{2.2}$$

Where

$$\epsilon \sim \mathcal{N}(0, I)$$

Is an auxiliary noise variable. This formulation allows gradients to flow through μ and σ during optimization.
By integrating VAEs into our architecture, their strengths in capturing complex data distributions and generating high-quality samples can enhance the overall performance and robustness of the system.

2.1.2 Medical Image Analysis

Medical image analysis plays an important role in medical research. It involves the extraction of meaningful information from various medical imaging modalities to assist in diagnosis, treatment planning, disease monitoring, and disease prediction. The main goal is to identify regions of the anatomy affected by disease, thereby making it easier to understand lesion progression [13].

2.1.2.1 Types of Medical Imaging Modalities

Medical imaging includes several types of methods, each providing unique insights into the human body [13]:

- X-ray Imaging: Utilizes ionizing radiation to capture images of dense structures like bones. It is one of the most common diagnostic tools globally, especially for chest and bone imaging [23].

- Computed Tomography (CT): Combines multiple X-ray images to create cross-sectional views, offering detailed information about internal organs. CT imaging provides high-resolution 3D visualization, crucial for detecting tumors and internal injuries [30].

- Magnetic Resonance Imaging (MRI): Employs magnetic fields and radio waves to generate detailed images of soft tissues, such as the brain and muscles. MRI is especially valued for its superior soft-tissue contrast without the use of ionizing radiation [8].

- Ultrasound: Uses high-frequency sound waves to visualize soft tissues and blood flow, commonly used in obstetrics and cardiology. It is widely adopted due to its real-time capability and safety [22].

2.1.2.2 Microscopy in Medical Image Analysis

Microscopy is necessary in medical research, especially for examining cells and tissues at high resolution. Advancements in technology and digitization have made it possible to enhance tissue-based research via digital microscopy and image analysis. Whole slide imaging scanners enable the digitization of histology slides to be stored.

2.1.3 Autoencoders in Medical Image Analysis

Autoencoders are a type of neural network architecture designed for unsupervised learning. These models have found significant applications in microscopy image analysis. Since, they can learn compact, meaningful representations of data, they are particularly suited for handling the complex high-dimensional nature of datasets.

One important advantage of autoencoders is that they are suitable for feature extraction and representation learning. In microscopy, images often contain intricate structures and patterns that are hard to analyze using traditional methods. These models address this by encoding the input images into a lower-dimensional latent space, capturing the most important features while reducing noise and redundancy.

On the other hand, microscopy images are often sensitive to different types of noise due to limitations in imaging techniques and environmental factors. Autoencoders, especially denoising autoencoders, have made it easier to improve image quality by reconstructing clean images from noisy inputs. This is a key step for accurate visualization and analysis of microscopic structures. In 2019, Niu et al. Introduced a Fully Convolutional Deep Denoising Autoencoder (DDAE) [21] model which effectively preserved important cellular features, such as cell boundaries, while reducing noise, so it made downstream analysis more accurate [20]. Similarly, recent work has shown that deep denoising autoencoders can enhance high-resolution microscopy images (such as ChromSTEM) without sacrificing key biological structures [2].

Furthermore, VAEs, a probabilistic extension of traditional autoencoders, have been utilized in microscopy for generative modeling tasks. These models learn the underlying distribution of the data and generate new, synthetic microscopy images that are statistically similar to the original dataset. This capability is valuable for data augmentation, especially in scenarios with limited available data.

2.2 Related Work

The integration of autoencoders into medical image analysis has significantly increased in recent research, which has led to advancements in feature extraction, image enhancement, and generative modeling. This section reviews key studies in these fields, highlighting architectures and findings that can contextualize this project.

One recent study has focused on using autoencoders for feature extraction and denoising in microscopy images. Casti et al. (2023)[4] introduced the S3-VAE (Supervised-Source-Separation Variational Autoencoder), which combines supervised learning with variational autoencoding to create the latent space. This architecture enhances class separability and disentangles confounding factors, thereby enabling more accurate discrimination of cell types in single-cell microscopy data. Similarly, Rotem et al. (2024) developed the

DISCOVER model, which integrates an Adversarial Autoencoder (AAE) with a classifier. By combining perceptual losses and adversarial training, their model achieves high-quality image reconstruction and preserves the features most relevant to classification tasks.

Another critical requirement in the microscopy image analysis is denoising. Images are often compromised by noise due to the limitations of imaging and environmental factors. A study by Yang et al. (2019) [31] proposed a Dual Adversarial Autoencoder for dermoscopic image analysis that addresses challenges such as data augmentation and noise reduction. This model improves the reliability of downstream diagnostic tasks because it enhances image quality. Similarly, Niu et al. (2019) introduced a Fully Convolutional Deep Denoising Autoencoder (DDAE) [21] specifically designed for improving the quality of microscopy images. This model uses advanced techniques such as three-photon fluorescence and third harmonic generation. Their work demonstrated that the architecture could preserve critical cellular features, like cell boundaries, while effectively reducing noise, which is essential for accurate image analysis.

Collectively, these studies highlight the versatility and robustness of autoencoders in microscopy image analysis. They play an important role in denoising, feature extraction, and generative modeling, which makes these models highly suitable for improving both the quality and interpretability of microscopy images. This is the reason why they are used in several medical fields.

3 Problem Description and Methodology

After discussing the background and motivation presented in chapter 1 and the literature survey in chapter 2, this chapter defines the precise computational problem we address and outlines the methodological framework used to tackle it. Specifically, we frame the challenge of learning compact yet expressive representations of high-resolution microscopy images with VAEs, and then describe the three-phase experimental strategy we follow:

- In Phase 1 (section 3.3), we identify a final VAE architecture for the Multiple Sclerosis (MS) dataset by comparing variants using different depth and downsampling strategies.
- In Phase 2 (section 3.4), we take the best MS-trained model as a starting point, reduce the number of layers to suit 20×20 lung cancer images, and select a specific VAE configuration for the lung cancer dataset.
- In Phase 3 (section 3.5), we derive and implement an adaptable VAE that dynamically computes the required number of layers based on input image size, using the outcomes of Phase 1 and Phase 2 and creating a single, dynamic framework.

By situating the problem and methodology in one chapter, we provide a clear description of our main challenge and an explanation of how we would like to overcome this problem. The results and analysis in chapter 4 will then directly build on the models and procedures defined here.

3.1 Problem Description

High-resolution microscopy images contain rich phenotypic information about cellular and subcellular structures, but their size and variability pose two main challenges:

1. Dimensionality and Noise: A single fluorescence microscopy image (e.g. 140×140 pixels, two channels) has a huge amount of pixel values, plus experimental noise and staining heterogeneity. Extracting a low-dimensional representation that retains biologically relevant features, while discarding noise, is nontrivial.
2. Cross-Platform Adaptability: A VAE architecture tuned for 140×140 MS images does not readily transfer to a 20×20 lung cancer dataset. Manual reconfiguration of layer counts and feature map sizes for each new resolution is time-consuming and error-prone.

Taking into account the above-mentioned challenges, our goal is to design an unsupervised framework that:

- Compresses each microscopy image into a low-dimensional latent space ($d_z = 2$), capturing the most important features.
- Denoises and normalizes across variable imaging conditions (e.g. illumination, staining).
- Scales automatically to different input resolutions (e.g. 140×140 vs. 20×20) without manual tuning of network depth.

In our project, this problem consists of three major sub-problems:

Phase 1: Identify and compare multiple VAE variants on the 140×140 MS dataset, selecting a final configuration that balances reconstruction fidelity and latent space regularization.

Phase 2: Starting from the Phase 1 final model, selectively reduce the number of layers to adapt to 20×20 lung cancer images, while preserving the internal block structure (kernel sizes, activation functions, etc.).

Phase 3: Formulate a layer-interpolation rule that, given any input image size, computes the number of blocks required. Implement a single adaptable VAE that uses this rule at runtime.

3.2 Methodology

This section outlines the three interconnected phases of our experimental strategy. Each phase corresponds to one of the sub-problems identified above, and builds directly on the preceding phase's outcome:

- section 3.3 (Phase 1): Implement six candidate VAE architectures on the MS dataset, varying downsampling technique (Max Pooling vs. Strided Conv+BatchNorm) and network depth (2, 3, or 4 blocks). Train all models under identical conditions and select the final MS model (`VAE_MP4` at $d_z = 2$) based on reconstruction loss, KL divergence, and parameter count.
- section 3.4 (Phase 2): Take the Phase 1 final model as a template. For the 20×20 lung cancer images, systematically reduce the number of downsampling layers (from 4 to 3 or 2), retaining the same per-block structure (kernel size 3, padding 1, ReLU activations, final Sigmoid). Compare variants—`VAE_MP3_128`, `VAE_MP3_256`, `VAE_MP2_64`, `VAE_MP2_128`—and choose a lung-specific final model (`VAE_MP2_64` at $d_z = 2$).
- section 3.5 (Phase 3): Derive a layer-interpolation formula that maps input image size (e.g. $N \times N$) to the required number of convolutional/downsampling blocks so that the bottleneck resolution stays in a target range (e.g. 8×8 for MS or 3×3 for lung). Implement an adaptable VAE that, given any N, dynamically allocates the appropriate number of blocks, reusing the same per-block design as Phases 1–2.

3.3 Phase 1: Implementing Variational Autoencoder using Multiple Sclerosis Data

3.3.1 Dataset Description

The dataset comprises immunofluorescence confocal microscopy images of memory B cells collected from two cohorts: multiple sclerosis (MS) patients treated with Natalizumab and healthy controls. Memory B cells were isolated from peripheral blood mononuclear cells (PBMCs) using fluorescence-activated cell sorting (FACS), targeting the surface markers CD19 and CD27 [11]. The cells were subsequently stained for a panel of proteins of interest, including CD38, CD43, CD11c, CXCR3, TBET, BTK, IgG, and IgM. In addition, the cell membranes were labeled using a lipophilic dye to facilitate morphological visualization [16].
For image analysis, the cells were segmented using the Cellpose algorithm [26], and individual cells were cropped from the original fields of view into smaller 140×140 pixel regions centered on each cell (see Figure 2 for examples).

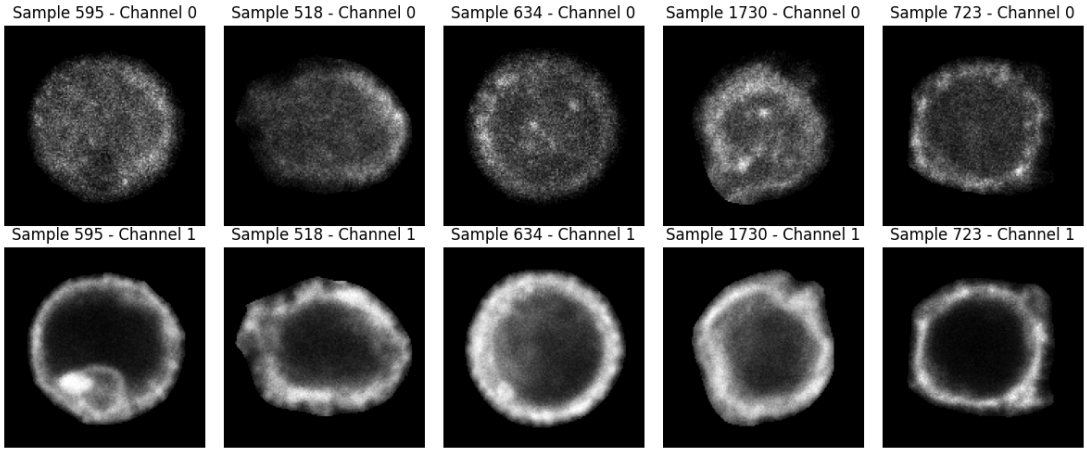


Figure 1: Randomly selected memory B cell samples from the MS dataset.

3.3.2 Model Architecture Experiments

In this part, six VAE architectures were developed and trained on the MS microscopy dataset. While the experiments were taken, the following design choices were held constant to allow a fair comparison:

- Input: Inputs include two-channel, 140 x 140-pixel microscopy images.
- Training parameters: Each model was trained for 4000 epochs on the same workstation using the same optimization settings (learning rate, batch size, etc.).
- Latent Sampling: Encoders project two linear heads (`fc_mu`, `fc_logvar`) and use the parametrization trick to sample latent vectors.

- Decoder Bridge: There is a single linear layer (`fc_decode`) that maps latent vectors back to the flattened spatial representation for the decoder.
- Activation Function: Every convolutional and transposed-convolutional layer (except the final output) uses a ReLU activation function.
- Output Constraint: The final decoder layer uses a Sigmoid activation to produce outputs in [0, 1]. This approach is required by the loss value calculations.

The main focus was to observe how different architectural choices influence the model's ability to reconstruct input images effectively. The structures are categorized based on two key design elements:

- Downsampling Technique: Usage of Max Pooling (MP) or strided convolutions combined with Batch Normalization (BN).
- Network Depth: Implementation of 2, 3, 4 convolutional layers in both the encoder and decoder.

The above-mentioned categorization is the cause of the following model labeling:

- **`VAE_MP2`**, **`VAE_MP3`**, **`VAE_MP4`**: Apply Max Pooling with 2, 3, or 4 layers.
- **`VAE_BN2`**, **`VAE_BN3`**, **`VAE_BN4`**: Utilize Batch Normalization with 2, 3, or 4 layers.

The reason for exploring Batch Normalization was encouraged by a paper where the S3-VAE model was implemented by Casti et al. (2023)[4], which improved training stability and convergence in similar contexts.

3.3.2.1 `VAE_MP2` *Architecture*

This architecture uses two convolutional blocks, each followed by a 2×2 Max Pooling layer to downsample the input 140×140 input to 70×70 and then to 35×35. The encoder uses 32 and 64 channels, which means a flattening step to $64 \times 35 \times 35$ feature map. Moving forward to a projection into a two-dimensional latent space via separate `fc_mu` and `fc_logvar` heads. The decoder mirrors this structure, using two ConvTranspose2d layers ($64 \to 32$, then $32 \to 2$ channels) and ReLU activation functions. As a last step, a Sigmoid activation produces the output in $[0, 1]$. This model has two convolutional layers, and it provides the simplest model, taking the base for the following models using Max Pooling.

3.3.2.2 `VAE_MP3` *Architecture*

As an extension of `VAE_MP2`, `VAE_MP3` includes a third convolutional block using Max Pooling. This approach reduces the spatial dimension to 17×17 and increases channels to 128 at the bottleneck. Here, the flattening step produces

a map with the size of $128 \times 17 \times 17$. Apart from that, the latent sampling and decoding operations work similarly to the MP2 version but with three ConvTranspose2d upsampling stages. Using three convolutional blocks explores the importance of richer feature representations at the cost of spatial details.

3.3.2.3 `VAE_MP4` *Architecture*

This Max Pooling architecture presents the deepest pooling variant, adding a fourth convolutional block using pooling that compresses inputs to an 8×8 feature map of 256 channels. As the number of layers was increased in the encoding, it was increased during the decoding phase as well, adding a fourth ConvTranspose2d layer before the final bilinear upsampling step as a guarantee of reaching 140×140 output.

3.3.2.4 `VAE_BN2` *Architecture*

Changing one of the main components in the experiments, this model uses two Conv2d layers, each followed by BatchNorm and ReLU, to downsample from $140 \to 70 \to 35$ and produce a $64 \times 35 \times 35$ feature map. Following the same pattern, the decoder inverts this with two blocks, including ConvTranspose2d and BatchNorm. The `VAE_BN2` architecture examines whether BatchNorm improves reconstruction quality compared to Max Pooling in `VAE_MP2`.

3.3.2.5 `VAE_BN3` *Architecture*

Moving further along the BatchNorm approach, this model deepens the previous architecture by one more block. It uses three strided convolutions ($140 \to 70 \to 35 \to 18$) and corresponding BatchNorm layers, producing a $128 \times 18 \times 18$ feature map. In case of `VAE_MP3`, this spatial dimension slightly differs from its 17×17 feature map. The reason of this discrepancy is how the strided convolution calculates output dimensions. In this case, each convolutional layer with kernel size $k = 3$, stride $s = 2$, and padding $p = 1$, follows the output dimension formula:

$$\text{out} = \left\lfloor \frac{\text{in} + 2p - k}{s} \right\rfloor + 1.$$

Applying this formula to an input of size 35×35, we get:

$$\text{out} = \left\lfloor \frac{35 + 2 \cdot 1 - 3}{2} \right\rfloor + 1 = 18.$$

As a result, the final decoder output from the 18×18 feature map requires an additional bilinear upsampling step to get the original input resolution of 140×140.

3.3.2.6 `VAE_BN4` *Architecture*

This variant of the BatchNorm approach uses a depth of four convolutional layers ($140 \to 70 \to 35 \to 18 \to 9$), reaching 256 channels at the deepest point. After the latent sampling, the decoder uses four times a ConvTranspose2d supplemented with BatchNorm, finalizing the process with a bilinear upsampling to reach 140×140 resolution. This model is closely aligned with the S3-VAE architecture introduced by Casti et al. (2023)[4] in terms of number of layers, usage of BatchNorm, and ReLU activation functions. Although the exact parameters and configurations were not publicly released for S3-VAE, this implementation mirrors its core architectural principle. Therefore, `VAE_BN4` tests whether extensive normalization can maintain stable training and high-quality reconstructions in very deep architectures.

3.3.3 Latent Space Dimensionality Experiments

Latent space's dimensionality has a big effect on VAEs. It has a key role in finding the trade-off between compression and reconstruction fidelity. Choosing a small latent space forces the network to discard more information, while promoting stronger compression and risking loss of meaningful features. This trade-off between compression and reconstruction quality is well-documented by Higgins et al. [12]. On the other hand, a higher-dimensional latent space can represent finer-grained variation but is also able to under-regularize the posterior and choose redundant components. To observe this trade-off, each of the six architectures (subsection 3.3.2) was evaluated at two different dimensions:

- $d_z = 2$ (lower-dimensionality compression)
- $d_z = 20$ (higher-dimensionality embedding)

In addition to these, using $d_z = 10$ was also observed as an intermediate latent dimensionality. However, the results for $d_z = 10$ closely showed those of $d_z = 20$, both in terms of reconstruction quality and total VAE loss. Since the difference between the two higher-dimensional setups was minimal, we chose to report only the outcomes for $d_z = 2$ and $d_z = 20$ in this thesis. Representing the extremes of the trade-off spectrum more clearly.

3.3.3.1 Experimental Setup

Each architecture from subsection 3.3.2 was trained with two different latent space size ($d_z = 2$ versus $d_z = 20$. Apart from that, all other settings (number of epochs, batch size, learning rate, optimizer, etc.) were the same as described in Section 2.2.

3.3.3.2 Evaluation Metrics

To be able to compare the performance of the models, the following loss values were calculated:

- Reconstruction Loss

$$\mathcal{L}_{\text{rec}} = \sum_{i=1}^{N} (x_i - \hat{x}_i)^2$$

  The model uses Mean Squared Error (MSE) as a reconstruction loss with sum reduction instead of the default mean reduction. It means that the squared errors are summed, hence tying the magnitude of the loss directly to image size and batch size.

- KL Divergence

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}\big(q(z \mid x) \,\|\, p(z)\big)$$

  Using this value measures how closely the learned posterior follows the prior distribution, encouraging the latent representations to be Gaussian.

- Total VAE Loss

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}}$$

  Combined loss provides a single optimization objective and balances reconstruction fidelity against latent regularization.

These evaluation metrics and their importance were published by Doersch et al. [7]

In this project, the main goal is to get the best quality reconstructions, but with the above-mentioned metrics, we can quantify not just reconstruction fidelity, but compression strength and latent regularization.

3.3.4 Model Selection

After the evaluation of total VAE loss $\mathcal{L}$, reconstruction loss $\mathcal{L}_{\text{rec}}$, and KL divergence $\mathcal{L}_{\text{KL}}$ for all six architectures at both latent dimension setups, MP4 version was chosen with $d_z = 2$ (see Figure 4 through Figure 18). On the other hand, BN4 model achieved the lowest numerical loss at $d_z = 20$, but MP4 at $d_z = 2$ has almost equivalent performance with fewer trainable parameters. This is the reason for the decision, hence making a favorable trade-off between compactness and accuracy as stated by Burgess et al. [3]

3.3.5 β-VAE Experiments

In standard VAEs, reconstruction and regularization terms are balanced equally, which can lead to entangled latent representations and suboptimal disentanglement of factors. To solve this problem, a β-VAE introduces a multiplier β on the KL divergence term, modifying the loss to

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \beta * \mathcal{L}_{\text{KL}}$$

Traditionally, increasing β above 1 emphasizes the regularization term and encourages the latent distribution to adhere more strictly to the Gaussian prior, causing more disentangled and interpretable latent factors. On the contrary, setting β below 1 relaxes the prior constraint, consequently improving reconstruction at the cost of the latent structure.
In this project, the goal is to have a compact, Gaussian-like latent space and a high-quality reconstruction on the given microscopy datasets. Thus, tuning the β value allows us to find the optimal trade-off point ([12]). Experiments were made using a range from β = 0.1 (nearly unregularized) up to β = 10 (strong regularization) to determine which setting is best for our expectations.

3.3.5.1 Experimental Setup

- Base Model: Previously selected MP4 architecture with latent dimension $d_z = 2$.
- β Values: $\{0.1, 0.5, 0.8, 1.0, 1.5, 4.0, 10.0\}$.
- Training parameters: As it was used in Section 3.1.2

3.3.5.2 Evaluation Metrics

Model performance was compared using the following metrics for each different β value: Reconstruction Loss ($\mathcal{L}_{\text{rec}}$): Using MSE with sum reduction. KL Divergence ($\mathcal{L}_{\text{KL}}$): Scaled by β in the total loss. Total Loss ($\mathcal{L} = \mathcal{L}_{\text{rec}} + \beta\,\mathcal{L}_{\text{KL}}$) Empirical Latent KL: After training, in the latent space, the KL divergence is calculated from $\mathcal{N}(0, I)$ and observes this value.

- Low empirical latent KL (close to zero) value indicates that the latent distribution closely matches the prior, which means good regularization and potential disentanglement.
- High empirical latent KL value means that the model diverges from the prior, and it can cause latent collapse, entanglement, or overfitting.

3.4 Phase 2: Implementing Variational Autoencoder using Lung Cancer Data

3.4.1 Dataset Description

This dataset consists of live-cell imaging experiments of lung cancer cells exposed to a growth inhibition treatment (EGFR inhibition) and tracked for 3 days [9]. The cells are labeled with a FRET biosensor that measures the activity of ERK/MAPK, a signaling pathway that is thought to be involved in cancer proliferation. In this experiment, some cells immediately died, while others continued to proliferate, and the reasons for this remain unknown.

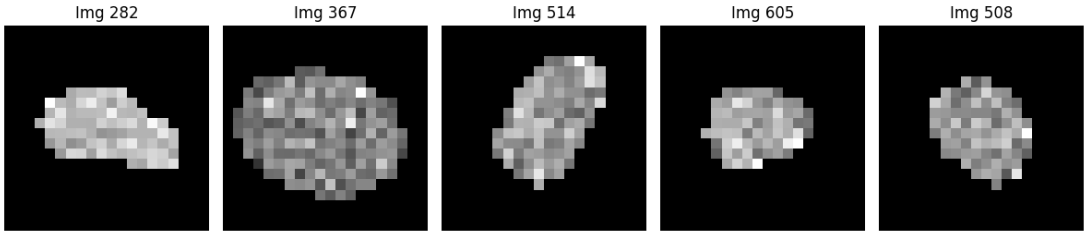


Figure 2: Randomly selected memory lung cancer cells from the dataset.

3.4.2 Model Architecture Experiments

In the previous section (subsection 3.3.2), we showed that the MP4 model is able to reconstruct good-quality images, and now the main focus is to find a model that can achieve similar results in the case of the lung cancer data. For this purpose, the MP4 model was used as a baseline. However, it is important to take into consideration the size of the new dataset because now the model has to face a much smaller input size (20×20 pixels). In this case, the original four-layer downsampling is not directly applicable because it causes the spatial dimension to collapse too aggressively [10].
To get around the problem, the core structure of the MP4 model was used. During the experiments, all the models kept kernel sizes, padding, activation functions, and the general encoder-decoder design. Our approach is to systematically reduce the number of downsampling layers and vary the number of output channels in the deepest encoder layer. The primary goal was to find an architecture that is minimal while providing reliable reconstructions without unnecessary computational overhead.
Talking about the training environment, all models were trained for 100 epochs using the same optimizer and loss metrics as before:

- Reconstruction Loss: $\mathcal{L}_{\text{rec}}$ - MSE with sum reduction
- KL Divergence: $\mathcal{L}_{\text{KL}}$ - Regularization term
- Total Loss: $\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}}$

Unlike before, the latent dimension was fixed, and the chosen size was 2 for all experiments. Since the dataset has 20×20 images, the goal is to capture just the most meaningful latent feature while avoiding the risk of overfitting to noise [1]. Furthermore, using only two dimensions also aligns with the core objective of this project, performing effective dimensionality reduction.
Experimenting with the lung cancer dataset, the following models were used:

- `VAE_MP4`: Baseline from MS experiments (section 3.3), using a four-layer architecture
- `VAE_MP3_128` and `VAE_MP3_256`: Models, using a three-layer architecture and different output channels in the deepest layer (128 versus 256)

- `VAE_MP2_64` and `VAE_MP2_128`: Models, using a two-layer architecture and different output channels in the deepest layer (64 versus 128)

For the following sections, you can find all the details for each model in Appendix A.

3.4.2.1 `VAE_MP4` *Architecture*

This is the model that was used in Section 3.1 for the MS dataset. It consists of four downsampling layers, which reduce the 20×20 input down to a 1×1 spatial feature map. However, using a big and complex model like this on such a small dataset causes aggressive compression, making it unsuitable in this context. With this approach, training became unstable, reconstructions were almost uniform, and the model failed to capture the relevant features of the images. Consequently, this architecture was excluded from further evaluation in the context of lung cancer.

For the next sections, similarly in subsection 3.3.2, for each experienced model, you can find further details collected in Appendix A.

3.4.2.2 `VAE_MP3_128` *Architecture*

This version uses a three-layer structure, reducing the 20×20 input to a 3×3 feature map in the encoder part supplemented with 128 output channels at the bottleneck. Similarly, in section 3.3, the decoder mirrors this structure with three transposed convolutional layers.

3.4.2.3 `VAE_MP3_256` *Architecture*

This model uses exactly the `VAE_MP3_128` structure but doubles the number of output channels in the deepest layer to 256. This means an increased capacity that allows the latent space to hold more representational power and potentially capture more fine-grained differences between samples. This approach can produce slightly better reconstruction quality, but it requires more memory and longer training time.

3.4.2.4 `VAE_MP2_128` *Architecture*

This variant uses only two layers, including Max Pooling. It means that the model compresses the input to a 5×5 feature map, which provides more spatial details than the MP3 approaches. At the deepest point, this model uses 128 channels, producing stable convergence and relatively sharp reconstructions.

3.4.2.5 `VAE_MP2_64` *Architecture*

This is the lightest architecture that was tested, using two Max Pooling layers and only 64 channels at the bottleneck. Even though it is the simplest model,

its performance is comparable to deeper and wider variants in terms of reconstructions, while offering improvements in training speed and parameter efficiency.

3.4.3 Model Selection

After evaluating the models on the lung cancer microscopy dataset, the `VAE_MP2_64` model was selected as the final model. While the other variants (`VAE_MP2_128`, `VAE_MP3_128`, `VAE_MP3_256`) achieved lower loss values, in the case of reconstruction, the differences were hardly noticeable (see Figure 29 to Figure 35). Since `VAE_MP2_64` had the lowest number of trainable parameters (see Table 9 to Table 12) and computational overhead, it provided the best balance between simplicity and performance.

3.4.4 β-VAE Experiments

Similarly, as discussed in subsection 3.3.5, the difference between the traditional VAEs and β-VAEs is that the latter introduces a regularization weight on the KL divergence loss term. This additional step allows controlled disentanglement and shaping of the latent space. This approach prioritizes balancing good quality reconstruction and enforcing a meaningful latent representation. Previously, in subsubsection 3.3.5.1 and subsubsection 3.3.5.2, β-value experiments were observed on the MS dataset, showing the trade-off, especially in terms of latent Gaussianity and empirical KL divergence.

In case of the lung cancer dataset, similar experiences were executed on the selected `VAE_MP2_64` model. This choice was made based on the architectural analysis highlighting this model's balance between performance and simplicity. β-VAE experiments were applied only to the selected model, since the focus was to refine the latent regularization properties and improve the performance of this model.

3.4.4.1 Experimental Setup

The experimental setup closely followed the setup used in subsubsection 3.3.5.1. Compared to the experiments with the MS dataset (subsubsection 3.3.5.1), in this case, the smaller input size (20×20) and the lower model complexity resulted in much shorter training times, making it possible to extend the range of β values. Since the model works with smaller image sizes, we decreased the number of epochs to 50 because after 50 epochs the model gives back proper results.

- Latent dimensionality: $d_z = 2$
- Training parameters: As it was used in subsection 3.3.2
- β Values: 0.1, 0.5, 0.8, 1, 1.5, 2, 5

The chosen β values presented the effects of low and high regularization terms. With these experiments, the goal was to observe how KL divergence behaves across a range of β values and how well the latent space remains Gaussian while maintaining reconstruction capability.

Using the lung cancer data, the evaluation metrics remain the same as described in subsubsection 3.3.5.2. These include the reconstruction loss ($\mathcal{L}_{\text{rec}}$), the KL divergence ($\mathcal{L}_{\text{KL}}$) and the empirical latent KL divergence (KL_{emp}).

3.5 Phase 3: Implementing an Adaptable Variational Autoencoder

One of the main focuses in this project is the development of an adaptable VAE. To achieve this, two separate architectures were created using different datasets - one for high-resolution MS microscopy images (140×140) and another for significantly smaller lung cancer images (20×20). To avoid the need for dataset-specific architectures, the main goal is to create a VAE framework that is able to dynamically adjust its structure based on the spatial resolution of the input. This approach enables easier model reuse and scalability across different microscopy imaging tasks, where input sizes often vary. During the implementation, the dimensionality of the latent space was fixed at two to maintain interpretability and enforce a consistent bottleneck regardless of input size.

3.5.1 Architecture Design

Based on the previous experiments, the primary goal of the adaptable VAE framework is to show that a VAE model can be successfully structured by having the same architecture in each layer. In this case, with respect to the use of Max Pooling and ReLU activations. This approach means that, by choosing the right number of layers, the model can still achieve good-quality reconstructions.

Building up the best model, the number of layers in the model is the primary variable in our experiments. The goal is to determine it based on the size of the input data. The structure for each layer remains the same, with the following components:

- Convolutional layers utilize (3×3) kernels with stride of 2
- Max Pooling as a next step
- Ending with ReLU activation function

This setup makes it possible for the model to have a uniform structure regardless of the input size. The significant difference lies in the number of layers in the encoder and decoder components, which are adjusted based on the input size [27].

3.5.2 Number of Layers Selection

The right number of convolutional layers was determined based on the input image size. Previously, in section 3.3 and section 3.4, the experiments have shown that the MS dataset (140×140 pixel images) performed best with four convolutional layers, while the lung cancer dataset (20×20 pixel images) achieved appropriate results even with only two layers. Knowing these concrete values, these two points can define a generalizable approach for setting the number of layers in a VAE. Our approach uses linear interpolation for this calculation.

3.5.2.1 Linear Interpolation

Linear interpolation is a numerical technique used to estimate values that lie between two known data points. Given two known points (x_0, y_0) and (x_1, y_1), the interpolated value y at a position x between x_0 and x_1 can be calculated with the following formula:

$$y = y_0 + (x - x_0) \cdot \frac{y_1 - y_0}{x_1 - x_0} \tag{3.1}$$

This approach assumes that the variables' relationship is approximately linear and can be used to interpolate unknown outcomes.

In this project, the x-axis corresponds to the logarithm base 2 of the total number of pixels in the input image, while the y-axis represents the number of convolutional layers. This calculation makes it possible to adaptively estimate the number of required layers for images of arbitrary sizes.

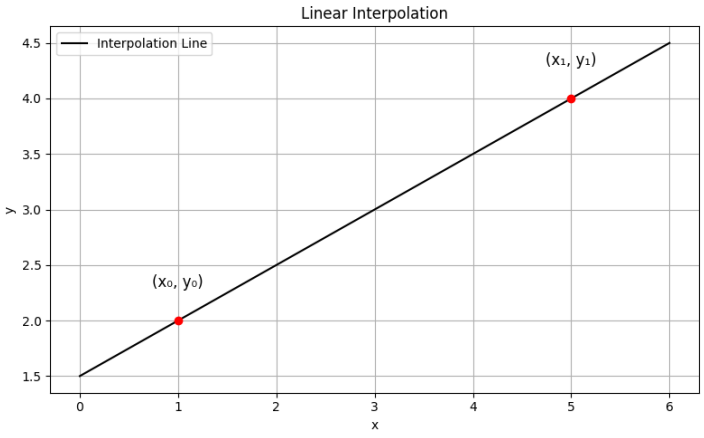


Figure 3: Linear interpolation between two anchor points (x_0, y_0) and (x_1, y_1).

3.5.3 Validation

To test the generalization feature of the adaptable VAE framework, we validated it on a new resolution image dataset that was not seen during the devel-

opment. Specifically, we used a lung cancer dataset, and with augmentation, it was resized from the original 20×20 images to 80×80. This size was chosen because it is a common standard in microscopy imaging and also provides a good opportunity to try something in the middle between the two previously used image sizes (140×140 and 20×20). The main purpose of this validation was to check whether the adaptable model could generalize and choose the proper number of convolutional layers using the size of the input data. To ensure that the calculated three-layer architecture is optimal, two additional baseline models were trained with two and four convolutional layers on the same 80×80 input data. This comparison helps assess whether three layers truly provide the best trade-off between under- and over-compression at this intermediate resolution.

3.5.3.1 Setup

For the validation, the following values were used:

- Dataset: Lung cancer dataset resized to 80×80 pixels.
- Latent dimensionality: $d_z = 2$
- Training epochs: 20 epochs.

After training, the adaptable model was evaluated using the same metrics as in the previous experiments. Including the following:

- Reconstruction Loss: $\mathcal{L}_{\text{rec}}$ - MSE with sum reduction
- KL Divergence: $\mathcal{L}_{\text{KL}}$ - Regularization term
- Total Loss: $\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}}$

These metrics are used to evaluate whether the number of layers determined through linear interpolation can still produce stable and high-quality reconstructions. To strengthen this evaluation, the same training setup was used on two-layer and four-layer variants, designed manually. These controlled tests allow us to determine if the interpolated three-layer model offers the most efficient and effective reconstruction for the 80×80 resolution, and whether simpler or deeper architectures might outperform it.
Furthermore, the model's validation helps to verify that the adaptable architecture can be effective when applied to different image sizes, even if they are outside the original training configurations.

4 Results

This chapter presents the outcomes of the experiments made in chapter 3. With the use of quantitative and qualitative metrics, the focus is on understanding how well these models can compress microscopy images into low-dimensional latent spaces without losing critical structural information.
One of the challenges is to minimize the number of layers required for meaningful reconstructions, thus getting an efficient architecture. Therefore, the goal is to identify how simple a model can be while still producing good-quality reconstructions, and to explore whether a single framework can be used for different datasets without significant manual tuning.
The results are organized into three parts. First, we analyze the MS dataset to evaluate the effect of different architectural depths and latent dimensionalities, followed by an investigation of the β parameter. Second, we examine how the selected model behaves on low-resolution lung cancer data and determine the most efficient architecture for this setting. Finally, we test the adaptable VAE design that automatically scales its depth based on input image size, validating its generalization to unseen resolutions.

4.1 Phase 1: Results on Multiple Sclerosis Dataset

4.1.1 Comparison of Model Architectures and Latent Dimensionalities

To compare the VAE models, we trained six different architectures using two different latent dimensionalities ($d_z = 2$ and $d_z = 20$) on the MS dataset. The models were introduced in subsection 3.3.2.

4.1.1.1 Quantitative Evaluation

This part focuses on the quantitative comparison of model performance across the six tested VAE frameworks. The evaluation is based on three key loss metrics:

- Reconstruction Loss: $\mathcal{L}_{\text{rec}}$ - MSE with sum reduction
- KL Divergence: $\mathcal{L}_{\text{KL}}$ - Regularization term
- Total Loss: $\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}}$

These metrics provide an explanation of how well each model balances accurate image reconstruction with regularization of the latent space. Using this quantitative approach, the goal is to identify which architectural design most effectively compresses the input data while preserving the most important

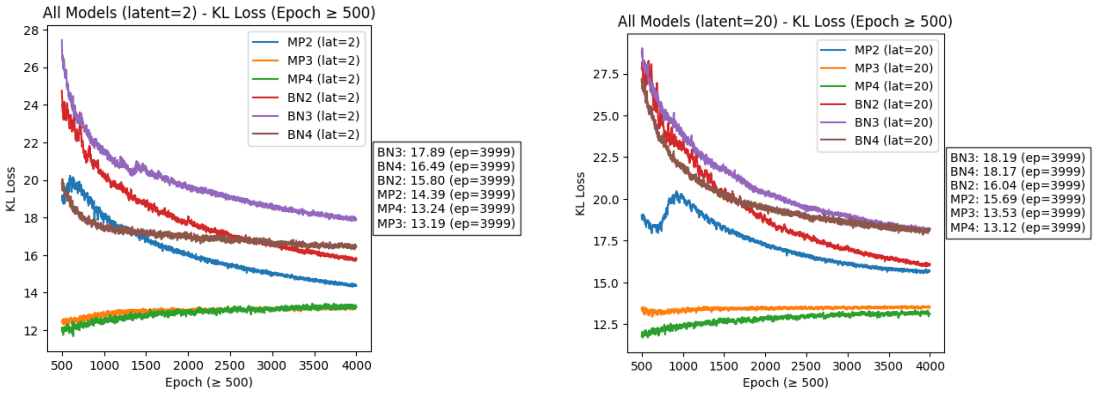
structural information.
For easier interpretability, the plotted loss curves are shown from epoch 500 onward, which helps to skip early instability and allows a more focused analysis of convergence, since the range of the y-axis is smaller. On top of that, each figure includes the final loss values at epoch 3999, sorted in descending order. This helps to compare models when the lines visually overlap or converge closely on the plots.

KL Divergence Loss

Figure 4a and Figure 4b show the KL divergence losses for all six VAE models trained with latent dimensionalities of $d_z = 2$ and $d_z = 20$. This metric presents how closely the learned latent distributions approximate the standard normal prior.

Observing a latent space of $d_z = 2$ (Figure 4a), the lowest final KL loss value was achieved by the MP3 model (13.19), closely followed by MP4 (13.24). MP2 performed slightly worse, achieving 14.39 in the end. On the other hand, using Batch Normalization in variants BN2-BN4 showed higher KL values, going up from 15.80 to 17.89, indicating looser regularization.

Evaluating the higher dimensionality, $d_z = 20$ in Figure 4b, KL divergence increased overall due to the expanded latent capacity. The increased dimensionality yielded similar results and patterns. MP4 and MP3 models remained the most regularized, with final 13.12 and 13.53 values, respectively. These models were followed by MP2 and its 15.69 value at the end, while the Batch Normalization models again trailed behind. Their final KL loss values were moving in the range of 16.04 to 18.19.



(a) KL divergence loss with latent dimension 2

(b) KL divergence loss with latent dimension 20

Figure 4: KL divergence loss across training epochs for all models. Final values are included in descending order for clarity.

Reconstruction Loss

Figure 5a and Figure 5b present the reconstruction loss across training

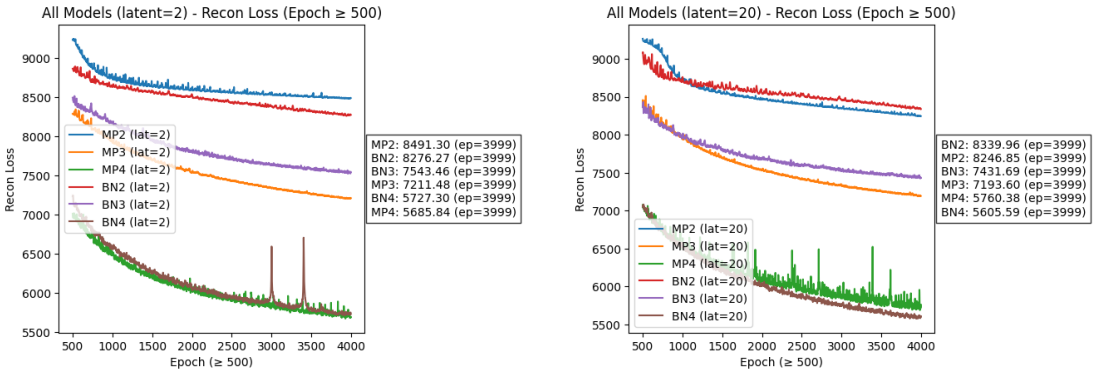
epochs for all the VAE models, using latent dimensionality $d_z = 2$ and $d_z = 20$. This metric shows how accurately the models reproduce the input microscopy images from their compressed latent representations.

Evaluating $d_z = 2$, MP4 achieved the lowest final reconstruction loss with 5685.84 followed closely by BN4 and its 5727.30 value. MP3 achieved 7211.48 in the end, while MP2 and BN2 were significantly worse, with final values 8491.30 and 8276.27, respectively. BN3 took place in the middle section, converging to 7543.46. These results suggest that increasing model depth improves reconstruction, and that using MaxPooling can mean better results than Batch Normalization variants when comparing across similar depths.

At $d_z = 20$, the results are similar to what was seen in the case of $d_z = 2$. MP4 and BN4 still have the smallest final reconstruction loss values, ending at 5760.38 and 5605.59, respectively. This means that changing dimensionality switched these models' order. MP3 and BN3 are taking place in the middle with their 7193.60 and 7431.69 values, respectively. Furthermore, MP2 and BN2 continued to lag behind (8246.85 and 8339.96).

Although the training did not show full convergence at epoch 3999, the most significant loss reduction occurred during the earlier stages. After that, the loss curves largely flattened out, and no visually meaningful differences were observed in the reconstructed images beyond that point. Hence, extending training would likely not have yielded significant performance improvements.

These findings support the intuition that deeper architecture designs capture more image details and structure, improving the reconstruction quality. At the same time, it shows one of the key concepts of the thesis, that is, the importance of balancing depth against complexity.



(a) Latent dimension = 2

(b) Latent dimension = 20

Figure 5: Reconstruction loss over training for all models. Final values at epoch 3999 are listed in descending order to aid readability.

Total Loss

Figure 6a and Figure 6b show the total VAE loss with the previous setup.

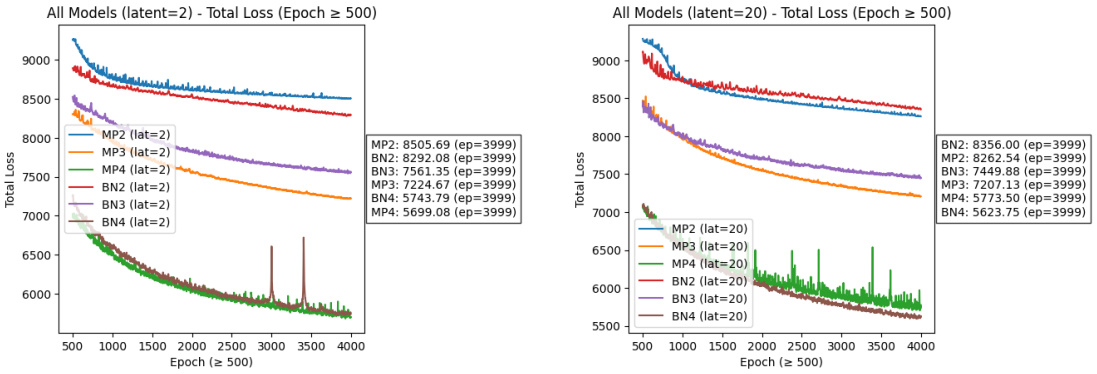
This combined metric captures both reconstruction quality and latent regularization, as defined in the following way:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{KL}}.$$

At $d_z = 2$, MP4 achieved the lowest final total loss (5699.08), slightly outperforming BN4 (5743.79). MP3 is the next in the line with its 7224.67 value, and the rest of the models (BN3, BN2, MP2) go above 7500. These results confirm earlier findings that deeper models, particularly those using MaxPooling, find a better balance between accurate reconstructions and effective latent space compression.

On the other hand, at $d_z = 20$, the results show a consistent gap between the models' loss values but with a switch again. In this setup, BN4 achieved the lowest total loss (5623.75 following by MP4 and it 5773.50 value. Changing the number of layers makes a significant change in the loss values. MP3 and BN3 considerably higher with their 7207.13 and 7449.88, while MP2 and BN2 continued to underperform, both remaining above 8200.

In the case of the total loss, KL term contributes less than the reconstruction loss numerically. This is the reason why total loss trends closely follow the observations from the reconstruction loss section. This is the reason why the loss curves had not fully converged by epoch 3999. Although the dominant performance differences had already materialized, further training would not likely change the ranking. Therefore, total loss remains the clearest summary indicator of each model's performance in balancing compression and accuracy.



(a) Latent dimension = 2

(b) Latent dimension = 20

Figure 6: Total VAE loss over training for all models. Final values at epoch 3999 are included in descending order in the legend to assist visual comparison.

4.1.1.2 Qualitative Evaluation

After the quantitative evaluation in subsubsection 4.1.1.1, the second approach was to perform a qualitative evaluation of the reconstructed images generated by the various VAE architectures. The setup still remains, and each model was trained for 4000 epochs. The reconstructions are evaluated visually to assess

the preservation of the biologically relevant morphology and structural fidelity across both data channels.
In this qualitative approach, we include both channels of the original dataset. The representative examples were selected from both the NTZ (healthy control) and BC (patient) groups. For each model, the results are placed in a grid structure showing the input and reconstructed outputs. The first two rows represent the original input images for each channel, while the bottom two rows show the corresponding reconstructions.
In case of BN2, the reconstructions of the second channel are smooth and preserve the outer morphological ring structure well, but the finer cytoplasmic details in the first channel are blurred out more. This model lacks capacity for finer granularity and using an increased latent dimensionality ($d_z = 20$ instead of $d_z = 20$), improves the performance slightly.

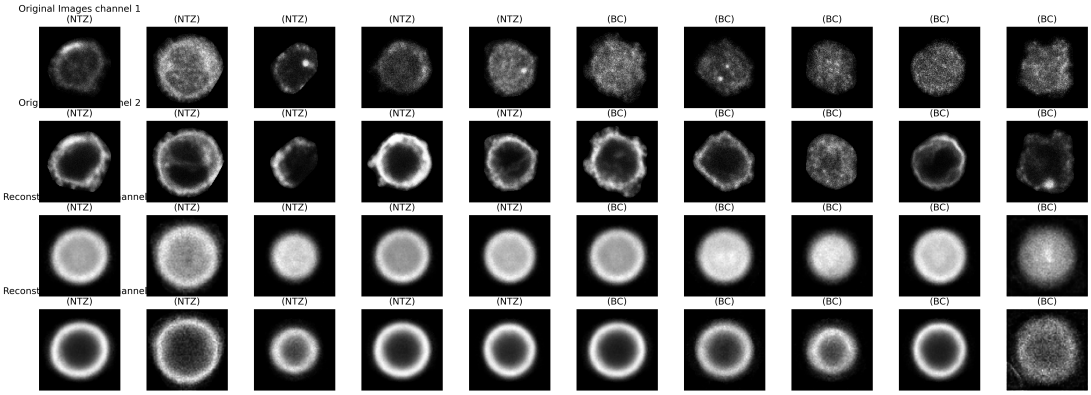


Figure 7: Reconstructions using BN2 architecture with latent dimension 2.

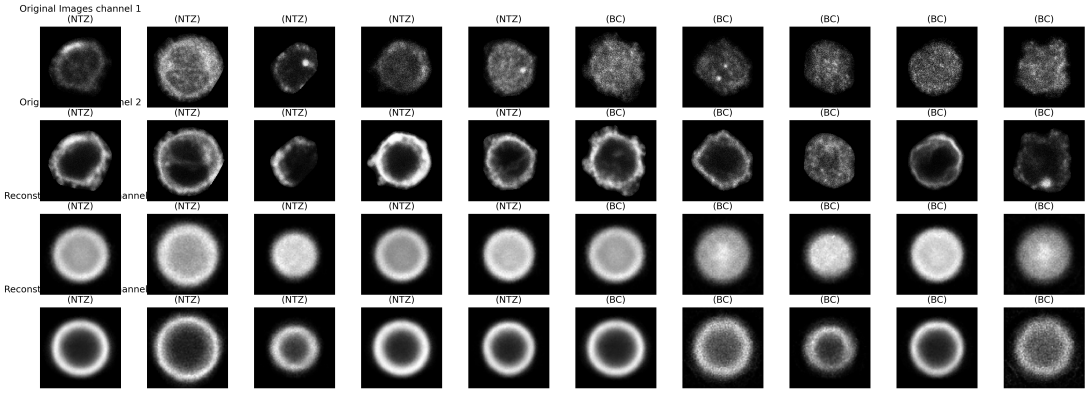


Figure 8: Reconstructions using BN2 architecture with latent dimension 20.

Observing the BN3 model, it shows sharper reconstructions than BN2. Channel 1 reconstructions capture more localized contrast. Also, increasing the latent dimensions from $d_z = 2$ to $d_z = 20$ improves the representation of the background, which means that the dimensionality increase added latent capacity benefits.

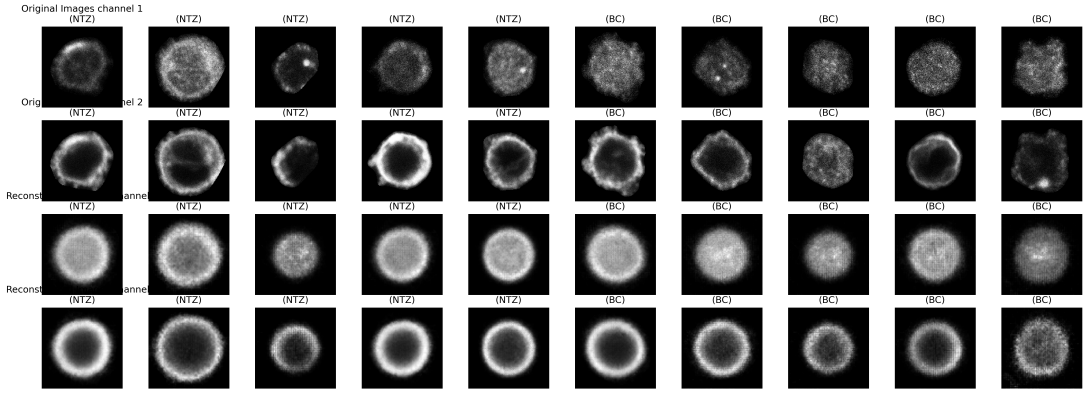


Figure 9: Reconstructions using BN3 architecture with latent dimension 2.

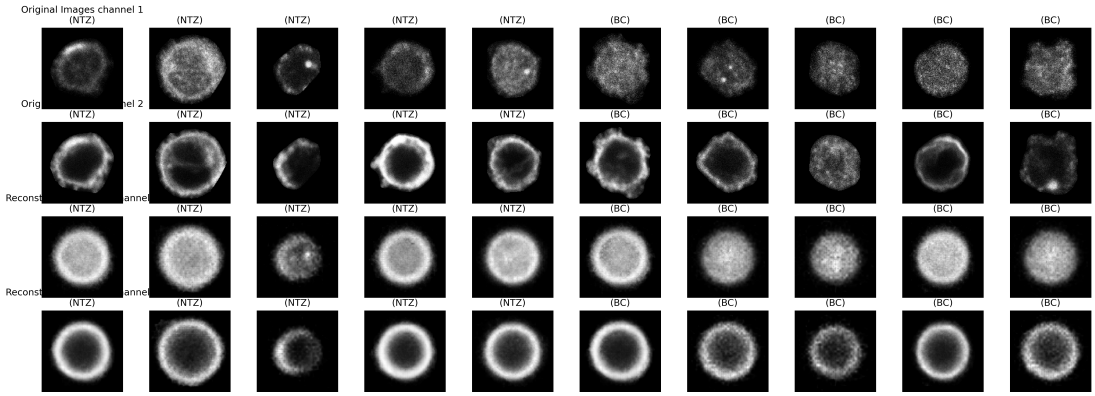


Figure 10: Reconstructions using BN3 architecture with latent dimension 20.

BN4 model shows stable reconstructions across both channels. It maintains a consistent representation of the circular segmentation stain in Channel 2. However, we can see that Channel 1 reconstructions show signs of over-smoothing in the $d_z = 2$ latent space. Using $d_z = 20$ latent space, reconstructions become slightly sharper and better capture the dense textural variance in patient cells as a result of increased background granularity.

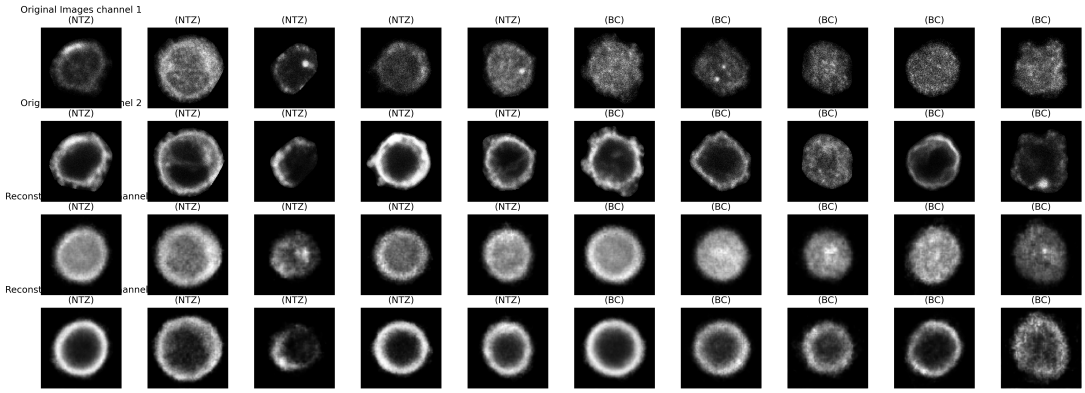


Figure 11: Reconstructions using BN4 architecture with latent dimension 2.

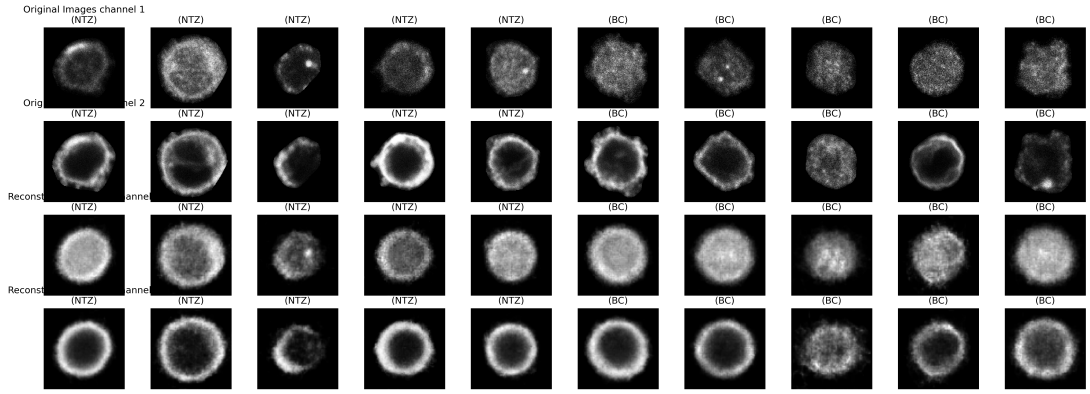


Figure 12: Reconstructions using BN4 architecture with latent dimension 20.

Moving to our other approach, where the Batch Normalization is changed to MaxPooling, first, we analyze the MP2 model. This architecture produced some of the clearest ring structures in Channel 2 across all configurations. However, Channel 1 is simplified way too much. However, using $d_z = 20$ instead of $d_z = 2$ improves the reconstructions in terms of reconstruction sharpness and the recovery of more diverse textures.

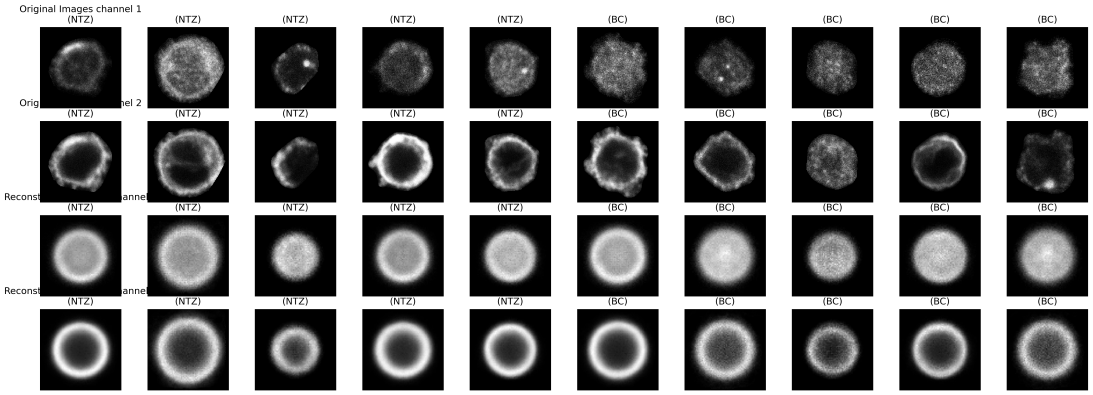


Figure 13: Reconstructions using MP2 architecture with latent dimension 2.

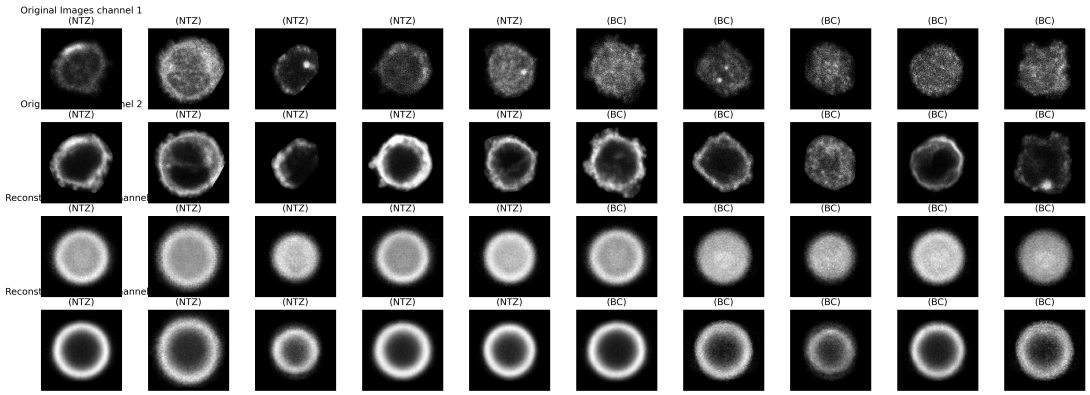


Figure 14: Reconstructions using MP2 architecture with latent dimension 20.

Moving forward to the MP3 model, using only $d_z = 2$ crisp morphological rings and moderately complex reconstructions of the cytoplasmic, are already seen after training. At $d_z = 20$, the model presents similar results, preserving fine texture and variability across both channels.

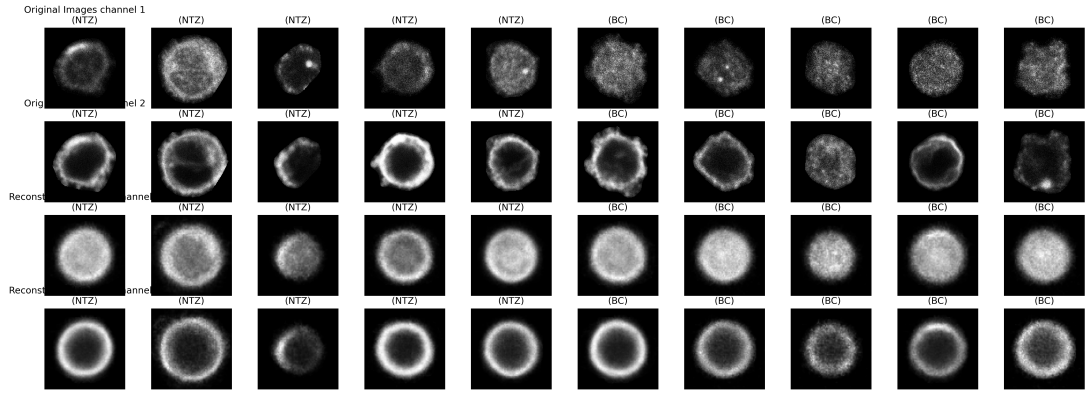


Figure 15: Reconstructions using MP3 architecture with latent dimension 2.

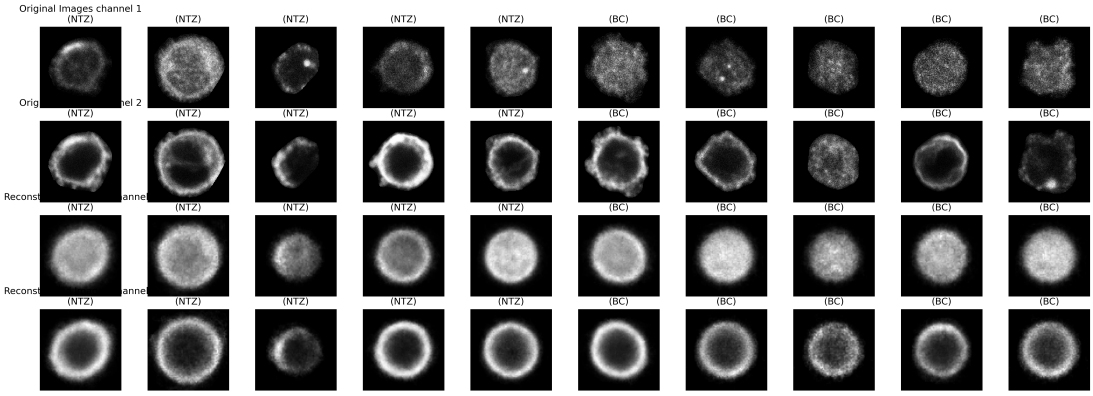


Figure 16: Reconstructions using MP3 architecture with latent dimension 20.

Last but not least, the reconstructions made by MP4 model show strong fidelity in Channel 2 across the dataset. In case of $d_z = 2$, the reconstructions appear blurry in Channel 1; however, at $d_z = 20$, reconstructions are visibly richer. It provides high interpretability and stable morphological structures.

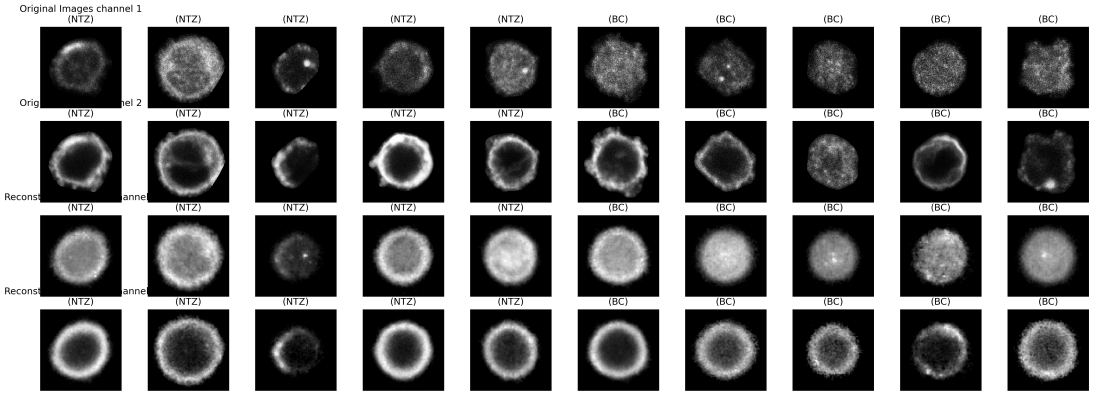


Figure 17: Reconstructions using MP4 architecture with latent dimension 2.

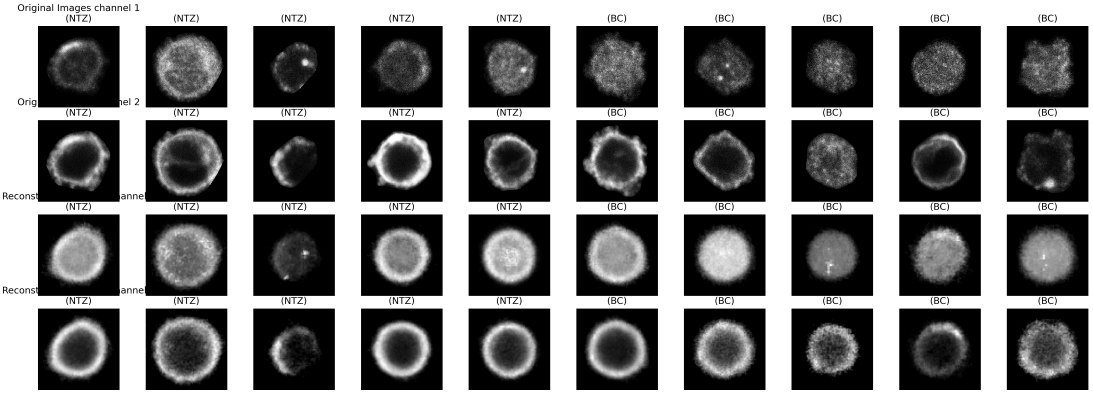


Figure 18: Reconstructions using MP4 architecture with latent dimension 20.

4.1.1.3 Model Selection

The final model was selected based on the quantitative (subsubsection 4.1.1.1) and qualitative (subsubsection 4.1.1.2) evaluations. In the end, the chosen model was MP4 using $d_z = 2$ for further work. This decision was made based on the balance between performance, simplicity and alignment with prior methods.

In case of qualitative results, MP4 achieved better results than BN4 in terms of structural clarity and morphological consistency. Both models have a 4-layer deep structure, but using MaxPooling provided better robustness in capturing details.

For further development, the latent space size was decided to $d_z = 2$. Although suing $d_z = 20$ yielded richer reconstructions in some cases, they also introduced complexity and potential overfitting risks without presenting a significant gain in qualitative interpretability. For our current goal, using $d_z = 2$ as a latent space representation is not only sufficient but also more practical.

Furthermore, this framework follows the S3-VAE ([4])setup in terms of a number of layers, where a 4-layer encoder/decoder structure was employed on microscopy data. By maintaining this structural consistency, we can make com-

parisons with earlier results while improving architectural refinement.
In summary, using MP4 model with a latent dimension $d_z = 2$, we can have a strong trade-off between performance and usability. It ensures reproducibility, interpretable latent space, and good-quality reconstructions across both channels.

4.1.2 Comparison of Different β Values on MP4 Architecture

Having selected the MP4 model with a 2-dimensional latent space as the final architecture, the next approach was to try this framework with different β parameters. This variant modifies the original VAE loss by adding a scaling factor β to the Kullback-Leibler divergence term. This allows control over the trade-off between latent space regularization and reconstruction fidelity.
To observe the impact of β, we trained the MP4 model for seven different values: $\beta = 0.1, 0.5, 0.8, 1, 1.5, 4, 10$. The setup was the same as before; each model was trained for 4000 epochs, and the results were evaluated using three key perspectives:

- Loss curves over time
- Latent KL divergence for each dimension
- Visual quality of reconstruction

4.1.2.1 Quantitative Evaluation

Loss Evaluation

During the quantitative evaluation, we plotted the evolution of losses over training epochs. These changes are shown in Figure 19, Figure 20, and Figure 21. To minimize early training noise, the data is shown after the first 500 epoch.

- KL Loss ($\mathcal{L}_{\mathrm{KL}}$) shows an increase with higher β as expected because of a stronger regularization pressure. While $\beta = 0.1$ reaches the lowest KL loss value around 1.54 in the end, using $\beta = 10$ makes an explosion in this term with its 105.02 final value.
- Reconstruction Loss ($\mathcal{L}_{\mathrm{rec}}$) changes in an inverse way. Choosing lower β values yields the best reconstruction accuracy, especially in the case of $\beta = 0.5$ with a final 5574.81 value. But on the other hand, high β values (4 or 10) decrease the quality of the reconstruction due to over-prioritizing the latent space.
- Total Loss ($\mathcal{L}$) collecting together all the information and in our experiments, it shows that $\beta = 0.5$ consistently achieves the best results, achieving the best trade-off between reconstruction quality and latent structure.

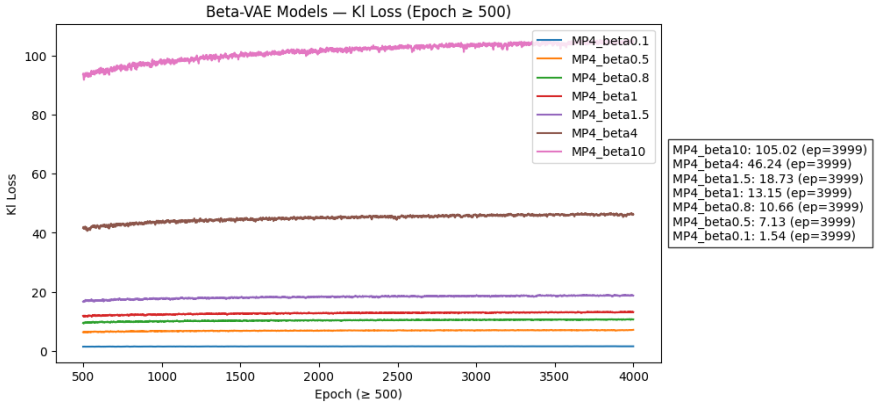


Figure 19: KL loss across epochs ($\geq$500) for different β values.

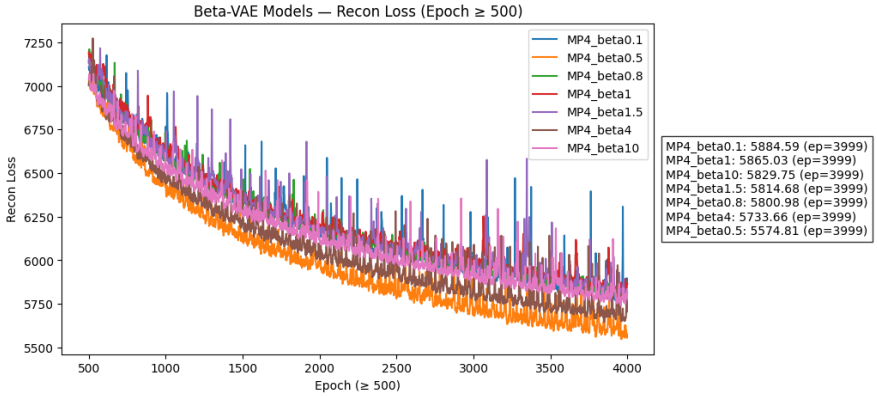


Figure 20: Reconstruction loss across epochs ($\geq$500).

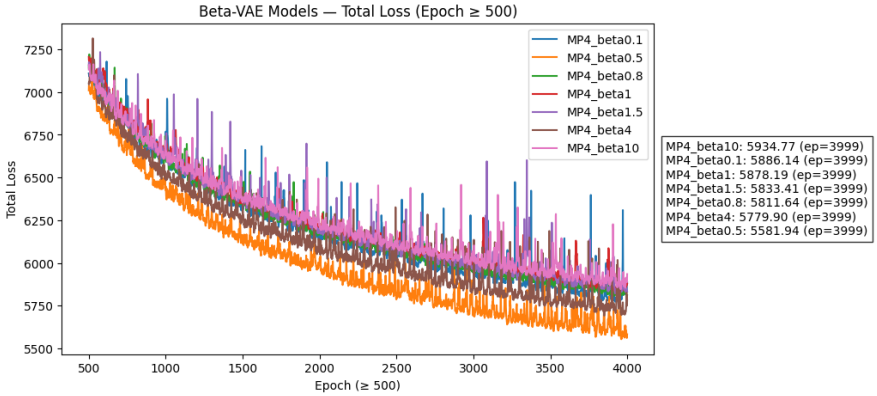


Figure 21: Total loss ($\mathcal{L}_{\mathrm{KL}}$ + $\beta \cdot \mathcal{L}_{\mathrm{KL}}$) across epochs ($\geq$500).

Evaluation of KL Divergence on the Latent Space

Developing VAEs, the KL divergence term plays an important role in shaping the latent space. It quantifies how much the learned posterior distribution

$q(z|x)$ deviates from the prior $p(z)$, typically assumed to be a standard normal distribution. Choosing the right factor for the KL loss ensures that the latent space remains smooth, continuous, and useful for sampling or downstream interpretability.
Our approach is to calculate the KL divergence for each latent dimension, which gives insight into how effectively each dimension contributes to encoding information. If this value is close to zero, this could indicate redundancy or under-utilization. On the other hand, extremely large KL values can suggest over-regularization, where the model sacrifices reconstruction quality to match the prior too aggressively.
In this project, the latent dimensionality was fixed to $d_z = 2$ and we calculated the final KL divergence of each dimension across all tested β values. These results are shown in Table 1.

β	KL (dim 1)	KL (dim 2)
0.1	0.1722	0.2081
0.5	0.1951	0.1834
0.8	0.1608	0.1587
1	0.1755	0.1820
1.5	0.1272	0.1690
4	0.1871	0.1230
10	0.1331	0.1505

Table 1: Final KL divergence values for each latent dimension (at epoch 3999).

The most symmetric result was achieved at $\beta = 0.5$, where both latent dimensions carry comparable levels of information, without collapsing or dominating each other. At $\beta = 085$ and $\beta = 1$, the KL values remain balanced but start to decrease, which could suggest underutilization. However, using high β values, especially $\beta = 4$ and $\beta = 10$, KL values drop significantly, compared to the previous values. It indicates that the prior is being enforced too strongly, reducing the model's expressive capacity.
Evaluating the different β value experiments, this analysis gives an overall conclusion that choosing the right β values, in this case $\beta = 0.5$, promotes a meaningful and balanced use of the latent space, which produces a good balance between loss curves and visual reconstructions.

4.1.2.2 Qualitative Evaluation

To complement the quantitative evaluation, examining the visual representations also plays a key role. This part shows the reconstructions generated by the MP4 model across different β values. Figure 22 through Figure 28 present original and reconstructed image pairs for all tested configurations. Using $\beta = 0.1$(Figure 22), high visual fidelity and sharp detail preservation are shown across both image channels. However, this often means poor latent space regularization, as seen in the KL divergence behavior.
In contrast, going above $\beta = 1$ such as $\beta = 4$ and $\beta = 10$ (Figure 27 and Figure 28) leads to overly smoothed reconstructions where fine morphological

structures are lost. This presents our earlier observation that high β values enforce the prior too strongly at the cost of expressiveness.
The best balance between quality and regularization is achieved at $\beta = 0.5$(Figure 23), where the reconstructions maintain core structural integrity. In subsubsection 4.1.2.1 this setting achieved the most symmetric KL distribution across latent dimensions. In summary, the qualitative evaluation also supports the choice of $\beta = 0.5$ as the optimal setting for further experiments on the MS dataset.

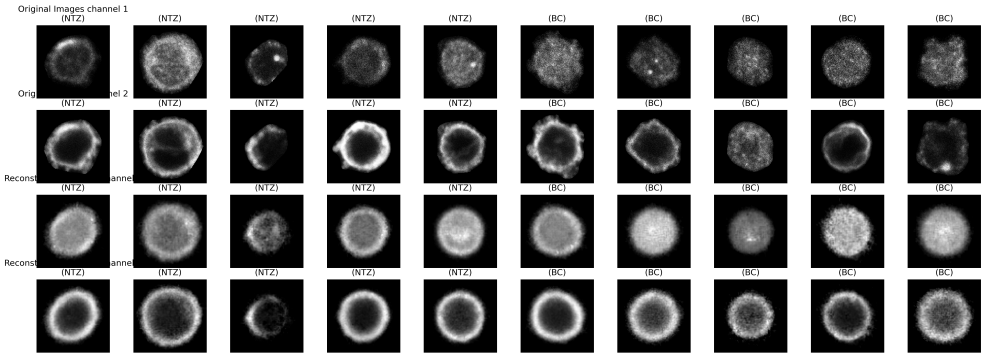


Figure 22: Reconstruction example for the MP4 model with $\beta = 0.1$.

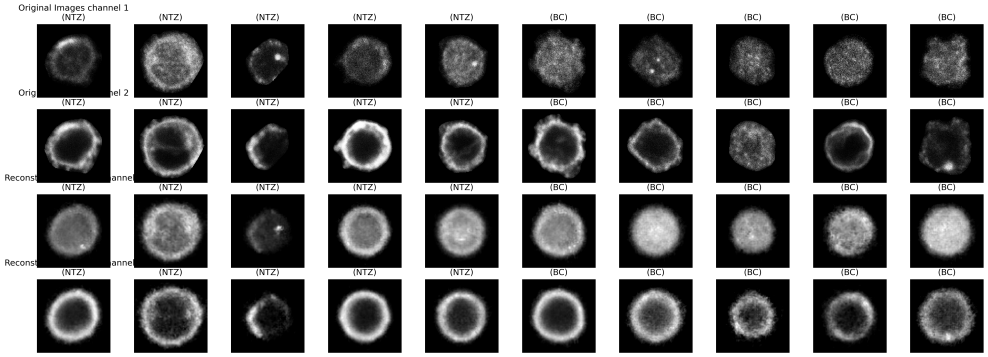


Figure 23: Reconstruction example for the MP4 model with $\beta = 0.5$.

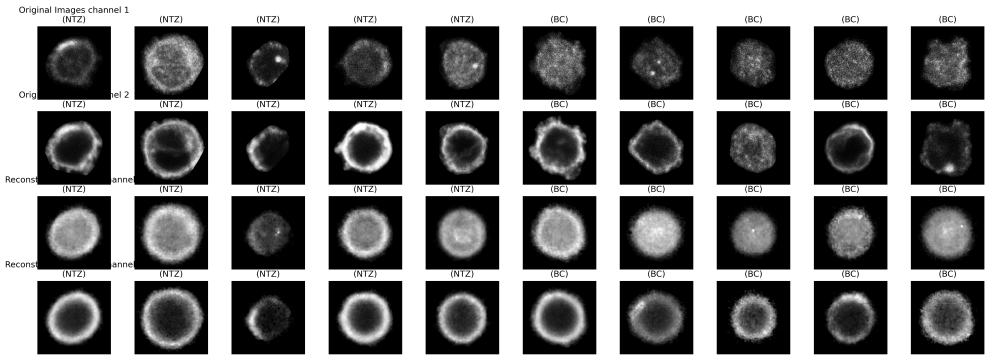


Figure 24: Reconstruction example for the MP4 model with $\beta = 0.8$.

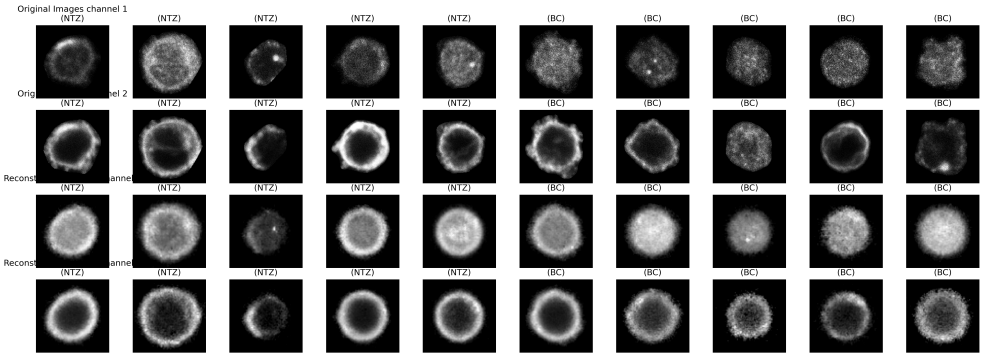


Figure 25: Reconstruction example for the MP4 model with $\beta = 1$.

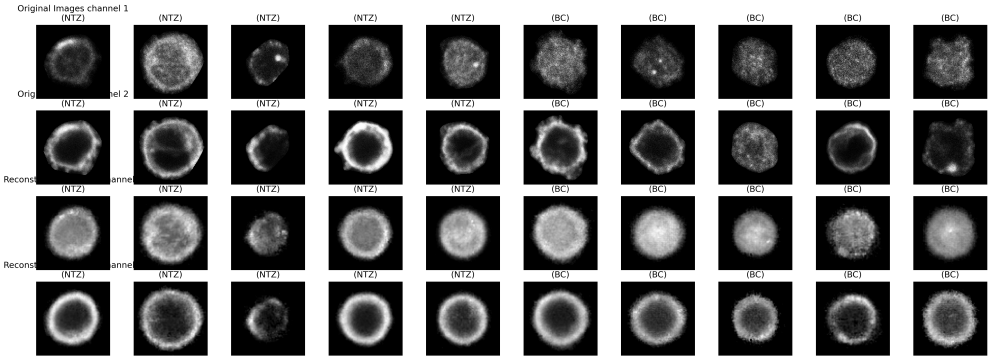


Figure 26: Reconstruction example for the MP4 model with $\beta = 1.5$.

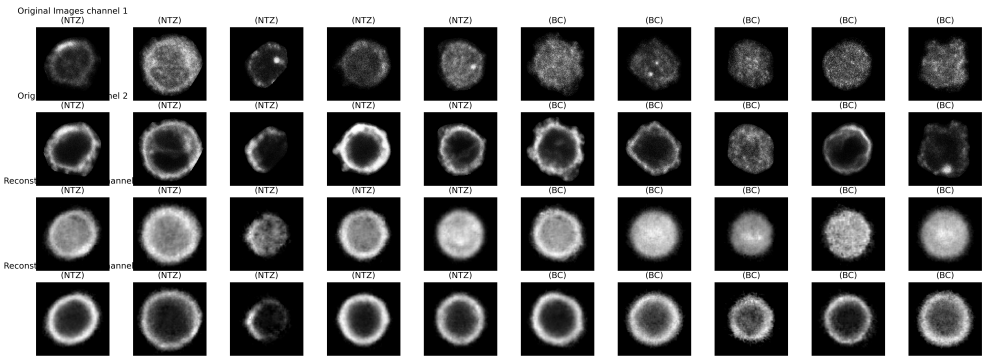


Figure 27: Reconstruction example for the MP4 model with $\beta = 4$.

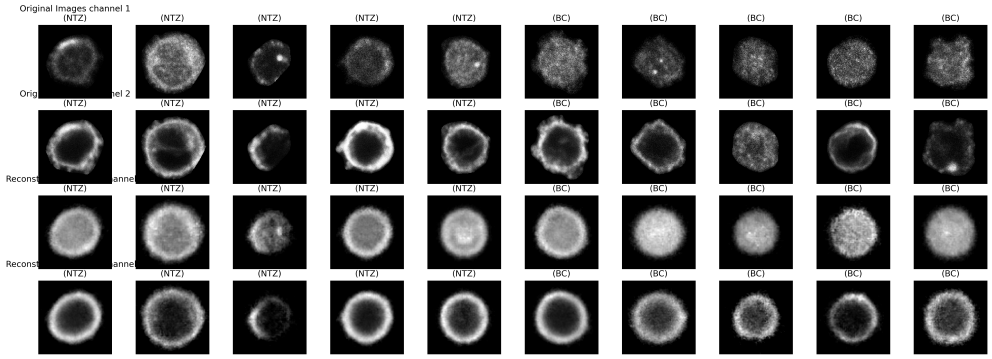


Figure 28: Reconstruction example for the MP4 model with $\beta = 10$.

In summary, the qualitative evaluation also supports the choice of $\beta = 0.5$ as the optimal setting for further experiments on the MS dataset.

4.2 Phase 2: Results on Lung Cancer Dataset

4.2.1 Comparison of Model Architecture

After the model development and analysis on the MS dataset, the final framework was applied to the lung cancer dataset. The goal was to create a model that works well on the new dataset and was made from the MP4 model from subsection 3.3.4. Several differences are observable between the two datasets. Lung cancer dataset consists of single-channel microscopy images with much smaller resolution (20×20). Furthermore, in this setup, the latent dimensionality was fixed at $d_z = 2$ for all experiments to preserve interpretability and enforce compact representations.

4.2.1.1 Quantitative Evaluation

To evaluate the performance of the implemented VAE architectures on the lung cancer dataset, similarly to the MS dataset, the loss metrics from subsubsection 4.1.1.1 were observed. On each plot, the legend includes a list of the evaluated models in descending order based on their final loss value at the last epoch. This provides a clear comparative ranking and makes it easier to understand.

KL Divergence Loss Loss

As shown in Figure 29, all models exhibit a smooth convergence in terms of KL divergence, reaching a stable value, typically after 40-50 epochs. Among them, the final values are relatively close to each other, ranging from 20.77 for `MP2_64` to 22.52 for `MP3_128`. The seen result indicates that all models impose a comparable level of regularization. However, `MP2_64` reaches the lowest value, which might suggest greater expressive flexibility.

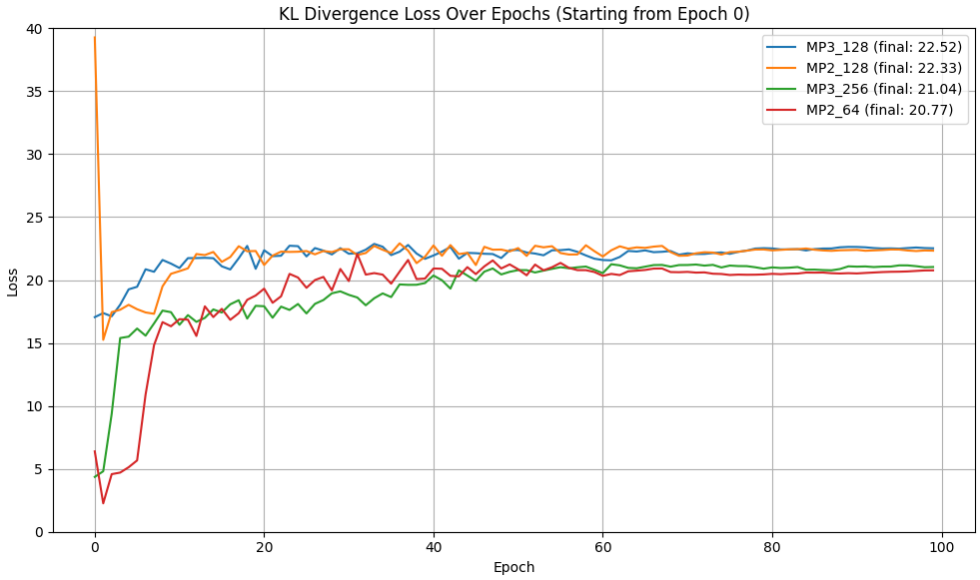


Figure 29: KL Divergence loss curves during training. The legend ranks models by final KL value (descending).

Reconstruction Loss

Figure 30 shows the reconstruction loss, which captures how well each model can reproduce the original input. It is shown that `MP3_128` achieves the lowest final reconstruction loss with 156.85, slightly outperforming `MP2_128` (157.24) and `MP3_256` (159.46). In this case, `MP2_64` is at the end of the list with its 165.95 final value. Since the first three models dispose of more layers and more output channels, it is not surprising that they have greater representational capacity.

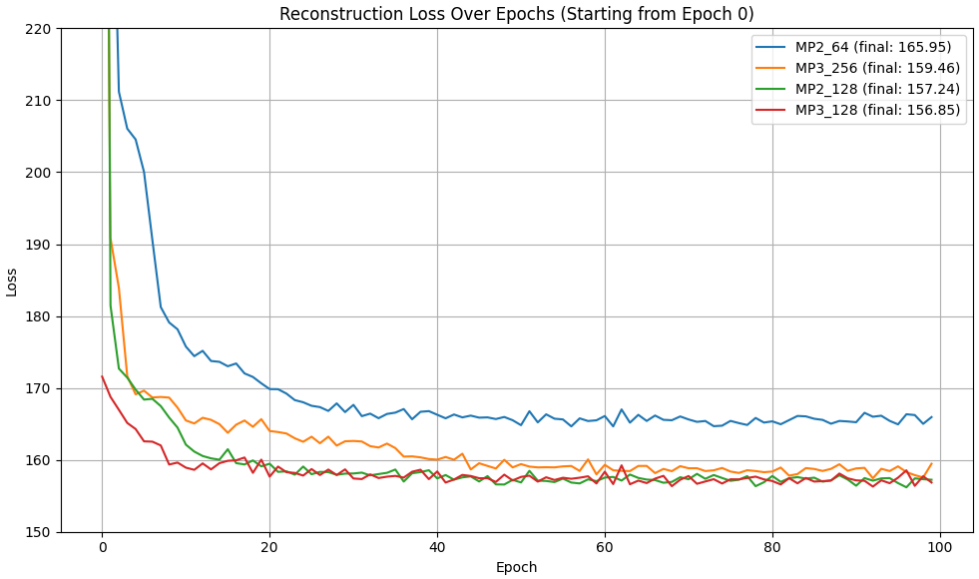


Figure 30: Reconstruction loss curves during training. The legend ranks models by final reconstruction loss (descending).

Total Loss

Figure 31 aggregates the two losses, showing overall model efficiency. All models converge efficiently by around epoch 30. `MP3_128` yields the lowest final value with 179.38. `MP3_256` and `MP2_128` have nearly the same values with 180.50 and 179.57, respectively. Regarding total loss, `MP2_64` ends higher at 186.72, which aligns with its limited reconstruction capacity but also shows its reduced model complexity.

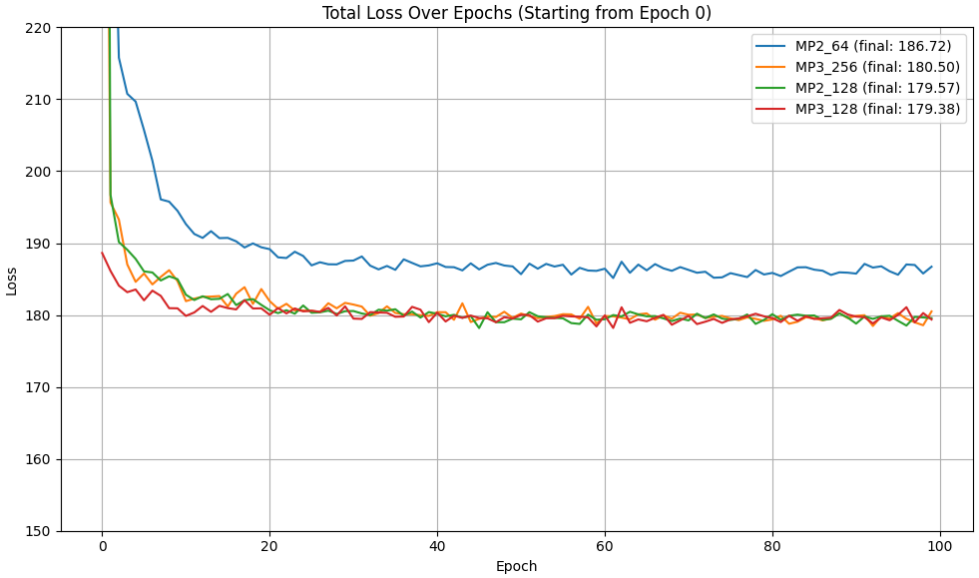


Figure 31: Total loss curves during training. The legend ranks models by final total loss (descending).

Overall, it is seen that deeper and wider models, like MP3_128, slightly outperform in loss metrics, the differences are marginal. These trends show the trade-off between complexity and performance and support the use of MP2_64 in scenarios where computational efficiency is a key concept.

4.2.1.2 Qualitative Evaluation

As we did in section 4.1, in addition to the quantitative analysis, qualitative evaluation was conducted to visually inspect how well each model preserves the structural properties of the lung cancer images. Since the input image size is only 20×20 and consists of a single channel, these reconstructions are expected to retain basic shape features rather than fine-grained cellular texture. The original and reconstructed images are shown in Figure 32 to Figure 35. All models capture the general circular morphology and the intensity gradients typical of the dataset. However, the reconstruction quality varies by architecture.

MP2_64 (Figure 32) shows consistent reconstructions with slightly blurry but stable results. The central region's brightness is captured well, however finer boundaries are smoothed out. It is expected, given the model's simplicity.

MP2_128 (Figure 33) model still uses a two-layer architecture, but it offers improved sharpness over MP2_64, with clearer contours and more distinct edges. However, its increased final channel number means higher model complexity.

MP3_128 and MP3_256 (Figure 34 and Figure 35) produce the most visually precise reconstructions. These models preserve spatial proportions and gradient transitions more faithfully. These models are more expressive but also require more computational capacity.

In summary, deeper models improve visual fidelity, but even the minimal MP2_64 model is capable of retaining the essential morphological patterns.

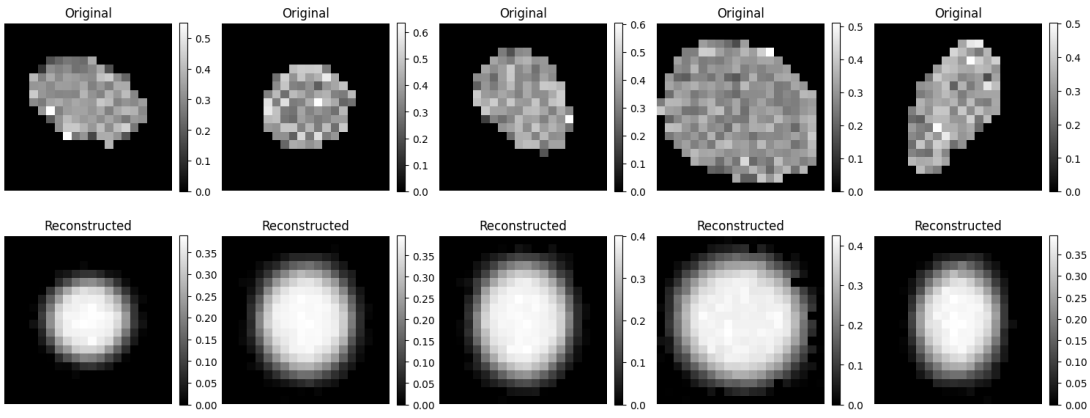


Figure 32: Original and reconstructed images from the MP2_64 model.

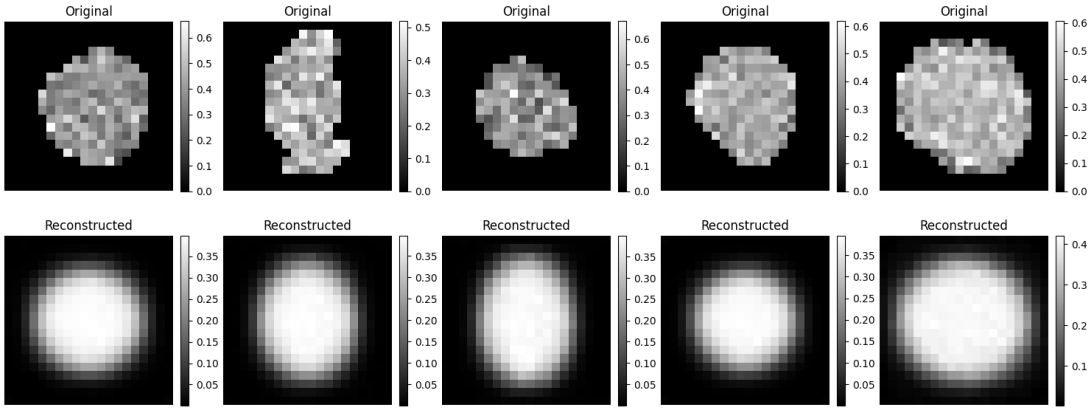


Figure 33: Original and reconstructed images from the `MP2_128` model.

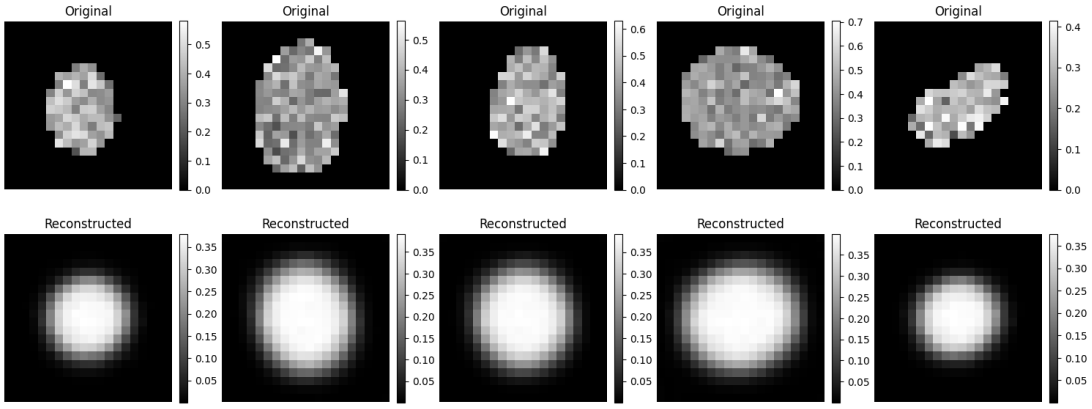


Figure 34: Original and reconstructed images from the `MP3_128` model.

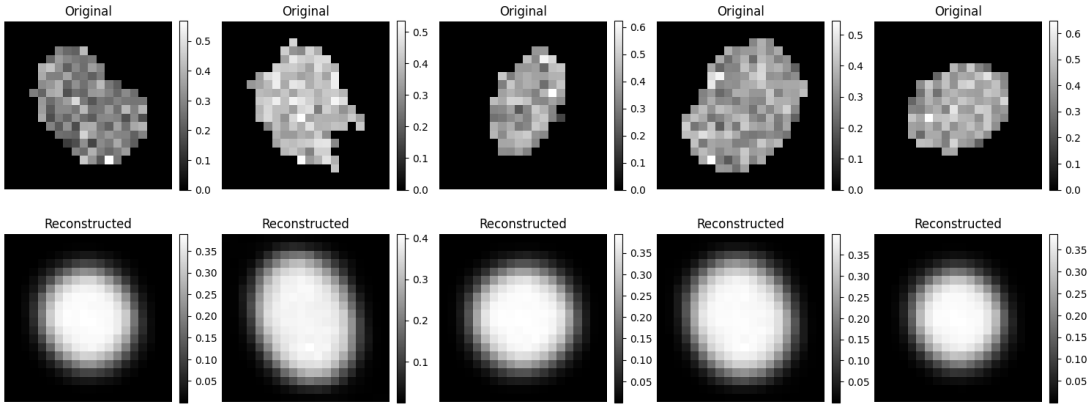


Figure 35: Original and reconstructed images from the `MP3_256` model.

4.2.1.3 Model Selection

In case of the lung cancer dataset, the final model was chosen based on the quantitative results (subsubsection 4.2.1.1) and the qualitative reconstructions

(subsubsection 4.2.1.2). Taking into consideration the above-mentioned concepts, the `MP2_64` architecture was selected for further experiments.
Quantitatively, `MP3_128` and `MP2_128` achieved slightly lower loss values; however, the performance differences were minimal, especially in terms of total and KL losses. In contrast, `MP2_64` provided stable convergence and delivered low final loss values with far fewer trainable parameters, making it more computationally efficient.
On the other hand, qualitatively, `MP2_64` showed consistent reconstructions that preserved overall morphology and central brightness. The results were less sharp than models with more complexity, although the reconstructions remained interpretable and structurally sound.
Considering the balance between efficiency, stability, and interpretability, `MP2_64` offers the most practical trade-off, making it suitable for further β-VAE experimentation on the lung cancer dataset.

4.2.2 Comparison of Different β Values on `MP2_64` Architecture

To explore the effect of varying β values on the `MP2_64` model, we trained the model for $\beta \in \{0.1, 0.5, 0.8, 1, 1.5, 2, 5\}$. The evaluation process is similar to subsection 4.1.2.

4.2.2.1 Quantitative Evaluation

Loss Evaluation

The evaluation of using different β values on the selected `MP2_64` model includes the observation of KL divergence, reconstruction loss, and total loss curves across epochs. The results are shown in Figures ??, ??, and ??.

- KL Divergence Loss ($\mathcal{L}_{\text{KL}}$) increases with higher β values, meaning stronger regularization pressure. $\beta = 0.5$ and $\beta = 0.8$ both converge around 24 (more precisely, 24.21 and 23.75, respectively). However, $\beta = 5$ leads to a complete collapse (KL ≈ 0), meaning the latent space is effectively ignored. On the other hand, $\beta = 0.1$ results in a final KL loss of 11.09, indicating weak enforcement of the prior. A KL loss around 20–25 typically reflects an appropriate tension between compression and expressiveness in the 2D latent space. Using this, $\beta = 0.5$ and $\beta = 0.8$ offer the most meaningful and stable regularization behavior.
- Reconstruction Loss ($\mathcal{L}_{\text{rec}}$) shows the expected inverse trend. $\beta = 0.1$ produces the best reconstruction with its 125.91 final value, while $\beta = 5$ yields the highest reconstruction loss at 194.12. This perfectly shows the trade-off between representation quality and latent compression.
- Total Loss ($\mathcal{L} = \mathcal{L}_{\text{rec}} + \beta \cdot \mathcal{L}_{\text{KL}}$) combines the effects and shows an overall result. In this setup, $\beta = 0.1$ achieves the lowest final loss at 137.00, suggesting good quality reconstruction but at the cost of poor latent structure. The next best losses are obtained with $\beta = 0.5$ and $\beta = 0.8$ (165.81

and 175.95, respectively), which provide a better balance between reconstruction and KL terms. Moving forward to higher values, such as $\beta = 2$ and $\beta = 5$ result in significantly higher values (193.57 and 194.12), indicating the poor latent representation.

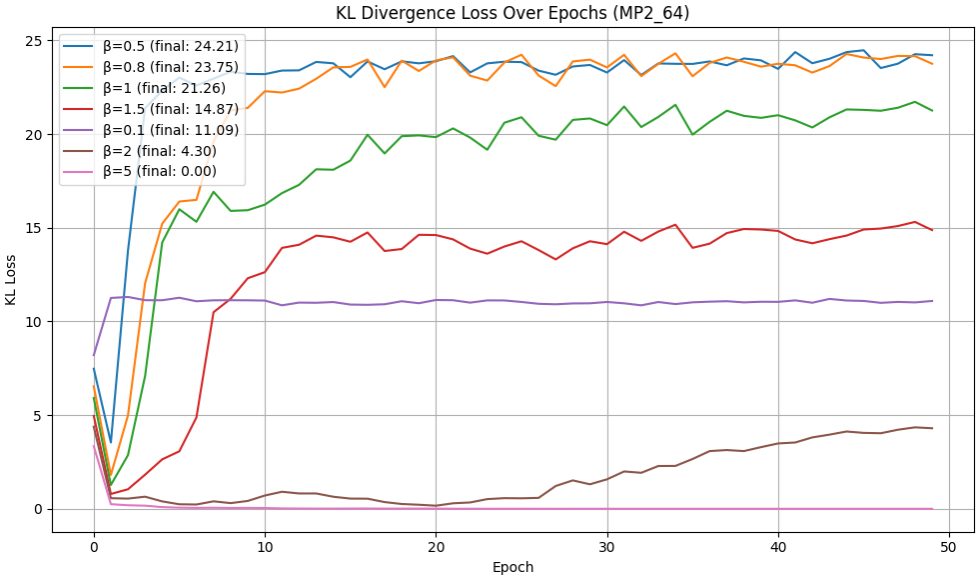


Figure 36: KL Divergence Loss over epochs for MP2_64 under different β values.

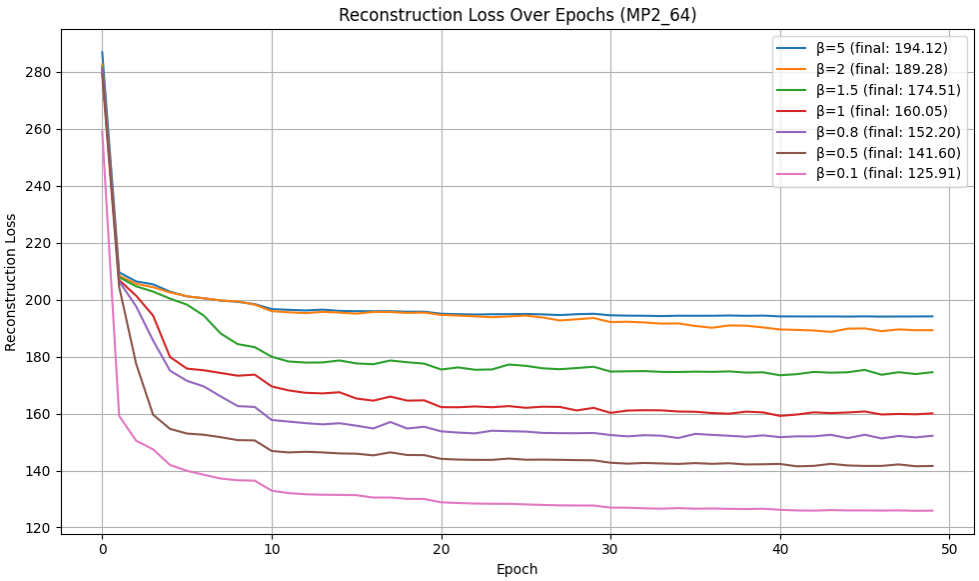


Figure 37: Reconstruction Loss over epochs for MP2_64 under different β values.

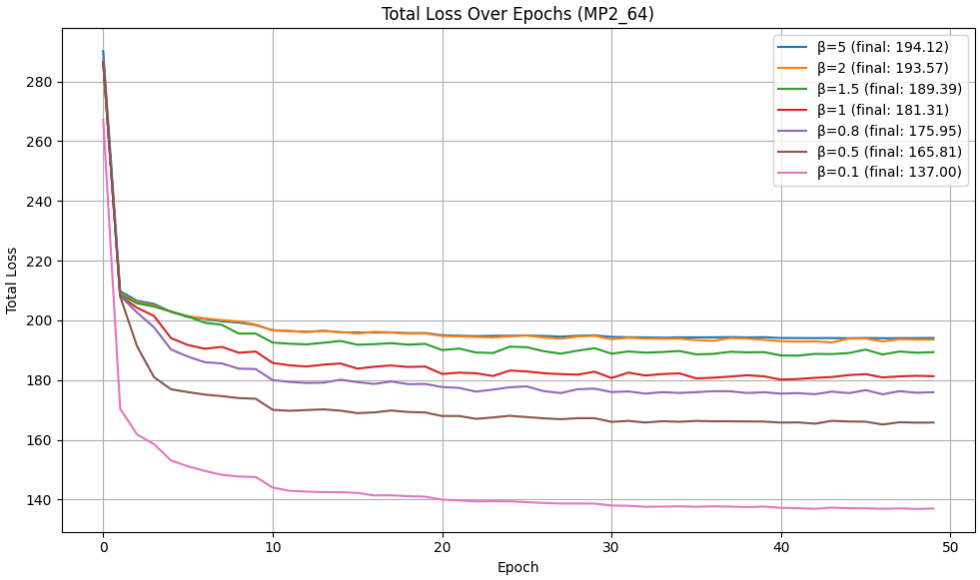


Figure 38: Total Loss ($\mathcal{L}_{\text{rec}} + \beta \cdot \mathcal{L}_{\text{KL}}$) over epochs for MP2_64 under different β values.

Evaluation of KL Divergence on the Latent

Table 2 shows each latent dimension's final KL divergence values at the end of training for different β values. The goal is to have both dimensions contribute similarly and remain close to the standard normal distribution.
The table shows that in case of very low β values (0.1 and 0.5) the results are minimal across both dimensions, indicating weak regularization. However, as β increases, dimension 1 begins to carry significantly more information than dimension 2. This phenomenon is noticeable at $\beta = 0.8$ and $\beta = 1$, where KL divergence value for dimension 1 jumps to 0.1280 and 0.2342, respectively, while dimension 2 still stays low. This suggests the usage of the latent space, using only one significantly contributing dimension.
At higher values, such as $\beta = 1.5$, 2, and 5, the values show disproportional growth. In case of $\beta = 5$, both dimensions exhibit high final KL values (above 2.2), indicating too aggressive forcing to match the prior distribution, leading to poor reconstructions.

β	KL (dim 1)	KL (dim 2)
0.1	0.0472	0.0606
0.5	0.0417	0.0541
0.8	0.1280	0.0471
1	0.2342	0.0579
1.5	0.8299	0.1042
2	0.3152	0.2350
5	2.2649	2.2335

Table 2: Final KL divergence values per latent dimension in MP2_64 across different β values.

Overall, the best trade-off is achieved at $\beta = 0.8$, where one dimension is active but not dominant, and the reconstruction quality remains high. This indicates the use of $\beta = 0.8$ in the final `MP2_64` setup.

4.2.2.2 Qualitative Evaluation

The following figures help the qualitative reconstruction of the `MP2_64` model for each tested β value. The figures include five representative images from the original dataset and their reconstructions.
Using $\beta = 0.1$ (Figure 39), the model's reconstructions present high fidelity and strong contrast, retaining fine-grained texture and pixel-level variation. However, this suggests overfitting and an under-regularized latent space that prioritizes reconstruction at the cost of generalization.
With $\beta = 0.5$ and $\beta = 0.8$ (Figure 40 and Figure 41), the reconstructions capture the essential structural patterns such as cell contours and intensity gradients while suppressing excessive noise. These setups show a better balance between the reconstruction quality and latent abstraction, supporting the quantitative findings.
As β goes above 1, the quality of the reconstruction is visibly deteriorating. At $\beta = 1.5$ and above (Figure 43, Figure 44, and Figure 45), outputs become significantly blurred and lose meaningful spatial feature details. Furthermore, using an extra value at $\beta = 5$, the results are almost uniform reconstructions, indicating that the latent space is too constrained and incapable of encoding relevant information.

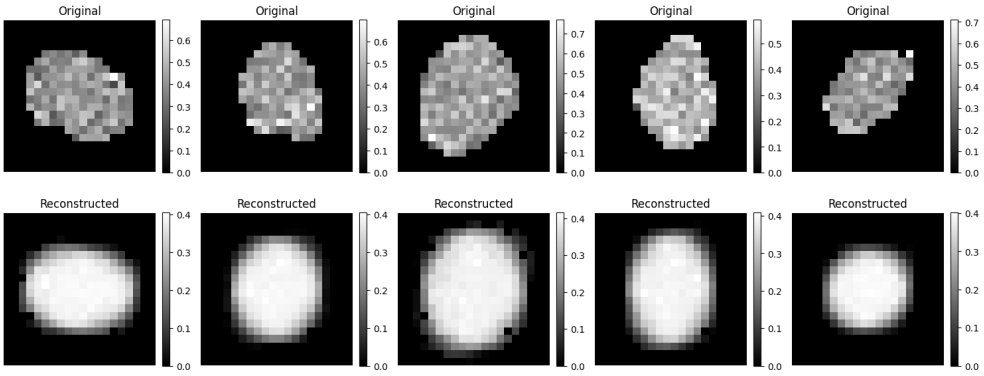


Figure 39: Reconstruction examples for $\beta = 0.1$.

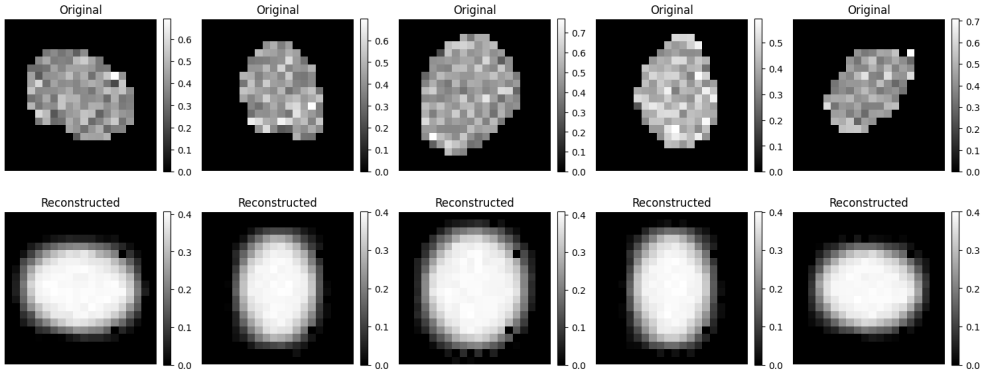


Figure 40: Reconstruction examples for $\beta = 0.5$.

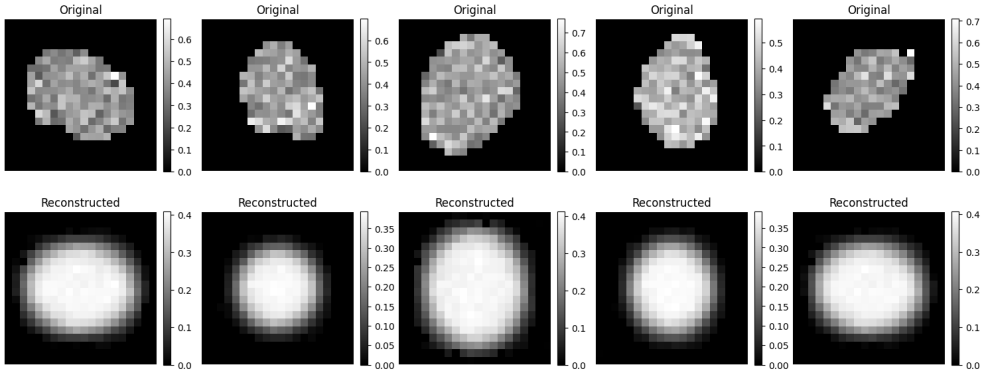


Figure 41: Reconstruction examples for $\beta = 0.8$.

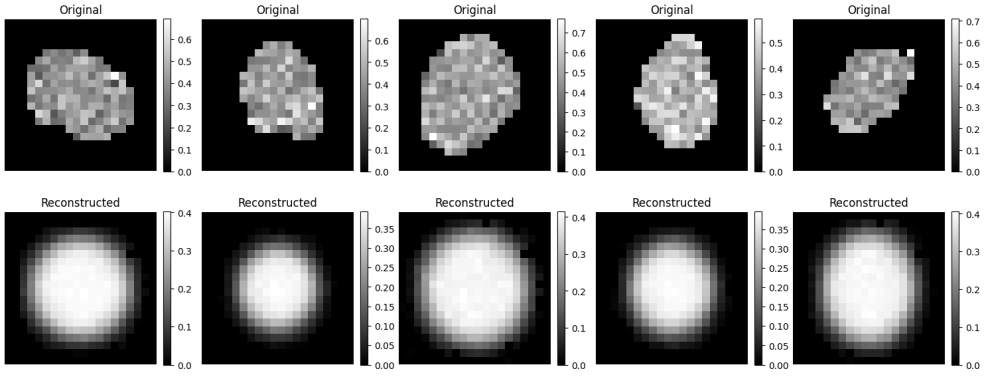


Figure 42: Reconstruction examples for $\beta = 1$.

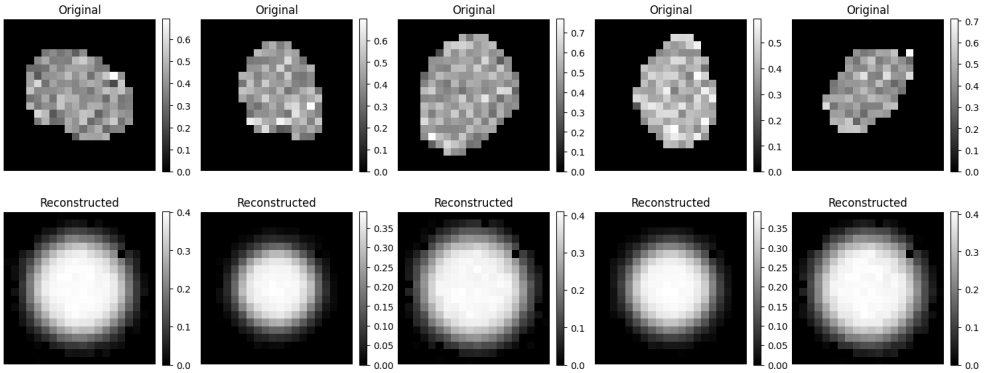


Figure 43: Reconstruction examples for $\beta = 1.5$.

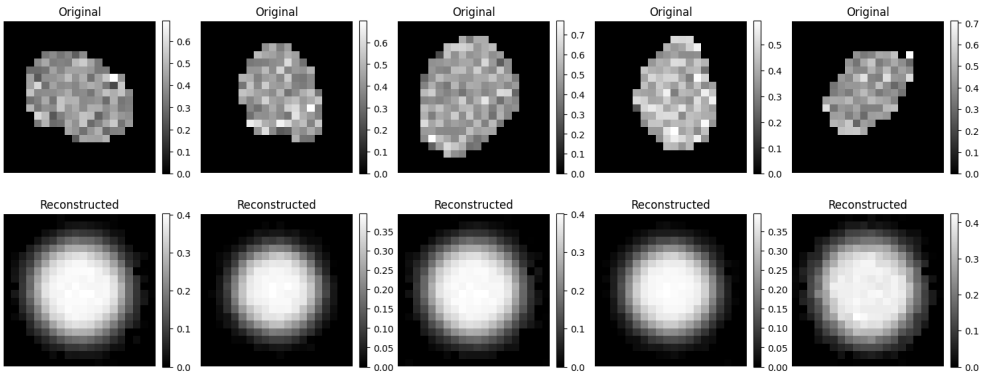


Figure 44: Reconstruction examples for $\beta = 2$.

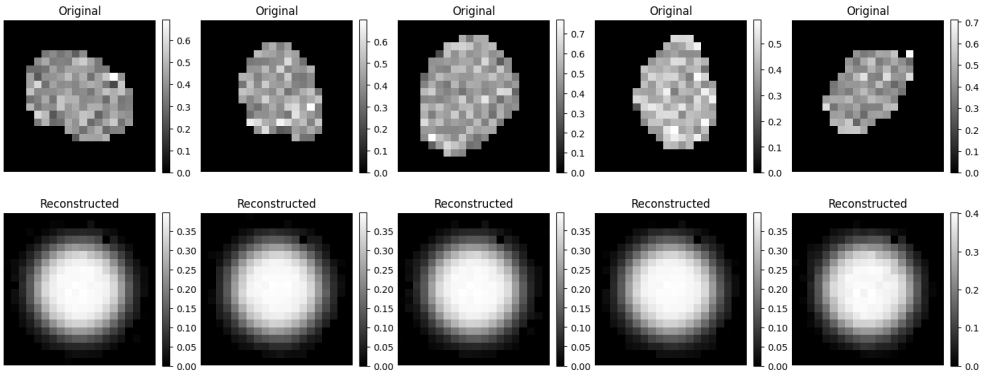


Figure 45: Reconstruction examples for $\beta = 5$.

subsubsection 4.2.2.1 and subsubsection 4.2.2.2 highlight the inherent trade-off between reconstruction quality and latent space regularisation. Lower β values (such as $\beta = 0.1$) yield excellent quality reconstructions but result in nearly collapsed or underutilized latent dimensions. On the other hand, going above $\beta = 1$ enforces strong regularization at the cost of reconstruction fidelity to the point of losing meaningful structure. Mid-range values such as $\beta = 0.5$ and $\beta = 0.8$ offer a better trade-off. Especially, using $\beta = 0.8$,

the model consistently demonstrates well-structured latent space usage while capturing essential image features during reconstruction. In total, $\beta = 0.8$ strikes the best compromise. It maintains coherent, smooth reconstructions that still capture data structure, making it the most balanced and generalizable configuration.

4.3 Phase 3: Evaluation of the Adaptabale VAE

To evaluate the effectiveness of the adaptable VAE, the model was tested on the resized lung cancer dataset with the size 80×80. This resolution was not part of the previous training parts, nor in section 3.3, nor section 3.4, making it a suitable target to assess generalization. The number of layers was automatically calculated via linear interpolation, as mentioned in subsection 3.5.2.

4.3.1 Layer Interpolation

The adaptable VAE framework dynamically determines the number of convolutional layers based on the input image size. Specifically, the number of pixels in the input image is used in a logarithmic interpolation function bounded between two reference sizes:

- $20 \times 20 = 400$ pixels $\rightarrow$ 2 layers
- $140 \times 140 = 19600$ pixels $\rightarrow$ 4 layers

To generalize between these extremes, the number of layers L is calculated in log-space using the following formula:

$$L = \text{round}\left(2 + 0.356 \cdot (\log_2(\text{pixels}) - 8.64)\right)$$

Here, $8.64 = \log_2(400)$ and $14.26 = \log_2(19600)$ define the lower and upper bounds in the logarithmic scale. Applying this formula to an $80 \times 80 = 6400$ pixel image yields:

$$L = \text{round}\left(2 + 0.356 \cdot (\log_2(6400) - 8.64)\right) = 3$$

This result means that the model should use 3 convolutional layers in both the encoder and decoder. This design principle avoids the use of manual configuration while maintaining architectural suitability across different image sizes.

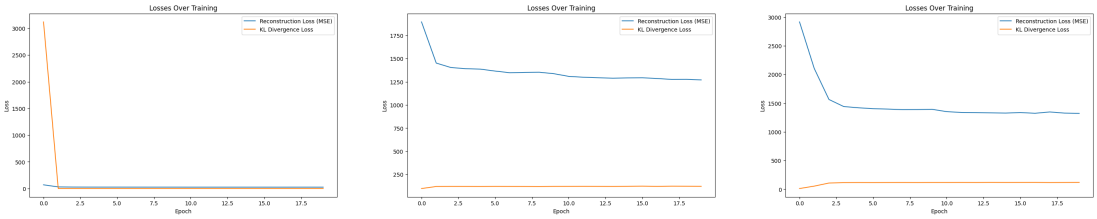
4.3.2 Quantitative Evaluation

To verify whether the interpolated 3-layer model is optimal, we also trained two additional baseline VAEs using 2 and 4 layers on the same dataset. Their respective training loss curves are shown in Figure 46. The key findings are as follows:

- 2-layer model: The KL divergence of this model configuration immediately collapses after the first epoch. The KL term remains near zero during the training, indicating ineffective latent space encoding. Despite low reconstruction loss, the model is essentially not using the latent space.

- 3-layer (interpolated) model: The automatically calculated framework demonstrates balance in the training. This version avoids collapse, exhibiting a stable KL divergence and declining reconstruction loss. Although it has slightly higher reconstruction loss than the 4-layer model, it better maintains latent space usage.

- 4-layer model: This approach achieves the lowest reconstruction loss among the three. However, the KL divergence of this model is lower than the 3-layer variant. While the results are strong, this approach adds more complexity that might not be necessary in all cases.



(a) 2-layer VAE (b) 3-layer (interpolated) (c) 4-layer VAE

Figure 46: Training loss curves for all architectures on the resized 80×80 lung cancer dataset.

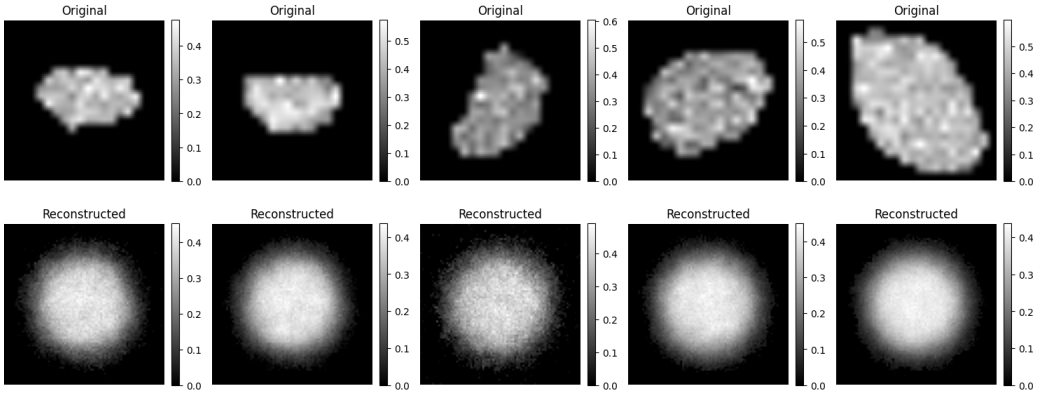
4.3.3 Qualitative Evaluation

From the qualitative side, reconstruction outputs help decision-making. All three architectures are shown in Figure 47. Each image reflects the trade-off between depth, representation capacity, and stability of the different variants.

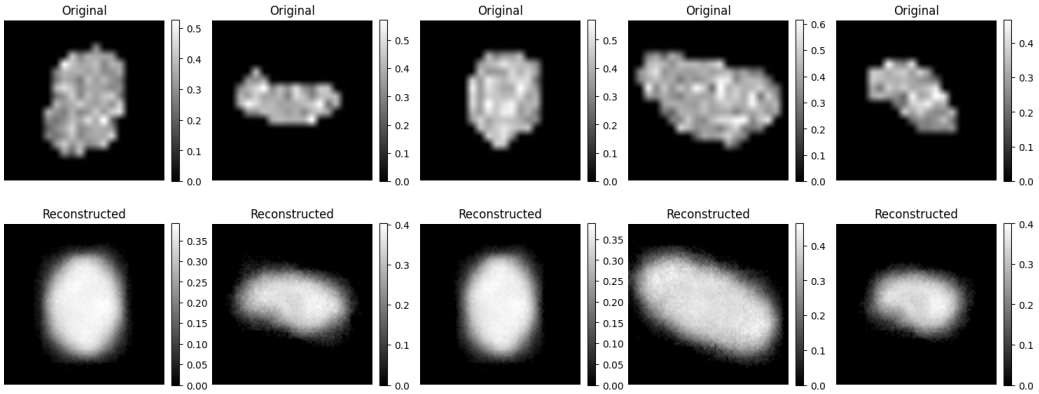
- 2-layer model: Produces noticeably blurred and oversimplified reconstructions. Morphological structures are largely washed out. This result shows the consistency with its flat KL divergence curve and suggests underuse of the latent space. The decoder lacks capacity to reconstruct complex image features from shallow encodings.

- 3-layer model (interpolated): Despite slightly higher reconstruction loss, the reconstructions are visually more stable and detailed. The model shows balance in abstraction and spatial accuracy. It captures overall cell morphology while retaining essential texture.

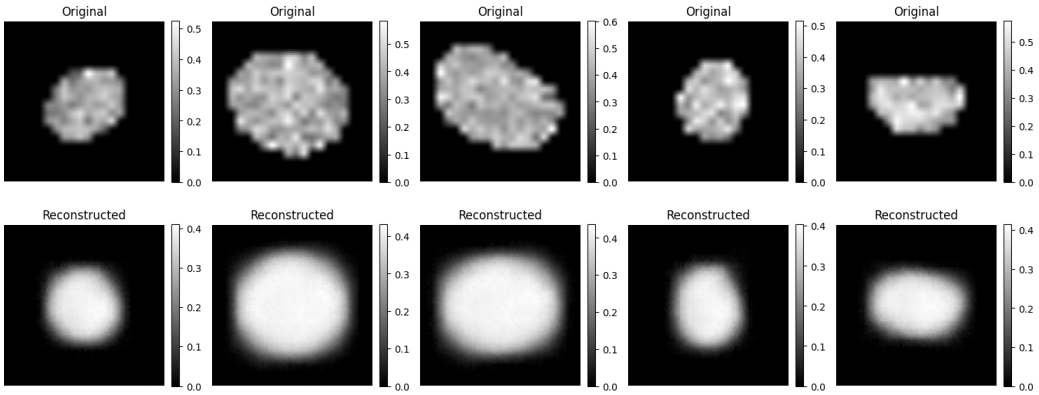
- 4-layer model: Although the reconstructions are smooth, they often exhibit distorted or lost shape boundaries. It can indicate that deeper architecture overcompresses the information.



(a) 2-layer VAE



(b) 3-layer (interpolated)



(c) 4-layer VAE

Figure 47: Reconstruction results for all layer configurations on resized 80×80 lung cancer images.

4.3.4 Summary

Observing more layers on the new dataset, allowed us to reveal the strength of the interpolated 3-layer design calculated by the adaptable VAE framework. It showed that the 2-layer variant lacks sufficient capacity to represent complex spatial features and effectively utilize the latent space. However, the 4-layer

version introduced unnecessary depth.
The 3-layer model, derived from the interpolated formula, offers a well-balanced solution. It maintains meaningful KL divergence throughout training, and delivers reconstructions that preserve both global morphology and internal texture. Its stability and visual fidelity make it the most robust and generalizable option for the given 80×80 lung cancer dataset.
Overall, the results validate the effectiveness of the layer interpolation strategy in selecting a framework that avoids both underfitting and over-compression, adapting to input resolution without manual tuning.

5 Conclusion and Future Work

This final chapter summarizes the main findings of this thesis and outlines possible further investigations. In Section 5.1 we revive the research objectives stated in chapter 1, summarize how the three-phase methodology of chapter 3 led to an adaptable VAE framework, and highlight the key results presented in chapter 4. Then, in Section 5.2, we propose concrete possible extensions that address the remaining challenges identified throughout this work.

5.1 Conclusion

This thesis explored the use of Variational Autoencoders for compressing and reconstructing biomedical microscopy data. By evaluating several different architectures on both MS and lung cancer datasets, we demonstrated that VAEs can effectively reduce the dimensionality of high-resolution imaging data while capturing the key structural features.
A primary contribution of this work is the development of an adaptable VAE framework that dynamically selects its depth based on input resolution. This design solves the problem of implementing dataset-specific architectures and enables a single model design to scale across varying spatial domains. Validating the model on unseen input sizes confirmed that this interpolation-based approach provides stable training and qualitatively promising reconstructions, highlighting its potential for wider applicability in biomedical image analysis.
In addition to architectural evaluations, β-VAE experiments were conducted to explore how scaled regularization influences reconstruction quality and latent space representation. While these results provided valuable insights into the balance between expressiveness and structure, a full disentanglement analysis is left for future research.
Overall, this work confirms that VAEs can offer both effective dimensionality reduction and architectural flexibility. The ability to adapt across varying image sizes without significant manual tuning makes them particularly suited for biomedical applications, where imaging protocols and data characteristics often differ. Through both fixed and adaptable architecture, VAEs demonstrate their potential as scalable and interpretable tools for microscopy image analysis. These models are able to capture relevant biological structures while reducing the complexity of downstream tasks.

5.2 Future Work

5.2.1 Integration with Downstream Predictive Tasks

The current study stops at reconstruction quality and latent compactness. Further development should integrate the VAE-derived embeddings into downstream tasks, such as early disease detection, cell fate prediction, or anomaly detection. By training classifiers (e.g., logistic regression) on the two-dimensional latent vectors, one can assess how well the compressed representations separate clinically meaningful classes.

5.2.2 Validation on Additional Modalities

This thesis focused on two microscopy datasets and still remains to be shown whether the same linear interpolation rule applies to other imaging modalities. In further experiments, the adaptable VAE should be tested on brightfield or phase-contrast images even with multi-channel properties. This will confirm whether adjusting the number of layers alone suffices when channel count or noise characteristics differ.

5.2.3 Comprehensive Disentanglement Analysis

A possible next step is to perform disentanglement study of the learned latent representations. For this purpose, metrics such as the Mutual Information Gap (MIG) or DCI Disentanglement score can quantify how individual latent dimensions correspond to interpretable biological factors. Collecting small labeled subsets would allow us to validate which latent axes capture specific morphological or molecular variations.

5.2.4 Validation in Larger and More Diverse Cohorts

In this thesis, experiments used curated MS and lung cancer datasets, applying the adaptable VAE to larger, more heterogeneous cohorts (e.g., multiple patient samples) is essential. This will test whether the chosen latent dimension ($d_z = 2$) remains optimal or whether a higher dimensionality is needed to capture increased biological diversity. Incorporating rare phenotypes or stress-induced morphological changes could reveal limitations in the current framework.

Bibliography

[1] Alexander Alemi et al. “Fixing a broken ELBO”. In: International Conference on Machine Learning. PMLR. 2018, pp. 159–168.

[2] Walter Alvarado et al. “Denoising Autoencoder Trained on Simulation-Derived Structures for Noise Reduction in Chromatin Scanning Transmission Electron Microscopy”. In: ACS Central Science 9.6 (2023), pp. 1200–1212. DOI: 10.1021/acscentsci.3c00178.

[3] Christopher P Burgess et al. “Understanding disentangling in beta-VAE”. In: arXiv preprint arXiv:1804.03599 (2018).

[4] P. Casti et al. “S3-VAE: A novel Supervised-Source-Separation Variational AutoEncoder algorithm to discriminate tumor cell lines in time-lapse microscopy images”. In: Expert Systems with Applications 232 (2023), p. 120861. DOI: 10.1016/j.eswa.2023.120861. URL: https://doi.org/10.1016/j.eswa.2023.120861.

[5] Nicolas Coudray et al. “Classification and mutation prediction from non–small cell lung cancer histopathology images using deep learning”. In: Nature medicine 24.10 (2018), pp. 1559–1567.

[6] Irene Dankwa-Mullan. “Health Equity and Ethical Considerations in Using Artificial Intelligence in Public Health and Medicine”. In: Preventing Chronic Disease 21 (2024), E240245.

[7] Carl Doersch. “Tutorial on variational autoencoders”. In: arXiv preprint arXiv:1606.05908 (2016).

[8] Sairam Geethanath and J. Thomas Jr. Vaughan. “Accessible magnetic resonance imaging: A review”. In: Journal of Magnetic Resonance Imaging 49.7 (2019), e65–e77. DOI: 10.1002/jmri.26638.

[9] Jared Gehring et al. Imaging cell fate in tissues via spatial transcriptomics and interpretable autoencoders. bioRxiv preprint. May 2024. DOI: 10.1101/2024.05.14.594112. URL: https://www.biorxiv.org/content/10.1101/2024.05.14.594112v1.

[10] Kaiming He et al. “Deep residual learning for image recognition”. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, pp. 770–778.

[11] Amanda N. Henning et al. “Assessing the impact of cell isolation method on B cell gene expression using next-generation sequencing”. In: Experimental Hematology 146 (2025), p. 104766. DOI: 10.1016/j.exphem.2025.104766.

[12] Irina Higgins et al. “beta-vae: Learning basic visual concepts with a constrained variational framework”. In: International Conference on Learning Representations (ICLR). 2017.

[13] Shah Hussain et al. “Modern Diagnostic Imaging Technique Applications and Risk Factors in the Medical Field: A Review”. In: BioMed Research International 2022 (2022), p. 5164970. DOI: 10.1155/2022/5164970.

[14] Diederik P Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: arXiv preprint arXiv:1312.6114 (2013).

[15] Burak Kocak et al. “Radiology AI and sustainability paradox: environmental, economic, and social dimensions”. In: Insights into Imaging 16.1 (2025), p. 88.

[16] Matthias Lettau et al. “Human CD27+ memory B cells colonize a superficial follicular zone in the palatine tonsils with similarities to the spleen. A multicolor immunofluorescence study of lymphoid tissue”. In: PLoS ONE 15.3 (2020), e0229778. DOI: 10.1371/journal.pone.0229778.

[17] Geert Litjens et al. “A survey on deep learning in medical image analysis”. In: Medical image analysis 42 (2017), pp. 60–88.

[18] Alireza Makhzani et al. “Adversarial Autoencoders”. In: arXiv preprint arXiv:1511.05644 (2015).

[19] Phuc Nguyen et al. “Unsupervised discovery of dynamic cell phenotypic states from transmitted light movies”. In: PLoS computational biology 17.12 (2021), e1009626.

[20] Sheng-Yong Niu et al. “Boundary-Preserved Deep Denoising of Stochastic Resonance Enhanced Multiphoton Images”. In: IEEE Journal of Translational Engineering in Health and Medicine 10 (2022), p. 1800812. DOI: 10.1109/JTEHM.2022.3206488.

[21] Sheng-Yong Niu et al. “Boundary-Preserved Deep Denoising of the Stochastic Resonance Enhanced Multiphoton Images”. In: arXiv preprint arXiv:1904.06329 (2019).

[22] Ammar A. Oglat. “A review of ultrasound contrast media”. In: F1000Research 12 (2024). [version 3; peer review: 2 approved], p. 1444. DOI: 10.12688/f1000research.140131.3.

[23] Pranav Rajpurkar et al. “Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists”. In: PLoS Medicine 15.11 (2018), e1002686. DOI: 10.1371/journal.pmed.1002686.

[24] Salah Rifai et al. “Contractive Auto-Encoders: Explicit Invariance During Feature Extraction”. In: Proceedings of the 28th International Conference on Machine Learning. Omnipress. 2011, pp. 833–840.

[25] Raghavendra Selvan et al. “Carbon footprint of selecting and training deep learning models for medical image analysis”. In: arXiv preprint arXiv:2203.02202 (2022).

[26] Carsen Stringer et al. “Cellpose: a generalist algorithm for cellular segmentation”. In: Nature Methods 18.1 (2021), pp. 100–106. DOI: 10.1038/s41592-020-01018-x.

[27] Mingxing Tan and Quoc V Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: International Conference on Machine Learning. PMLR. 2019, pp. 6105–6114.

[28] Effy Vayena, Alessandro Blasimme, and I Glenn Cohen. “Machine learning in medicine: Addressing ethical challenges”. In: PLoS medicine 15.11 (2018), e1002689.

[29] Pascal Vincent et al. “Extracting and Composing Robust Features with Denoising Autoencoders”. In: Proceedings of the 25th International Conference on Machine Learning. ACM. 2008, pp. 1096–1103.

[30] Philip J. Withers et al. “X-ray computed tomography”. In: Nature Reviews Methods Primers 1 (2021), p. 18. DOI: 10.1038/s43586-021-00015-4.

[31] Hao-Yu Yang and Lawrence H. Staib. “Dual Adversarial Autoencoder for Dermoscopic Image Generative Modeling”. In: 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI). IEEE, 2019, pp. 1247–1250. DOI: 10.1109/ISBI.2019.8759525. URL: https://doi.org/10.1109/ISBI.2019.8759525.

A Detailed Tables from Methodology

In this appendix, we collect all of the detailed tabular summaries that were originally embedded in Chapter chapter 3.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 140, 140]	608
ReLU	[1, 32, 140, 140]	0
MaxPool2d	[1, 32, 70, 70]	0
Conv2d	[1, 64, 70, 70]	18,496
ReLU	[1, 64, 70, 70]	0
MaxPool2d	[1, 64, 35, 35]	0
Linear (μ head)	[1, 2]	156,802
Linear ($\log \sigma$ head)	[1, 2]	156,802
Linear (decoder input)	[1, 78,400]	235,200
ConvTranspose2d	[1, 32, 70, 70]	18,464
ReLU	[1, 32, 70, 70]	0
ConvTranspose2d	[1, 2, 140, 140]	578
Sigmoid	[1, 2, 140, 140]	0
Total parameters		586,950
Trainable parameters		586,950
Non-trainable parameters		0
Total mult-adds (MB)		204.90
Input size (MB)		0.16
Forward/backward pass (MB)		9.72
Params size (MB)		2.35
Estimated total size (MB)		12.23

Table 3: Detailed summary of the `VAE_MP2` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 140, 140]	608
ReLU	[1, 32, 140, 140]	0
MaxPool2d	[1, 32, 70, 70]	0
Conv2d	[1, 64, 70, 70]	18,496
ReLU	[1, 64, 70, 70]	0
MaxPool2d	[1, 64, 35, 35]	0
Conv2d	[1, 128, 35, 35]	73,856
ReLU	[1, 128, 35, 35]	0
MaxPool2d	[1, 128, 17, 17]	0
Linear (μ head)	[1, 2]	73,986
Linear ($\log \sigma$ head)	[1, 2]	73,986
Linear (decoder input)	[1, 36,992]	110,976
ConvTranspose2d	[1, 64, 35, 35]	131,136
ReLU	[1, 64, 35, 35]	0
ConvTranspose2d	[1, 32, 70, 70]	32,800
ReLU	[1, 32, 70, 70]	0
ConvTranspose2d	[1, 2, 140, 140]	1,026
Sigmoid	[1, 2, 140, 140]	0
Total parameters		516,870
Trainable parameters		516,870
Non-trainable parameters		0
Total mult-adds (MB)		534.75
Input size (MB)		0.16
Forward/backward pass (MB)		11.27
Params size (MB)		2.07
Estimated total size (MB)		13.50

Table 4: Detailed summary of the `VAE_MP3` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 140, 140]	608
ReLU	[1, 32, 140, 140]	0
MaxPool2d	[1, 32, 70, 70]	0
Conv2d	[1, 64, 70, 70]	18,496
ReLU	[1, 64, 70, 70]	0
MaxPool2d	[1, 64, 35, 35]	0
Conv2d	[1, 128, 35, 35]	73,856
ReLU	[1, 128, 35, 35]	0
MaxPool2d	[1, 128, 17, 17]	0
Conv2d	[1, 256, 17, 17]	295,168
ReLU	[1, 256, 17, 17]	0
MaxPool2d	[1, 256, 8, 8]	0
Linear (μ head)	[1, 2]	32,770
Linear ($\log \sigma$ head)	[1, 2]	32,770
Linear (decoder input)	[1, 16,384]	49,152
ConvTranspose2d	[1, 128, 17, 17]	524,416
ReLU	[1, 128, 17, 17]	0
ConvTranspose2d	[1, 64, 35, 35]	131,136
ReLU	[1, 64, 35, 35]	0
ConvTranspose2d	[1, 32, 71, 71]	32,800
ReLU	[1, 32, 71, 71]	0
ConvTranspose2d	[1, 2, 143, 143]	1,026
ReLU	[1, 2, 143, 143]	0
Upsample	[1, 2, 140, 140]	0
Total parameters		1,192,198
Trainable parameters		1,192,198
Non-trainable parameters		0
Total mult-adds (MB)		776.96
Input size (MB)		0.16
Forward/backward pass (MB)		12.04
Params size (MB)		4.77
Estimated total size (MB)		16.97

Table 5: Detailed summary of the `VAE_MP4` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 70, 70]	608
BatchNorm2d	[1, 32, 70, 70]	64
ReLU	[1, 32, 70, 70]	0
Conv2d	[1, 64, 35, 35]	18,496
BatchNorm2d	[1, 64, 35, 35]	128
ReLU	[1, 64, 35, 35]	0
Linear (μ head)	[1, 2]	156,802
Linear ($\log \sigma$ head)	[1, 2]	156,802
Linear (decoder input)	[1, 78,400]	235,200
ConvTranspose2d	[1, 32, 70, 70]	32,800
BatchNorm2d	[1, 32, 70, 70]	64
ReLU	[1, 32, 70, 70]	0
ConvTranspose2d	[1, 2, 140, 140]	1,026
Sigmoid	[1, 2, 140, 140]	0
Total parameters		601,990
Trainable parameters		601,990
Non-trainable parameters		0
Total mult-adds (MB)		207.02
Input size (MB)		0.16
Forward/backward pass (MB)		7.21
Params size (MB)		2.41
Estimated total size (MB)		9.78

Table 6: Detailed summary of the `VAE_BN2` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 70, 70]	608
BatchNorm2d	[1, 32, 70, 70]	64
ReLU	[1, 32, 70, 70]	0
Conv2d	[1, 64, 35, 35]	18,496
BatchNorm2d	[1, 64, 35, 35]	128
ReLU	[1, 64, 35, 35]	0
Conv2d	[1, 128, 18, 18]	73,856
BatchNorm2d	[1, 128, 18, 18]	256
ReLU	[1, 128, 18, 18]	0
Linear (μ head)	[1, 2]	82,946
Linear ($\log \sigma$ head)	[1, 2]	82,946
Linear (decoder input)	[1, 41,472]	124,416
ConvTranspose2d	[1, 64, 36, 36]	73,792
BatchNorm2d	[1, 64, 36, 36]	128
ReLU	[1, 64, 36, 36]	0
ConvTranspose2d	[1, 32, 72, 72]	18,464
BatchNorm2d	[1, 32, 72, 72]	64
ReLU	[1, 32, 72, 72]	0
ConvTranspose2d	[1, 2, 144, 144]	578
Sigmoid	[1, 2, 144, 144]	0
Upsample	[1, 2, 140, 140]	0
Total parameters		476,742
Trainable parameters		476,742
Non-trainable parameters		0
Total mult-adds (MB)		253.19
Input size (MB)		0.16
Forward/backward pass (MB)		9.07
Params size (MB)		1.91
Estimated total size (MB)		11.14

Table 7: Detailed summary of the `VAE_BN3` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 70, 70]	608
BatchNorm2d	[1, 32, 70, 70]	64
ReLU	[1, 32, 70, 70]	0
Conv2d	[1, 64, 35, 35]	18,496
BatchNorm2d	[1, 64, 35, 35]	128
ReLU	[1, 64, 35, 35]	0
Conv2d	[1, 128, 18, 18]	73,856
BatchNorm2d	[1, 128, 18, 18]	256
ReLU	[1, 128, 18, 18]	0
Conv2d	[1, 256, 9, 9]	295,168
BatchNorm2d	[1, 256, 9, 9]	512
ReLU	[1, 256, 9, 9]	0
Linear (μ head)	[1, 2]	41,474
Linear ($\log \sigma$ head)	[1, 2]	41,474
Linear (decoder input)	[1, 20,736]	62,208
ConvTranspose2d	[1, 128, 18, 18]	524,416
BatchNorm2d	[1, 128, 18, 18]	256
ReLU	[1, 128, 18, 18]	0
ConvTranspose2d	[1, 64, 36, 36]	131,136
BatchNorm2d	[1, 64, 36, 36]	128
ReLU	[1, 64, 36, 36]	0
ConvTranspose2d	[1, 32, 72, 72]	32,800
BatchNorm2d	[1, 32, 72, 72]	64
ReLU	[1, 32, 72, 72]	0
ConvTranspose2d	[1, 2, 144, 144]	1,026
Sigmoid	[1, 2, 144, 144]	0
Upsample	[1, 2, 140, 140]	0
Total parameters		1,224,070
Trainable parameters		1,224,070
Non-trainable parameters		0
Total mult-adds (MB)		604.79
Input size (MB)		0.16
Forward/backward pass (MB)		9.90
Params size (MB)		4.90
Estimated total size (MB)		14.95

Table 8: Detailed summary of the `VAE_BN4` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 20, 20]	320
ReLU	[1, 32, 20, 20]	0
MaxPool2d	[1, 32, 10, 10]	0
Conv2d	[1, 64, 10, 10]	18,496
ReLU	[1, 64, 10, 10]	0
MaxPool2d	[1, 64, 5, 5]	0
Conv2d	[1, 128, 5, 5]	73,856
ReLU	[1, 128, 5, 5]	0
MaxPool2d	[1, 128, 3, 3]	0
Linear (μ head)	[1, 2]	2,306
Linear ($\log \sigma$ head)	[1, 2]	2,306
Linear (decoder input)	[1, 1152]	3,456
ConvTranspose2d	[1, 64, 5, 5]	73,792
ReLU	[1, 64, 5, 5]	0
ConvTranspose2d	[1, 32, 10, 10]	32,800
ReLU	[1, 32, 10, 10]	0
ConvTranspose2d	[1, 1, 20, 20]	513
Sigmoid	[1, 1, 20, 20]	0
Total parameters		207,845
Trainable parameters		207,845
Non-trainable parameters		0
Total mult-adds (MB)		9.16
Input size (MB)		0.00
Forward/backward pass (MB)		0.23
Params size (MB)		0.83
Estimated total size (MB)		1.06

Table 9: Detailed summary of the `VAE_MP3_128` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 64, 20, 20]	640
ReLU	[1, 64, 20, 20]	0
MaxPool2d	[1, 64, 10, 10]	0
Conv2d	[1, 128, 10, 10]	73,856
ReLU	[1, 128, 10, 10]	0
MaxPool2d	[1, 128, 5, 5]	0
Conv2d	[1, 256, 5, 5]	295,168
ReLU	[1, 256, 5, 5]	0
MaxPool2d	[1, 256, 3, 3]	0
Linear (μ head)	[1, 2]	4,610
Linear ($\log \sigma$ head)	[1, 2]	4,610
Linear (decoder input)	[1, 2304]	6,912
ConvTranspose2d	[1, 128, 5, 5]	295,040
ReLU	[1, 128, 5, 5]	0
ConvTranspose2d	[1, 64, 10, 10]	131,136
ReLU	[1, 64, 10, 10]	0
ConvTranspose2d	[1, 1, 20, 20]	1,025
Sigmoid	[1, 1, 20, 20]	0
Total parameters		812,997
Trainable parameters		812,997
Non-trainable parameters		0
Total mult-adds (MB)		35.94
Input size (MB)		0.00
Forward/backward pass (MB)		0.46
Params size (MB)		3.25
Estimated total size (MB)		3.71

Table 10: Detailed summary of the `VAE_MP3_256` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 64, 20, 20]	640
ReLU	[1, 64, 20, 20]	0
MaxPool2d	[1, 64, 10, 10]	0
Conv2d	[1, 128, 10, 10]	73,856
ReLU	[1, 128, 10, 10]	0
MaxPool2d	[1, 128, 5, 5]	0
Linear (μ head)	[1, 2]	6,402
Linear ($\log \sigma$ head)	[1, 2]	6,402
Linear (decoder input)	[1, 3200]	9,600
ConvTranspose2d	[1, 64, 10, 10]	131,136
ReLU	[1, 64, 10, 10]	0
ConvTranspose2d	[1, 1, 20, 20]	1,025
Sigmoid	[1, 1, 20, 20]	0
Total parameters		229,061
Trainable parameters		229,061
Non-trainable parameters		0
Total mult-adds (MB)		21.19
Input size (MB)		0.00
Forward/backward pass (MB)		0.39
Params size (MB)		0.92
Estimated total size (MB)		1.31

Table 11: Detailed summary of the `VAE_MP2_128` architecture.

Layer (Type)	Output Shape	Parameters
Conv2d	[1, 32, 20, 20]	320
ReLU	[1, 32, 20, 20]	0
MaxPool2d	[1, 32, 10, 10]	0
Conv2d	[1, 64, 10, 10]	18,496
ReLU	[1, 64, 10, 10]	0
MaxPool2d	[1, 64, 5, 5]	0
Linear (μ head)	[1, 2]	3,202
Linear ($\log \sigma$ head)	[1, 2]	3,202
Linear (decoder input)	[1, 1600]	4,800
ConvTranspose2d	[1, 32, 10, 10]	32,800
ReLU	[1, 32, 10, 10]	0
ConvTranspose2d	[1, 1, 20, 20]	513
ReLU	[1, 1, 20, 20]	0
Upsample	[1, 1, 20, 20]	0
Total parameters		63,333
Trainable parameters		63,333
Non-trainable parameters		0
Total mult-adds (MB)		5.47
Input size (MB)		0.00
Forward/backward pass (MB)		0.20
Params size (MB)		0.25
Estimated total size (MB)		0.45

Table 12: Detailed summary of the `VAE_MP2_64` architecture.