KU LEUVEN

FACULTY OF BIOSCIENCE ENGINEERING

Deep Learning Methods for Bacterial Image-Based Profiling

Promoter:
Prof. Sander Govers
Department of Biology
Microbial Systems Cell Biology Group

Dissertation presented in
fulfillment of the requirements
for the degree of Master of Science:
Bioinformatics

Theodoro GASPERIN TERRA CAMARGO

June 2025

Contents

Acknowledgments

A very special thanks to my supervisor, Bart Steemans, for his thoughtful insights and countless hours in the trenches helping me over the past year. I learned a lot from him, and his guidance helped me navigate the many obstacles I encountered along the way. Without him, this thesis would not exist. I would also like to thank Professor Sander Govers for his knowledgeable insights during our biweekly meetings, his thoughtful suggestions, and the general care he showed to everyone in the lab. Next, I would like to thank my fellow office colleagues who shared the ups and downs of doing a master's thesis with me: Dries, Luna, Jolien, and Keana. Having them beside me made my days much more enjoyable than they would have been if I had worked alone. Our mutual support and jokes carried us through to the finish line. I would also like to thank everyone in the lab: Alix, Kristina, Matthew, Jiaan, Joëlle, Kaat, Els, and Cathy, with whom I shared countless meals and stories. They made our basement much brighter than it would have been without them. A very special thanks goes to some of my dear friends and colleagues from the Master of Bioinformatics program who, over these past two years, helped me through assignments, exams, bad weather, and life in general: Kodai, Tarek, Thanos, Wout, Laura, Emma, Niamh, William, Alita, Cristiano, Angelica, Bram, Disha, Maria, Casper, Irate, Harry, Sofya, Evgeniya, August, Ismael, Sparsh, and Andrew. Another special thanks to all my dear friends from the winding road that is life: Pierre, Ruben, Imad, Tyffany, Natalya, Wadih, Thiago, Isadora, Timothy, Ilies, Antoine, Julian, Zach, Enzo, Lou, Guga, Leonel, Matheus and Aalam. Finally, I thank my parents and my brother for their love and support since the day I was born. Even from far away, I know they are always there with me wherever I go. I also thank the rest of my family, whom I love dearly and miss very much.

List of Abbreviations and Symbols Used

Abbreviations:

AUC: Area Under the Curve
CNN: Convolutional Neural Network
CRISPR: Clustered Regularly Interspaced Short Palindromic Repeats
DAPI: 4',6-Diamidino-2-Phenylindole
DBSCAN: Density-Based Spatial Clustering of Applications with Noise
FtsZ: Filamenting Temperature-Sensitive Mutant Z
GMM: Gaussian Mixture Model
HCI: High-Content Imaging
JUMP: Joint Undertaking in Morphological Profiling
MBConv: Mobile Inverted Bottleneck Convolution
PALM/STORM: Photoactivated Localization Microscopy/Stochastic Optical Reconstruction Microscopy
PCA: Principal Component Analysis
PHATE: Potential of Heat-diffusion for Affinity-based Trajectory Embedding
RBF: Radial Basis Function
RoIAlign: Region of Interest Align
SEM: Scanning Electron Microscopy
SeqA: Sequestration Protein A
SIM: Structured Illumination Microscopy
siRNA: Small Interfering RNA
STED: Stimulated Emission Depletion
SVM: Support Vector Machine
TEM: Transmission Electron Microscopy
t-SNE: t-Distributed Stochastic Neighbor Embedding
UMAP: Uniform Manifold Approximation and Projection
ViT: Vision Transformer
WGA: Wheat Germ Agglutinin

Symbols:

φ: Compound scaling coefficient (used in EfficientNet architecture)
µm: Micrometer (unit of length for bacterial cell size)
nm: Nanometer (unit of length for resolution limits in microscopy)

List of Tables

List of Figures

Abstract

Image-based profiling enables the extraction of quantitative morphological data from cellular microscopy to study phenotypic responses. Over the past decade, this approach has advanced considerably, propelled by developments in deep learning and high-throughput imaging technologies. While eukaryotic cell profiling has seen widespread application and methodological development, bacterial image-based profiling remains comparatively underexplored. To address these limitations, this thesis applies deep learning techniques to enhance bacterial image-based profiling. Specifically, it focuses on brightfield microscopy for cell segmentation and multi-channel fluorescence imaging for phenotypic analysis. Two primary objectives were pursued: (1) to establish brightfield microscopy as a viable modality for accurate bacterial cell segmentation, and (2) to employ convolutional neural networks (CNNs) for unbiased feature extraction to capture phenotypic variability and infer cell cycle progression in *Escherichia coli*. To achieve this, we first focused on enhancing segmentation of brightfield images by retraining a CNN-based Omnipose model using paired phase contrast and brightfield data. Leveraging transfer learning proved especially beneficial, enabling strong performance even with limited training data. Through systematic exploration of key segmentation parameters—including batch size, mask thresholds, and flow thresholds—we identified configurations that significantly improved segmentation quality. These experiments offer valuable insights into how such models can be effectively tuned for bacterial datasets. Building on this, we developed a CNN-based feature extraction pipeline using EfficientNet-B0, trained on multi-channel fluorescence images that included phase contrast, nucleoid (DAPI), FtsZ (Venus), and SeqA (mCherry) markers. By generating cell-centered image patches and applying supervised training, we extracted high-dimensional embeddings that captured nuanced morphological patterns. These embeddings enabled the identification of phenotypically distinct mutants and suggested partial cell cycle trajectories. Our approach demonstrates how deep learning can replace manual feature engineering by capturing biologically relevant signals in a scalable and unbiased manner. Despite limitations in dataset diversity and the interpretability of learned features, this work establishes a reproducible computational pipeline—publicly available on GitHub—that advances high-throughput bacterial phenotyping. It lays the groundwork for future integration with multi-omics data and applications in antibiotic screening and functional genomics.

1 Context and Aims

Image-based profiling is a technique that involves extracting rich quantitative information from microscopy images to study cellular structure, function, and behavior [1]. It allows researchers to systematically analyze how cells respond to genetic, chemical, or environmental changes, often revealing subtle phenotypic differences that are not easily captured by traditional methods [2, 3].

Bacteria are single-celled microorganisms that play essential roles in ecosystems, human health, and biotechnology [4]. Understanding bacterial phenotypes—the observable characteristics shaped by genetic and environmental factors—is crucial for studying their behavior, identifying antibiotic resistance, and uncovering new biological mechanisms. However, applying image-based profiling to bacterial systems remains challenging due to their small size, morphological variability, and the technical limitations of standard imaging and analysis pipelines [5].

While image-based profiling has shown great promise in eukaryotic systems, its application in bacterial research is still limited. One contributing factor is that traditional bacterial image-processing techniques are not precise enough and outdated [6]. In addition, the reliance on fluorescence or phase contrast imaging can be experimentally demanding: fluorescence requires genetic modification or staining [7], while phase contrast, though relatively inexpensive, may still require specialized equipment or alignment procedures [8]. Brightfield microscopy, by contrast, is simpler and more accessible, but its lower contrast and noisier images make automated analysis difficult [9].

Recent advances in deep learning provide powerful image-analysis tools to address the challenges of bacterial image-based profiling. Convolutional neural networks (CNNs) excel in computer vision tasks such as segmentation and feature extraction [10], yet their application to bacterial systems remains underexplored. This thesis explores the potential of applying deep learning methods in bacterial image-based profiling. This work is divided into two primary efforts, each leveraging CNNs to overcome traditional limitations and unlock new biological insights.

The first aim is to validate and enhance the use of deep learning for brightfield microscopy segmentation in bacterial cells. Brightfield imaging, while widely accessible, suffers from low contrast, making automated analysis challenging. To address this, we retrain a CNN, Omnipose [11], on paired phase contrast and brightfield images, enabling the model to learn accurate segmentation from brightfield data alone. This approach improves the reliability of cell boundary detection, upgrading the segmentation component of the profiling pipeline. The second aim is to employ a CNN, EfficientNet [12], to extract unbiased, high-dimensional features from unprocessed multi-channel bacterial images. These features are analyzed to uncover subtle phenotypic variations, infer cell cycle progression, and identify distinct characteristics of mutant bacterial populations. By learning complex patterns directly from raw data, this deep learning-based feature extraction improves the ability to reveal biologically significant insights that traditional methods might overlook.

2 Literature Review

2.1 Image-Based Profiling

Image-based profiling is a powerful computer vision approach in biological research that uses quantitative image analysis to extract rich, multidimensional data from images of cells or tissues [1]. This method allows researchers to assess phenotypic variations at the cellular level, often in response to genetic, chemical, or environmental perturbations, and is used extensively in fields like drug discovery, disease modeling, and functional genomics [2, 3]. By capturing diverse structural and functional properties, image-based profiling offers a window into complex biological systems, providing a bridge between molecular data and visual observations of cellular phenotypes [13].

The general workflow of image-based profiling follows 5 main steps as shown in Figure 1: sample preparation, image acquisition, image segmentation, feature extraction, data analysis.

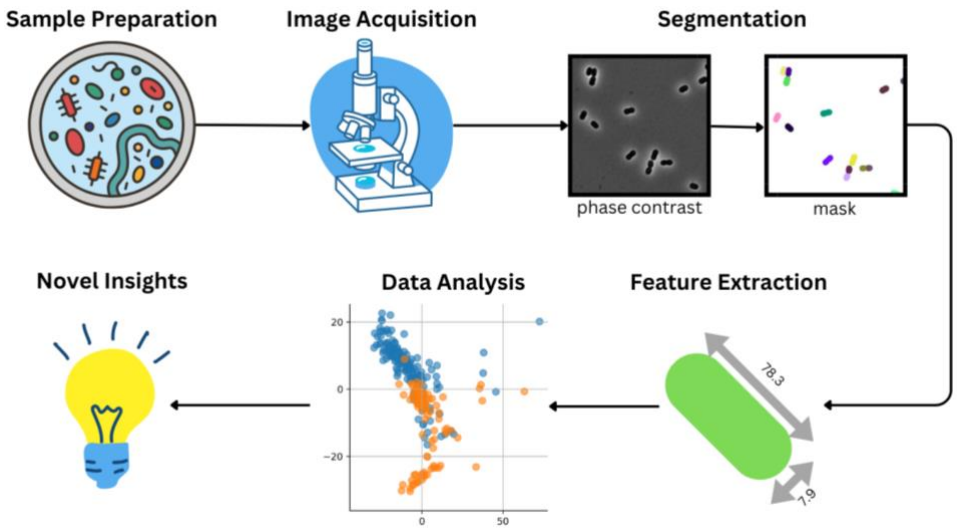


Figure 1: General workflow for image-based profiling. Workflow follows the subsequent steps: sample preparation, image acquisition, segmentation, feature extraction, data analysis.

2.1.1 Sample Preparation

Sample preparation is a critical step in image-based profiling, ensuring that cellular structures and perturbation effects can be accurately observed and quantified. This process involves carefully designing experimental conditions, applying perturbations, and fluorescently labeling specific cellular components.

Perturbations can be introduced to study cellular responses at various levels. Chemical perturbations involve treating cells with small molecules, drugs, or toxins to observe their effects on cellular morphology, viability, and subcellular structures [1, 14, 15]. Genetic perturbations rely on techniques such as CRISPR, siRNA-mediated knockdown or transposon

mutagenesis, providing powerful methods for disrupting genes to study their functional roles [16]. Environmental perturbations, such as changes in temperature, osmotic stress, pH shifts, or nutrient availability, provide insights into cellular adaptation mechanisms [17]. Each type of perturbation can induce distinct phenotypic changes, which can be captured and analyzed through high-content imaging [1].

Fluorescent labeling of specific cellular structures is essential for distinguishing organelles and understanding subcellular organization as can be seen in Figure 2. In eukaryotic cells, membrane markers such as DiI or wheat germ agglutinin (WGA) highlight plasma membrane integrity and morphology, while nuclear stains like DAPI (Figure 2 B) or Hoechst allow visualization of nuclear size and chromatin organization [18, 19]. Actin filaments can be labeled with phalloidin to reveal cytoskeletal architecture and cellular shape, whereas mitochondria-specific dyes such as MitoTracker (Figure 2 E) provide insights into metabolic activity and organelle dynamics [20, 21]. In bacterial cells, like the eukaryotic cell, nucleoid stains such as DAPI, Hoechst or SYBR Green are also used to visualize DNA distribution, and markers like SYTO RNASelect (Figure 2 C) are used to highlight RNA presence in the cell [22, 23]. The outer and inner membranes can be distinguished using dyes such as FM4-64 (Figure 2 D) or TMA-DPH, which provide information on membrane integrity and permeability [24, 25].

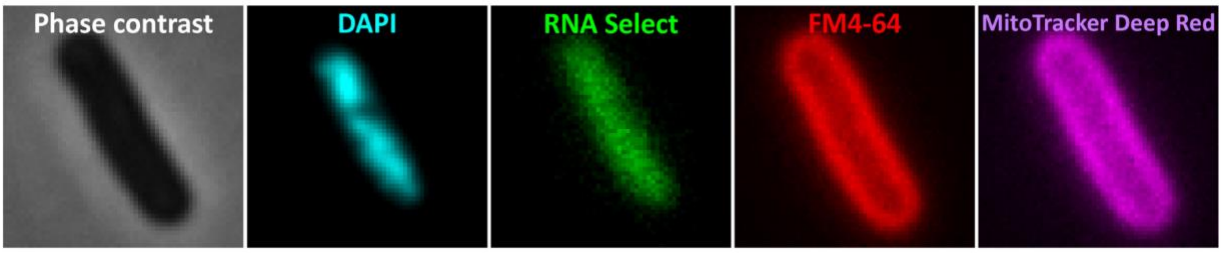


Figure 2: Multicolor fluorescence microscopy of a single cell of *E. coli* MG1655. (A) Phase contrast image showing cell morphology. (B) DAPI staining (blue) highlighting the nucleoid. (C) RNA Select staining (green) indicating RNA localization. (D) FM4-64 staining (red) marking the outer membrane. (E) MitoTracker Deep Red staining (magenta) marking the inner membrane. Image courtesy of Steemans et al., unpublished.

2.1.2 Image Acquisition

Advancements in automated imaging technologies have significantly improved the efficiency and reproducibility of image acquisition. High-content imaging (HCI) systems, commonly used in large-scale biological experiments, integrate automated fluorescence microscopy with robotic sample handling and high-throughput screening [26]. Automated imaging platforms, such as confocal and spinning disk microscopes, further enhance resolution and depth by capturing multiple focal planes and minimizing out-of-focus light [27]. In addition, deep learning-based autofocus algorithms and image processing pipelines optimize acquisition settings in real time, allowing for dynamic adaptation to sample variability [28].

Common imaging methods include brightfield, phase contrast, and fluorescence microscopy. Brightfield imaging is one of the most accessible and widely used techniques due to its speed and low cost; however, it requires careful optimization to achieve adequate contrast for cellular

structures [29]. Phase contrast imaging is particularly useful for visualizing live, unstained cells, as it enhances contrast between the cell and the background by exploiting differences in refractive indices, resulting in clearly delineated cell boundaries [30], while fluorescence imaging allows researchers to target specific cellular components with high specificity by using fluorescent dyes or tags [31]. A notable application of fluorescence microscopy is CellPainting, which uses a combination of fluorescent dyes to label multiple cellular structures simultaneously, providing a comprehensive multi-dimensional image of biological samples [1]. Each technique contributes unique advantages and challenges in capturing the features necessary for comprehensive profiling.

2.1.3 Image Segmentation

After acquiring the images, the next step is segmentation, which involves acquiring the outlines of the cells within the image. In microscopy images, for example, the background and cells are classified with distinct labels. Segmentation occurs at the pixel level to define the precise outline of each object within its class. There are three main types of segmentation: instance, semantic, and panoptic [32]. Instance segmentation detects similar objects and assigns a unique label to each one, distinguishing individual entities [32]. Semantic segmentation, on the other hand, detects similar objects but assigns the same label to all instances within a category [32]. Panoptic segmentation combines both semantic and instance segmentation, providing a comprehensive labeling of the entire image, where pixels from different instances within the same category are labeled differently [32].

For image-based profiling, instance segmentation is often the preferred choice because it allows for individual cell analysis, even in densely packed or overlapping regions [33]. Unlike semantic segmentation, which assigns a class label to each pixel without differentiating between distinct objects [34], instance segmentation ensures that each cell or structure is uniquely identified as seen in Figure 3 panel B [33]. This distinction is crucial for extracting accurate morphological and intracellular features, such as cell size and shape which are fundamental for downstream quantitative analyses. Such accurate delineation of cells ensures that biologically significant variations, like abnormal nucleoid shapes or protein distributions, are not obscured by segmentation artifacts [33]. Consequently, the accuracy of segmentation directly influences the quality of extracted measurements, making instance segmentation a critical component of robust image-based profiling.

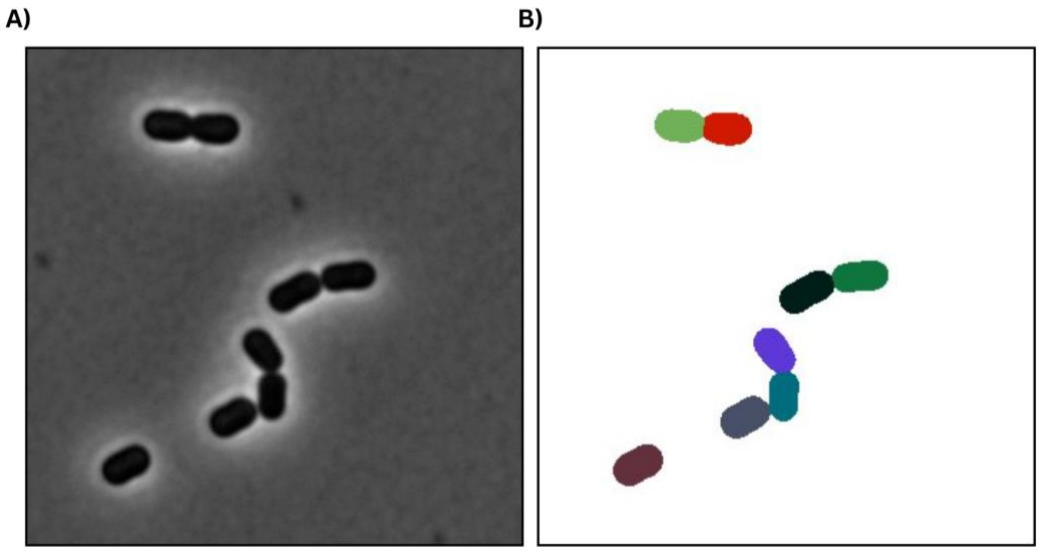


Figure 3: Instance segmentation of microbial phase contrast image. A) Phase contrast image. B) Resulting segmentation mask, each pixel composing a cell is assigned a specific integer label and color.

2.1.4 Feature Extraction

Following segmentation, features are extracted from each individual cell or region of interest to quantify cellular properties. Common features include cell shape, such as area and perimeter, texture, and intensity, measured through levels of fluorescence or other indicators [1]. These features capture information about morphology, cellular components, cell cycle states, or specific responses to treatments, offering quantitative insights into cellular function and behavior. To perform this detailed quantification, researchers rely on specialized tools, each bringing a distinct approach to feature extraction, such as CellProfiler, Fiji, and DeepCell, which address a wide range of experimental needs.

One such tool, CellProfiler, an open-source platform, facilitates feature extraction by processing images through automated, user-designed pipelines [35]. Researchers can assemble a sequence of modules to detect cells and quantify features. These modules include object identification, boundary refinement, and measurement tools, which quantify features such as area, shape eccentricity, or fluorescence intensity [35]. This automation proves invaluable for high-throughput extraction of standardized features across large datasets, making it particularly well-suited for profiling treatment effects in drug screens [35].
In contrast, Fiji, a distribution of ImageJ, offers a more hands-on approach rooted in traditional image processing techniques [36]. By relying on user-defined workflows, researchers apply filters, thresholding, and segmentation algorithms—such as watershed or edge detection—to isolate cells, then use plugins to measure properties like perimeter or texture complexity [36]. This flexibility makes Fiji a versatile tool for detailed image analysis, particularly when leveraging plugins like MorphoLibJ to study cellular morphology with high precision [37]. However, its reliance on manual adjustments can make it time-consuming and less consistent

across users, particularly when compared to automated tools like CellProfiler that streamline large-scale analyses [36, 38].

On the other hand, DeepCell harnesses the power of deep learning to extract features, training convolutional neural networks (CNNs) on annotated image sets to automatically segment and quantify cellular traits [39]. Once trained to recognize cell boundaries and patterns, it measures features like shape or intensity with remarkable accuracy, even in crowded or noisy images, and scales seamlessly to analyze thousands of cells [39]. This adaptability makes DeepCell particularly effective for identifying cell types or states in heterogeneous populations, a key strength in applications like cancer research or developmental biology [40]. Still, its dependency on substantial computational power and high-quality training datasets can limit its accessibility for smaller research groups compared to the more lightweight setups of CellProfiler or Fiji [41].

Together, these tools enable high-throughput and high-content screening, proving especially valuable in drug discovery, where extracted features illuminate a drug's activity, toxicity, or mechanism of action [2]. The choice between them depends on the scale and complexity of the analysis and the specific demands of their feature extraction goals.

2.1.5 Data Analysis

Once features are extracted, data analysis techniques are employed to identify patterns, correlations, or clusters within the cell population. This step often involves machine learning or clustering algorithms to group cells with similar characteristics, linking these groups to experimental conditions or genetic variations [38]. Analysis can focus on different scales: population-level features or single-cell-level features. Population level features, such as the average cell length per mutant, provide metrics to compare across the different experimental conditions (populations). While single-cell-level features emphasize on the variability of characteristics within a population, such as the distribution in length in a particular mutant group [42]. This level of analysis is particularly valuable for capturing heterogeneity and rare phenotypes, where the spread of features often holds more biological significance than averages [43]. Both levels of analysis represent valid approaches to studying the data, with the choice of analysis depending on the specific experimental design and objectives.

2.1.6 Image-Based Profiling Combined with Other Profiling Methods

By merging the visual power of image-based profiling with the molecular depth of proteomic, metabolomic, and transcriptomic approaches, researchers can unlock a more holistic perspective on cellular function and organization. While proteomic profiling offers insights into protein expression and interactions [44], metabolomic profiling focuses on the biochemical metabolites within a system [45], and transcriptomic profiling, particularly single-cell RNA sequencing (scRNA-seq), uncovers gene expression patterns at the single-cell level [46], these methods often lack the detailed spatial context that image-based profiling provides. Image-based profiling allows researchers to visualize how cellular components are organized and interact in situ, offering a powerful complement to these molecular profiling approaches [47].

Integrating image-based profiling with other molecular techniques can provide a more comprehensive understanding of cellular behavior. For instance, combining image-based profiling with transcriptomic approaches like scRNA-seq enables the correlation of spatial information with gene expression data, revealing not just what genes are active or not but also where and how they influence cellular organization [48]. Similarly, combining proteomic and metabolomic profiling with image-based analysis allows researchers to investigate how protein interactions or metabolic changes influence cellular morphology and organization. Such integrations can reveal biological processes that are difficult to capture using any single approach, leading to novel insights into cellular responses and behavior.

2.2 Machine Learning in Image-Based Profiling

Machine learning has become an integral part of image-based profiling, offering advanced techniques to analyze complex biological image datasets [1]. By automating feature extraction, segmentation, and pattern recognition, machine learning enables researchers to uncover cellular phenotypes and derive insights from high-content imaging data [49]. Below, we explore the key aspects of machine learning in this domain.

2.2.1 Deep Learning Models for Image Segmentation

Segmentation is a critical step in image-based profiling, as it focuses on isolating cells or structures of interest from the surrounding background. Traditionally, segmentation methods relied on techniques such as thresholding, edge detection, and region growing [50]. While these methods were effective for simpler images, they often struggled with the complexities of biological imaging [51]. Issues like noisy backgrounds, overlapping structures, and variations in image quality made these traditional approaches less effective, particularly in microscopy, where image complexity and noise levels are high [52].

Recent advancements in machine learning have transformed segmentation tasks in microscopy [53, 54]. Neural networks mimic the brain with layers of interconnected neurons that learn patterns from data [55]. These networks adjust their connections during training to recognize features, enabling them to adapt to diverse and complex inputs like microscopy images [56]. Deep learning, a subset of machine learning, employs neural networks with many layers to capture hierarchical patterns, from simple edges to intricate structures [57]. In convolutional neural networks (CNNs), a key technique is convolution, where small learnable filters scan over the image to detect spatial features such as edges, textures, and boundaries—capturing local patterns while preserving spatial relationships [56]. Building on this capability, powerful models, such as U-Net and Mask R-CNN, have emerged to tackle segmentation challenges that traditional methods could not address. These deep learning models have become the golden standard for delineating individual cells and organelles, even in crowded or complex environments, and their adaptability to diverse imaging modalities ensures high accuracy across various experimental setups [58].

U-Net, in particular, has become a cornerstone in biomedical image segmentation. Introduced with a encoder-decoder framework, U-Net is designed to map every pixel of an image to its corresponding class, enabling precise semantic or instance segmentation [53, 11]. The network follows a symmetric U-shaped structure (Figure 4), where each downsampling step in the encoder is mirrored by a corresponding upsampling step in the decoder, facilitating the recovery of spatial details [53]. Each stage of the architecture consists of two successive convolutional layers with ReLU activation, followed by a max-pooling layer in the encoder and an up-convolution (transposed convolution) in the decoder (Figure 4)[53]. Additionally, U-Net's ability to handle limited datasets, through extensive data augmentation, makes it particularly suitable for medical and microscopy imaging applications [53]. It has been widely adopted for tasks such as identifying lesions in CT and MRI scans, segmenting cellular structures in microscopy, and analyzing other biomedical images. The success of U-Net has inspired numerous variants, including models optimized for 3D imaging, multi-modal inputs, and other diverse imaging challenges, cementing its role as a key tool in medical image analysis [59].

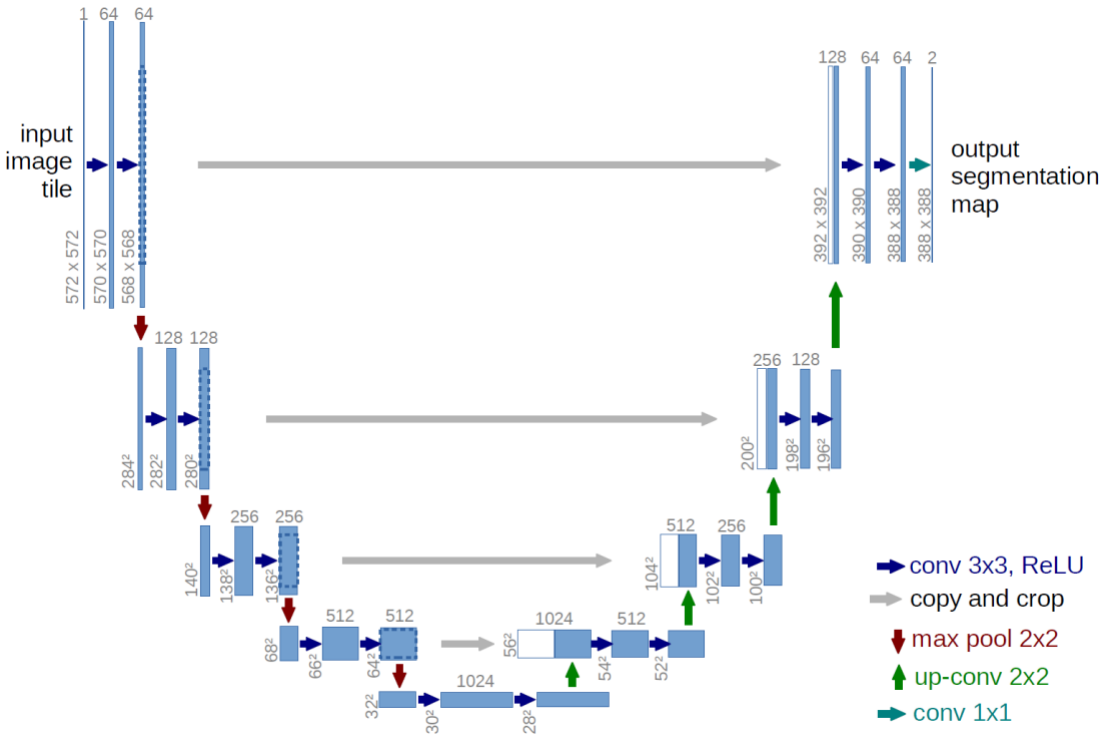


Figure 4: U-Net architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. Figure and caption adapted from Ronneberger et al., 2015.

Mask Region-Based Convolutional Neural Network (Mask R-CNN) is another widely used deep learning model for instance segmentation. It builds upon Faster R-CNN, a model built for object detection, which involves detecting and classifying things while drawing bounding boxes around them [54]. Faster R-CNN works in two stages: first, a Region Proposal Network (RPN) scans the image and suggests areas where objects might be located [54]. Second, a more refined network classifies these proposed regions and adjusts the bounding boxes to better fit each

object [54]. Mask R-CNN extends Faster R-CNN by introducing an additional segmentation branch that predicts pixel-level masks for each detected object [54]. This means that instead of merely drawing a rectangle around an object, Mask R-CNN determines the exact outline of the object.

Mask R-CNN uses Region of Interest Align (RoIAlign), a technique that ensures precise spatial alignment of feature maps (Figure 5) [54]. Feature maps are intermediate representations of an image generated by a Convolutional Neural Network (CNN) during feature extraction. Rather than processing raw pixel values directly, CNNs apply a series of convolutional operations using learnable filters (kernels) to detect meaningful patterns such as edges, textures, and object structures. Mask R-CNN has been successfully applied in cell segmentation tasks, distinguishing individual cells in dense clusters, identifying subcellular components, and analyzing diverse biomedical images with high precision [60].

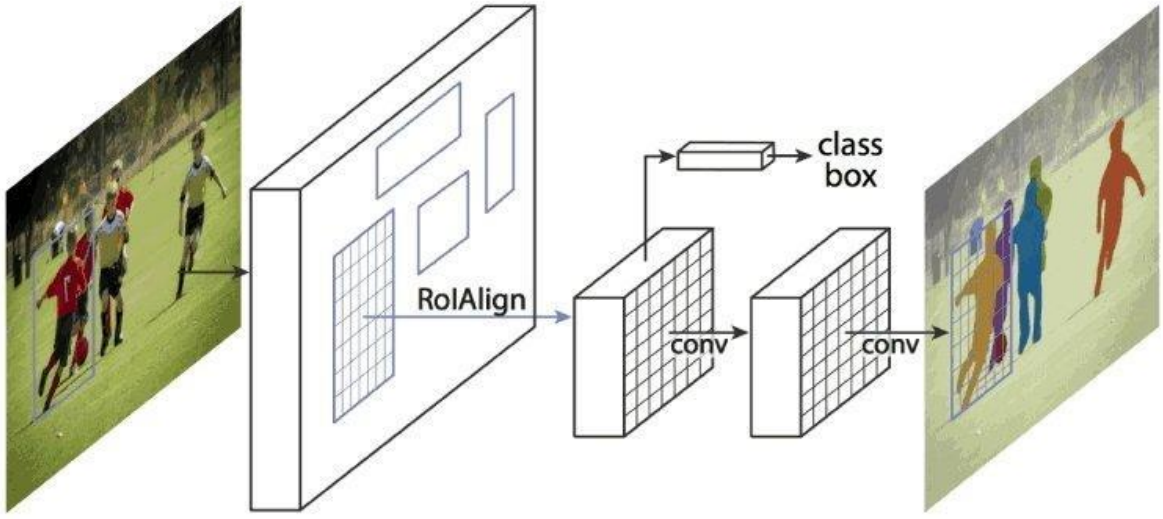


Figure 5: The Mask R-CNN framework for instance segmentation. Figure and caption adapted from He et al., 2017 [54].

2.2.2 Deep Learning for Unbiased Feature Extraction in Image-Based Profiling

Feature extraction is a foundational step in image-based profiling, converting raw image data into quantifiable metrics [61]. Traditionally, feature extraction involved defining specific morphological or textural properties based on prior biological knowledge. CellProfiler, an open-source software for image analysis, exemplifies this extraction step by enabling users to specify and extract designed features such as cell shape, size, granularity, and fluorescence intensity [35]. This approach, while effective, is often subjective and limited by human-defined parameters.

In contrast, deep learning models, particularly convolutional neural networks (CNNs), have transformed feature extraction by learning hierarchical representations directly from data, offering both precision and impartiality [61]. The key idea is that CNNs can automatically

identify the most critical features without requiring predefined descriptors. For example, in a classification task distinguishing between treatment and control cells, a trained CNN holds a rich, high-dimensional feature representation before its output layer. This feature vector, or embedding, encodes high-level patterns such as cell morphology, texture, and intensity variations, providing an unbiased and data-driven approach to feature extraction. This transition from traditional methods to automated feature extraction has been pivotal in enabling large-scale phenotypic screening, particularly in drug discovery and functional genomics, and has been increasingly more adopted [62, 63].

2.2.3 Deep Learning Architectures for Images Feature Extraction

EfficientNet, introduced by Tan and Le in 2019 [12], is a family of convolutional neural networks (CNNs) designed to achieve high accuracy with minimal computational cost. Its architecture is grounded in the concept of compound scaling, which distinguishes it from traditional CNNs that scale only one dimension, such as depth or width [12]. EfficientNet starts with a baseline model, EfficientNet-B0, and systematically scales it to produce variants (B1 to B7) that balance model complexity and performance [12]. The core innovation lies in its ability to uniformly scale three dimensions—network depth (number of layers), width (number of channels), and input resolution—using a compound scaling coefficient, φ [12]. This coefficient determines the extent of scaling, ensuring that computational resources are allocated efficiently while maximizing accuracy [12]. The architecture relies on mobile inverted bottleneck convolution (MBConv) blocks, which use depthwise separable convolutions and squeeze-and-excitation mechanisms to capture spatial patterns and recalibrate channel-wise features, respectively [12]. These blocks are organized into stages with increasing channel depth and decreasing spatial resolution, culminating in a global average pooling layer that yields a feature vector for tasks like object detection or transfer learning [12].

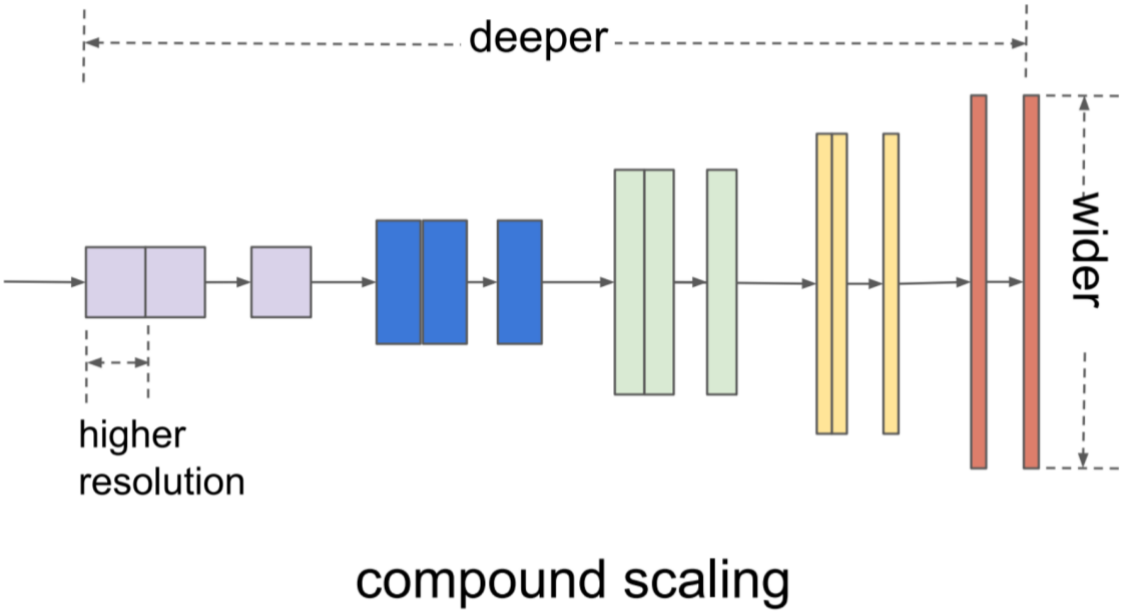


Figure 6: EfficientNet compound scaling method which uniformly scales all three dimensions with a fixed ratio. Figure and caption adapted from Tan and Le in 2019 [12].

The Vision Transformer (ViT), introduced by Dosovitskiy et al. in 2020 [64], revolutionizes feature extraction by applying transformer architectures, originally designed for natural language processing [64], to computer vision tasks. Unlike traditional CNNs like EfficientNet, ViT avoids convolutions entirely, processing images as sequences of fixed-size patches [64]. Each patch is flattened into a vector, linearly embedded, and fed into a transformer encoder with self-attention mechanisms to capture global dependencies across the image [64]. This approach enables ViT to excel in tasks requiring holistic image understanding, such as image classification [64]. ViT's architecture scales efficiently with dataset size, achieving state-of-the-art performance on large datasets like ImageNet when pre-trained [64]. Its flexibility and ability to model long-range interactions make it a powerful tool for feature extraction in deep learning applications [64].

2.2.4 Leveraging Pretrained Models for Transfer Learning in Image-Based Profiling

Transfer learning leverages pretrained models to overcome the challenges of limited data in biological research. Deep learning models trained on large-scale image datasets, such as ImageNet, capture rich feature representations that can be repurposed for biological applications, including cell classification, segmentation, and phenotypic profiling [65]. By reusing these model's learned weights, researchers can significantly reduce computational costs and data requirements, making deep learning more accessible in biomedical research [65].

A key advantage of transfer learning is its ability to extract generalizable features from large datasets and apply them to specialized tasks. For instance, a pretrained CNN can be fine-tuned by adjusting only the final layers to differentiate between healthy and diseased cells, allowing the model to leverage its existing knowledge of textures, edges, and spatial patterns while adapting to the new domain [66]. This fine-tuning process is particularly useful when labeled biological data is scarce, as it enables models to achieve high performance with fewer annotations.

Beyond standard fine-tuning, domain adaptation techniques further enhance transfer learning by addressing differences between source and target datasets. Biological images often have unique characteristics, such as high noise levels, irregular textures, and varying contrast, which may differ from the datasets used to train conventional deep learning models. Domain adaptation methods, such as adversarial learning or feature alignment, help pretrained models adjust to these variations, improving their robustness and accuracy in biological applications [66].

2.2.5 Enhancing Image-Based Profiling with Data Augmentation Techniques

Deep learning models require large datasets for effective training; however, data availability remains a challenge in bacterial image-based profiling due to labor-intensive processes like sample preparation, image acquisition, and segmentation quality control steps. To mitigate this limitation, data augmentation techniques can be employed to artificially expand datasets and improve model performance. Data augmentation involves applying transformations to existing

images to create new, slightly altered versions while preserving key biological features [67]. This approach increases dataset diversity, enhances model robustness, and mitigates issues like class imbalances and overfitting, ultimately improving generalizability [68, 53]. Common augmentation methods include geometric transformations, intensity-based modifications, noise injection, and color and channel manipulations. These techniques, such as rotation, brightness adjustment, Gaussian noise, and channel swapping, simulate real-world variations to improve model performance across diverse imaging conditions.

2.2.6 Dimensionality Reduction of High-Dimensional Phenotypic Data

Biological image analysis often generates high-dimensional feature spaces, where each image or cell is represented by a large number of extracted features, including morphological, texture, and intensity-based measurements. While these features contain valuable information, working with high-dimensional data poses challenges such as redundancy, increased computational costs, and difficulties in visualization. Dimensionality reduction techniques help address these issues by projecting data into a lower-dimensional space while preserving essential structure and variability.

Principal Component Analysis (PCA) is a widely used linear method that transforms the original feature space into a set of uncorrelated principal components, which are ordered by the amount of variance they capture [69]. This allows for efficient feature selection and noise reduction while maintaining much of the dataset's variability [69]. PCA is particularly useful for preprocessing high-dimensional image-derived data before clustering or classification tasks [69]. However, since it relies on linear transformations, it may not effectively capture complex, nonlinear relationships between features, which can limit its applicability in certain biological datasets [70].

t-Distributed Stochastic Neighbor Embedding (t-SNE), in contrast to PCA, is a nonlinear technique specifically designed for visualizing high-dimensional data in two or three dimensions [71]. By focusing on preserving local relationships, t-SNE is highly effective at revealing clusters and identifying subpopulations within biological images [72]. This makes it a popular choice for analyzing single-cell imaging data, where phenotypic variation needs to be explored [72]. However, t-SNE comes with certain drawbacks, including high computational cost, sensitivity to hyperparameters such as perplexity, and an inability to preserve global structure, meaning that relative distances between clusters may not always be meaningful [72].

Uniform Manifold Approximation and Projection (UMAP) is a more recent nonlinear method that balances local and global structure preservation while being computationally more efficient than t-SNE [73]. It constructs a high-dimensional graph representation of the data and optimally projects it into a lower-dimensional space, making it well-suited for large-scale biological datasets [74]. Compared to t-SNE, UMAP is more scalable and generally less sensitive to hyperparameters, yet it still requires tuning, particularly in determining the number of neighbors to consider [72]. Additionally, while it preserves more of the overall data structure than t-SNE, some distortions may still occur, especially when reducing data to very low dimensions [72].

Potential of Heat-diffusion for Affinity-based Trajectory Embedding (PHATE) is a nonlinear dimensionality reduction method that captures both local and global structures in high-dimensional biological image data [75]. Using a diffusion-based approach, PHATE reveals continuous trajectories and branching patterns, ideal for visualizing phenotypic transitions in single-cell imaging [75]. Compared to t-SNE and UMAP, PHATE offers more interpretable visualizations with less sensitivity to hyperparameters, though it can be computationally intensive for large datasets [75]. Its ability to model intrinsic data geometry makes it particularly effective for studying dynamic biological processes.

2.2.7 Classification of Cellular Phenotypes

Classifying cellular phenotypes enables the identification of cellular states, responses to treatments, and genetic variations [76]. Classification involves supervised learning techniques that assign predefined labels to cells based on extracted features. In this approach, a model is trained on labeled datasets where each cell has an associated phenotype category, such as normal versus cancerous cells or drug-treated versus untreated conditions. Various machine learning algorithms are used for this task, each with strengths suited to different types of cellular data.

Random Forest is an ensemble learning method that constructs multiple decision trees and aggregates their outputs to improve accuracy and robustness [77]. It is well-suited for high-dimensional cellular imaging data, particularly when feature selection and interpretability are important [78]. The algorithm generates multiple decision trees, each trained on a random subset of the data and features [77]. By averaging their predictions or taking a majority vote, Random Forest reduces the risk of overfitting and improves generalization [77]. One of its key advantages is the ability to provide feature importance scores, allowing researchers to identify the most relevant cellular characteristics influencing classification outcomes [77]. However, while Random Forest performs well on structured datasets, it may struggle with very high-dimensional raw image data unless combined with dimensionality reduction techniques [79].

Support Vector Machines (SVMs) are powerful classifiers that work by finding an optimal hyperplane that separates different classes in a dataset [80]. In cellular phenotype classification, SVMs are particularly effective when phenotypic differences are subtle and complex [81]. The algorithm transforms input data into a higher-dimensional space using kernel functions, enabling it to separate non-linearly distributed classes [80]. The choice of the kernel, such as linear, polynomial, or radial basis function (RBF), significantly impacts performance [80]. SVMs are widely used in biomedical applications because they handle small to medium-sized datasets well and are less prone to overfitting compared to deep learning models [82]. However, they require careful tuning of hyperparameters and may become computationally expensive with very large datasets.

Gradient boosting methods, such as XGBoost and LightGBM, build strong predictive models by sequentially improving weak learners [83]. These algorithms construct decision trees iteratively, where each tree corrects the errors made by the previous ones [83]. They are

particularly effective when feature selection is crucial and interpretability is needed [83]. In cellular phenotype classification, gradient boosting methods excel at handling tabular datasets derived from image features, providing high accuracy and robustness against noise [84]. Compared to Random Forest, gradient boosting methods tend to be more computationally efficient and offer fine-tuned control over model performance through hyperparameter optimization [85]. However, they can be prone to overfitting if not properly regularized and require careful tuning to balance complexity and generalization like any other algorithm.

Neural networks, particularly deep learning architectures such as Convolutional Neural Networks (CNNs), have gained prominence in cellular phenotype classification due to their ability to learn complex, hierarchical patterns directly from raw images [86]. CNNs use layers of convolutional filters to automatically extract spatial features from microscopy images, eliminating the need for manual feature engineering [86]. This makes them particularly well-suited for tasks involving large-scale image-based profiling, where variations in cell morphology and texture are critical [87]. However, they require large amounts of labeled data and significant computational resources for training [86]. Fully connected neural networks, though less commonly used for direct image classification, can be effective when applied to structured datasets derived from extracted image features [88]. Despite their advantages, deep learning models are often considered black boxes, making interpretability a challenge.

The choice of classification algorithm depends on factors such as dataset size, feature dimensionality, and the complexity of phenotypic variation. While traditional models like Random Forest and SVMs perform well on structured feature sets, deep learning methods are often preferred for raw image-based classification.

2.2.8 Clustering of Cellular Phenotypes

Clustering is an unsupervised learning approach that groups similar cells based on extracted features, uncovering hidden patterns without predefined labels [89]. This method is particularly valuable for identifying novel cellular states or phenotypic transitions that might not be captured in predefined categories. Several clustering algorithms are commonly applied in cellular phenotype analysis:

k-Means Clustering is one of the simplest and most widely used clustering algorithms. It partitions a dataset into a predefined number of clusters, k, by minimizing intra-cluster variance [90]. The algorithm begins by randomly selecting k cluster centroids and then iteratively assigns each data point to the nearest centroid based on Euclidean distance [90]. After assignment, the centroids are updated by computing the mean position of all points in each cluster, and this process repeats until convergence [90]. k-Means is computationally efficient and works well when the clusters are roughly spherical and evenly sized [90]. However, its main limitation is the assumption that clusters are of equal variance and shape, which may not hold in complex biological datasets where cellular phenotypes often exhibit diverse and overlapping characteristics [91]. Additionally, the algorithm requires the number of clusters to be defined in advance, which can be challenging when analyzing unknown phenotypic variations.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a clustering method particularly well-suited for datasets where clusters vary in density [92]. Unlike k-Means, which assumes that clusters are spherical, DBSCAN identifies dense regions in the feature space and groups them together, making it robust for analyzing heterogeneous cellular populations [92]. The algorithm works by defining a neighborhood radius around each data point and counting the number of points within that radius [92]. Points in high-density regions form core clusters, while points that are close to core clusters are considered part of the cluster boundary [92]. Any point that does not belong to a dense region is classified as noise or an outlier [92]. DBSCAN is especially useful in cellular phenotype studies where rare or abnormal cell states exist, as it can naturally separate outliers without requiring a predefined number of clusters [92]. However, its performance depends on carefully chosen parameters, such as the neighborhood radius and minimum number of points required to form a cluster [92].

Gaussian Mixture Models (GMMs) provide a probabilistic approach to clustering by modeling data as a mixture of multiple Gaussian distributions [93]. Unlike k-Means, which assigns each point to a single cluster, GMMs compute the probability that each data point belongs to multiple clusters, making them well-suited for biological data where phenotypic variations often exhibit overlapping characteristics [94]. The algorithm estimates the parameters of each Gaussian distribution using the Expectation-Maximization (EM) algorithm, iteratively refining the cluster assignments until the model converges [93]. GMMs are particularly effective when cellular phenotypes transition gradually rather than having well-defined boundaries [95]. However, similar to k-Means, the number of clusters must be specified in advance, and the model assumes that the data follows a Gaussian distribution, which may not always be the case in real-world biological datasets.

Hierarchical Clustering is a tree-based method that constructs a hierarchy of clusters based on pairwise similarity measures [96]. It can be performed in two ways: agglomerative (bottom-up) or divisive (top-down) [96]. In agglomerative clustering, each data point starts as its own cluster, and the algorithm iteratively merges the most similar clusters until a single cluster is formed [96]. In divisive clustering, the process starts with all data points in a single cluster, which is then recursively split into smaller clusters [96]. One of the major advantages of hierarchical clustering is that it does not require the number of clusters to be specified beforehand, making it well-suited for exploratory analysis [96]. The resulting dendrogram provides a visual representation of the relationships between cellular subtypes, allowing researchers to determine cluster groupings based on the desired level of granularity [96]. However, hierarchical clustering can be computationally expensive, particularly for large datasets, and the choice of linkage criteria (such as single linkage, complete linkage, or average linkage) can significantly impact the results [97].

Clustering methods are frequently combined with classification techniques to enhance analysis. For example, clustering can provide exploratory insights into unknown phenotypic groupings, while classification models can later be trained to recognize these groups in new datasets.

2.3 Single Cell Bacterial Image-Based Profiling

Image-based profiling of bacterial cells offers a way to analyze and characterize cellular morphology, cell cycle, growth patterns, and responses to various environmental or drug treatments. This section discusses the challenges of profiling bacterial cells, the key features used for analysis, the relationship between cell cycle and morphology, and the biological significance of these profiling efforts.

2.3.1 Image-based Profiling for Bacteria: Challenges and Opportunities

Profiling bacteria through imaging presents challenges, some not commonly encountered when studying eukaryotic cells. These challenges stem from the small size and variable morphology of bacteria, as well as limitations in image resolution, contrast, and fluorescent staining and labeling. As a result, generating high-quality, reproducible data from bacterial images often requires specialized imaging techniques, preprocessing steps, and analytical approaches.

Small Size and Image Resolution Limitations
Bacteria are significantly smaller than eukaryotic cells, with the average diameter of E coli being 0.99 micrometers [98]. These dimensions can vary significantly among different bacterial species and are influenced by factors such as nutrient availability, genetic regulation, and environmental conditions [99]. This small size imposes limitations on conventional microscopy techniques, such as light microscopy, where the diffraction limit of light (approximately 200 nanometers) hinders standard optical resolution from capturing detailed morphological features like the cell wall, the chromosomal DNA, or the divisome [100]. To address this, high-resolution imaging techniques such as super-resolution microscopy (e.g., STED, SIM, PALM/STORM) or electron microscopy (e.g., SEM and TEM) are viable options [101]. However, these methods can be costly, time-consuming, and impractical for high-throughput bacterial profiling in contrast to brightfield, phase contrast, or conventional fluorescent imaging, which remain more accessible despite their lower resolution.

Morphological Cell Variability
In addition to their small size, bacterial cells exhibit significant morphological heterogeneity (Figure 7), even within genetically identical populations [102]. This variability arises due to multiple factors, including nutrient availability, environmental stress, cell cycle stage, and genetic mutations [103, 104]. Some bacterial species can switch between rod-shaped, spherical, filamentous, or irregular morphologies in response to external stimuli [105]. For example, Streptomyces coelicolor transitions from branching filamentous hyphae to spherical spores under nutrient-limited conditions, while Helicobacter pylori can adopt a coccoid form under stress conditions [106, 107]. Additionally, bacteria at different cell cycle stages can exhibit distinct shapes and sizes [108]. For instance, during cell division, rod-shaped bacteria like Escherichia coli elongates before septation, leading to temporary morphological changes [109]. This variability in morphology can impact the accuracy of cell segmentation models, particularly those relying on machine learning or deep learning approaches. These models are often trained on datasets that may not fully capture the breadth of morphological diversity,

leading to challenges in accurately identifying and segmenting cells with atypical shapes or sizes.

Figure 7: Morphological diversity in α-proteobacteria. Figure adapted from Govers, S. K., & Jacobs-Wagner, C. (2020) [104].

Segmentation of Bacterial Brightfield Images

Another challenge is that brightfield microscopy—commonly used for bacterial imaging—often produces low-contrast images [9]. This is particularly problematic because the optimal focus in brightfield microscopy is typically either slightly above or below the sample plane rather than directly on it. When the focus is directly on the sample, bacterial cells become nearly invisible due to the lack of contrast between the cells and the background. This phenomenon occurs because brightfield microscopy relies on light absorption and scattering [110], which are minimal for small, transparent bacterial cells. As a result, the cells are often only visible when the focus is slightly offset, creating a small shadow effect. This can make it difficult to accurately segment individual cells, especially in dense bacterial cultures. In addition, noise from cell debris and background artifacts can further impact segmentation accuracy [111].

Despite these limitations, brightfield imaging remains a favored method for large-scale bacterial microscopy screens due to its speed, simplicity, and cost-effectiveness [112]. Furthermore, brightfield microscopy offers a significant advantage over phase-contrast microscopy for bacterial segmentation due to its inherent lack of halo artifacts around cell boundaries. While phase-contrast microscopy excels at visualizing transparent specimens by converting phase shifts into amplitude variations, it often introduces prominent halo effects around cell edges [113]. These halos can lead to inaccurate cell size measurements and obscure the boundaries of closely packed cells, making precise segmentation challenging, especially in dense bacterial communities. In contrast, brightfield images, despite their lower intrinsic contrast, generally present defined cell boundaries without such artifacts, which can be beneficial for developing more robust and accurate segmentation models, particularly when coupled with advanced image processing and machine learning techniques.

However, despite its widespread use and advantages, the availability of robust brightfield segmentation models specifically designed for bacterial images remains limited, posing a significant bottleneck for accurate and scalable high-throughput analysis.

Fluorescently Labeling Bacterial Subcellular Structures

Unlike eukaryotic cells, bacteria lack membrane-enclosed organelles [114], leading to a non-compartmentalized intracellular space where fluorescent markers, such as dyes or protein tags, must directly target features like the nucleoid, cell membrane, or specific proteins. This structural constraint results in diffuse cytoplasmic signals, complicating the precise localization of labeled structures in bacterial image-based profiling [115]. Fluorescent protein tagging, often using plasmid-based systems to introduce markers like green fluorescent protein (GFP), enables tracking of protein localization and dynamics but is less common in high-throughput screens due to plasmid instability [116]. The small, densely packed nature of bacterial cells causes fluorescent signals to overlap, hindering clear resolution of individual structures and challenging segmentation in dense cultures [115]. Moreover, high dye concentrations or protein overexpression can disrupt bacterial physiology, altering growth, division, or gene regulation, which may skew phenotypic data in large-scale microscopy experiments [117]. Variability in labeling efficiency across samples, driven by differences in membrane permeability or metabolic states, further complicates reliable signal detection, particularly in high-throughput settings [118]. These challenges necessitate advanced image processing and dye optimization strategies to enhance signal clarity and ensure accurate phenotypic analysis.

2.3.2 Key Cellular and Subcellular Characteristics to Phenotype Bacteria

Bacterial profiling relies on the ability to identify and quantify specific cellular components that reflect cell cycle progression, physiological states, and stress responses. By analyzing bacterial morphology, nucleoid organization, and coordination of other intracellular structures researchers can infer critical biological processes such as DNA replication, cell division, and adaptation to environmental stressors.

Cell Shape and Size

Morphological features like cell length, width, and aspect ratio (length-to-width ratio) are used for studying bacterial growth and division. For example, in E. coli cell constriction only occurs after a constant cell length has been reached [109]. Additionally, size and shape abnormalities can reveal bacterial responses to different nutrient conditions, antibiotic exposure, or other external stimuli, making morphology a key parameter for profiling bacterial populations [119].

The Bacterial Nucleoid

Closely linked to cell morphology, the nucleoid in bacterial cells is a region where the genetic material is located, typically in the form of a single circular chromosomal DNA molecule (Figure 8C). Unlike eukaryotic cells, bacteria do not have a membrane-bound nucleus, and the nucleoid region is not enclosed. The structure and organization of the nucleoid can vary depending on the bacterial species and environmental conditions [120]. During cell division, the nucleoid undergoes dynamic changes to ensure that the genetic material is accurately distributed between daughter cells [121]. In some bacteria, the nucleoid forms distinct foci that are involved in DNA replication and segregation [122]. In profiling bacteria, variations in nucleoid morphology, such as changes in DNA compaction or foci formation, can indicate critical events in cell growth, division, and response to stress [123]. For instance, an enlarged nucleoid may signal a delay in cell division, possibly due to DNA replication stress or DNA damage response [124]. Additionally, disruptions in nucleoid integrity can be indicative of cellular stress, antibiotic resistance, or genomic instability [125].

The duplication of the bacterial genome prior to cell division is ensured by the coordinated process of DNA replication. Initiator proteins, such as DnaA, bind and unwind the double-stranded DNA to generate a replication fork at a specified location known as the origin of replication (oriC) [126]. The enzyme DNA helicase further unwinds the DNA, separating the two strands [126], while single-strand binding proteins stabilize the exposed single-stranded DNA [126]. DNA polymerase III, the primary replication enzyme in bacteria, synthesizes new DNA strands by adding nucleotides complementary to the template strands in the 5’ to 3’ direction [126]. Since DNA replication is semi-continuous, the leading strand is synthesized continuously, while the lagging strand is formed in short segments called Okazaki fragments, which are later joined by DNA ligase [126]. This process ensures that bacterial cells can rapidly divide and maintain genetic integrity [126].

Intracellular organization of the divisome in dividing bacteria

In addition to the nucleoid, the divisome is another critical complex assembled in preparation for cell division. It centers around the protein FtsZ, which polymerizes to form the Z ring at the midcell, marking the division site (Figure 8D) [127]. The divisome, anchored by the Z ring, recruits a suite of division proteins, including FtsA and ZipA, to build the septum and drive cell separation during binary fission [127]. Its dynamic assembly is tightly regulated by the cell cycle, ensuring division occurs precisely in time and space. Studying the divisome’s formation and stability under different conditions—like nutrient stress or antibiotic exposure—reveals

how bacteria manage division and adapt to challenges [128]. Disruptions in divisome function, such as malformed or mislocalized Z rings, can indicate division defects and hinder bacterial growth. For instance, antibiotics targeting FtsZ polymerization disrupt divisome assembly, making it a key focus for understanding susceptibility and resistance mechanisms [129].

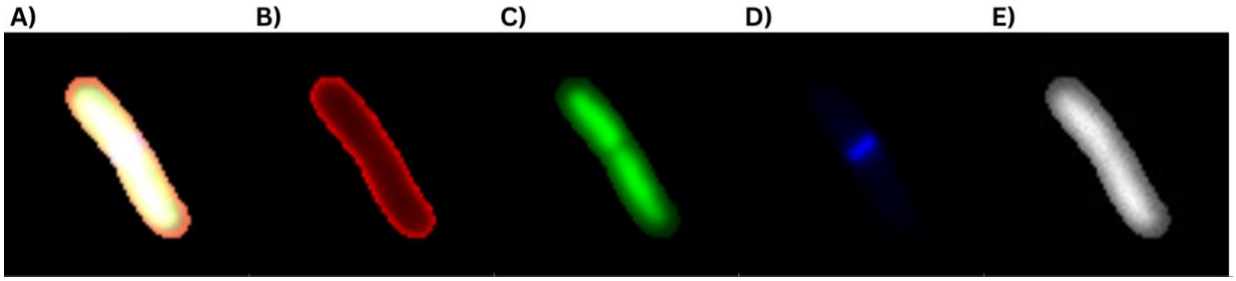


Figure 8: Splitted multi-channel single cell *E. coli*. A) Composite image. B) Phase-contrast. C) DAPI staining DNA. D) FtsZ labeled with Venus sandwich fusion fluorescent marker. E) SeqA protein labeled with mCherry fluorescent marker.

Composition and structure of the bacterial Cell envelope

As the primary structural barrier, the bacterial cell wall is a fundamental structural component that provides mechanical strength and protection against environmental stress [130]. Differences in cell wall thickness and composition serve as key indicators of bacterial adaptation, stress responses, and antibiotic resistance mechanisms [130]. For instance, Gram-positive bacteria possess a thick peptidoglycan layer (Figure 9, left side), which enhances their resistance to osmotic stress but simultaneously makes them more susceptible to β-lactam antibiotics that inhibit peptidoglycan synthesis [131]. In contrast, Gram-negative bacteria have a thinner peptidoglycan layer but are equipped with an outer membrane containing lipopolysaccharides (LPS) (Figure 9, right side), which provides an additional barrier against antibiotics and immune system attacks [132].

Beyond these general structural differences, some bacteria can dynamically modify their cell wall composition in response to environmental stress or antibiotic exposure. This adaptability can lead to the formation of cell wall-deficient, or L-form, variants that lack a rigid peptidoglycan layer, allowing them to evade immune detection and persist under unfavorable conditions [133]. While the direct measurement of cell wall thickness typically requires electron microscopy, alternative methods such as fluorescence microscopy can offer indirect insights. Fluorescent dyes, including FM4-64 for membrane visualization and Van-FL for peptidoglycan labeling, provide valuable tools for studying cell wall architecture in live-cell imaging [134].

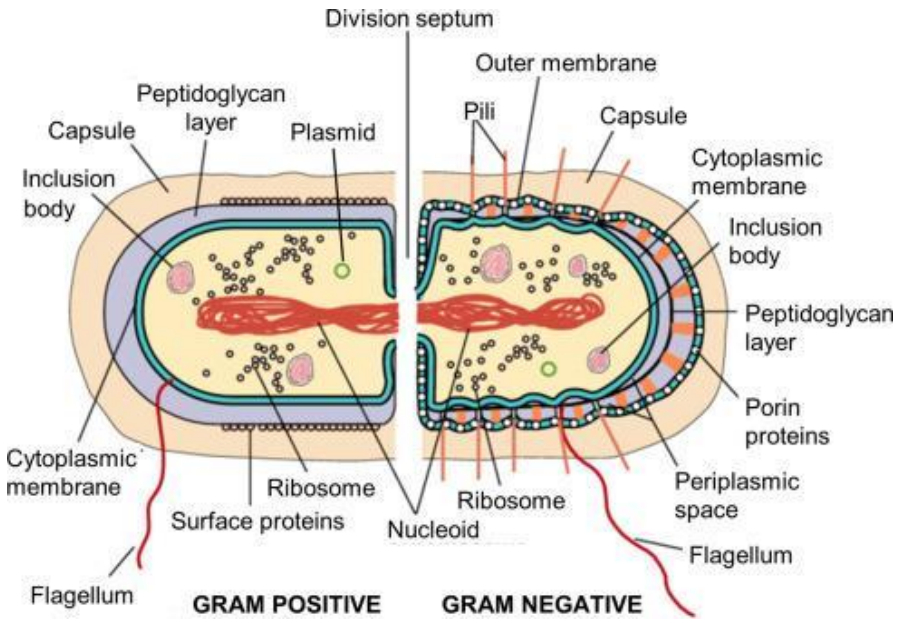


Figure 9: Bacteria Cell Structure in Gram Positive and Gram Negative [135].

2.3.3 Cell Morphology and Cell Cycle Relationship

Bacteria exhibit remarkable adaptability in coordinating growth and cell division, despite lacking the intricate checkpoint mechanisms found in eukaryotic cells, such as cyclins and cyclin-dependent kinases (CDKs) [136]. Instead of relying on dedicated regulatory proteins to control cell cycle transitions, E. coli and other bacteria use size-dependent rules to maintain robust proliferation across diverse environmental conditions [109].

The bacterial cell cycle is a tightly regulated process that enables bacterial cells to grow, replicate their DNA, and divide into two daughter cells. Unlike eukaryotic cells, which undergo mitosis, bacteria reproduce primarily through binary fission, a simpler yet highly coordinated process. The bacterial cell cycle consists of three main stages (Figure 10): B period (growth phase), C period (DNA replication), and D period (cell division) [137]. During the B period, the cell increases in size, synthesizing essential macromolecules and preparing for DNA replication [137]. The C period involves the duplication of the bacterial chromosome, which begins at a specific origin of replication (OriC) and proceeds bidirectionally until the entire genome is copied [137]. Finally, in the D period, cytokinesis is initiated by the assembly of the Z-ring [137].

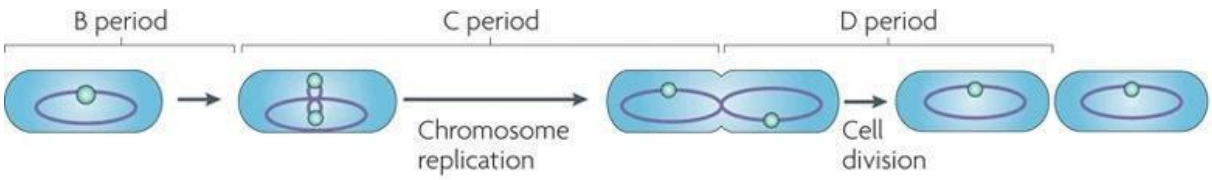


Figure 10: Bacteria Cell Cycle [137].

The Z-ring (FtsZ ring) is functionally analogous to the eukaryotic contractile ring and serves as the primary determinant of division site placement [138]. The assembly of the FtsZ ring marks the initiation of cytokinesis, ensuring that division occurs precisely once per cell cycle [139]. Unlike DNA replication, which can initiate multiple times in rapidly dividing cells, FtsZ ring formation is spatiotemporally constrained [109]. In E. coli, two primary factors regulate its formation: (1) a minimum cell length threshold (~2.84 μm in slow-growing cells), and (2) generational timing, wherein the ring forms only once per division cycle, even if multiple replication rounds have begun [109].

DNA replication follows a volume-based initiation mechanism, ensuring a constant number of replication origins relative to cell size [109]. This suggests that bacteria rely on biophysical rather than biochemical checkpoints to regulate cell cycle transitions. As replication progresses, the nucleoid must segregate before division can proceed. Studies indicate that nucleoid constriction occurs at a fixed length, highlighting a potential role for chromosomal organization in determining the timing of cytokinesis [109]. In nutrient-rich environments, nucleoid occlusion mechanisms prevent premature FtsZ ring formation, ensuring that division does not occur until chromosome segregation is complete [109]. However, under nutrient-poor conditions, division can proceed despite incomplete segregation, suggesting adaptive flexibility in division timing [109].

A key feature of bacterial size regulation is the timing of constriction and division. Unlike eukaryotic cells, which coordinate cytokinesis through complex signaling pathways, E. coli appears to employ a growth-dependent division strategy. Experimental evidence indicates that cell constriction begins after a constant length increment is added, rather than occurring at a fixed time post-birth [109]. Once constriction is initiated, division follows a fixed time delay (~15 minutes at 37°C), reinforcing the hypothesis that constriction serves as a key regulatory checkpoint in bacterial proliferation [109].

2.3.4 Applications of Bacterial Image-Based Profiling

Profiling bacterial cells through imaging holds the potential for advancing microbiology, drug discovery, and environmental studies. By analyzing bacterial morphology, researchers can uncover insights into cellular behavior, response to stimuli, and overall population dynamics. This approach enables a deeper understanding of bacterial physiology, helping to address various challenges in healthcare, industry, and ecological research. Below are some of the primary potential applications of bacterial profiling.

Mode-of-Action Prediction of Antibiotics in Drug Discovery

As an application of bacterial image-based profiling, mode-of-action (MoA) prediction plays a central role in antimicrobial drug discovery. By capturing antibiotic-induced morphological changes in bacterial cells, high-content imaging allows for the extraction of rich phenotypic fingerprints that reflect underlying biological responses. These profiles, when analyzed using machine learning, can accurately classify antibiotics based on their MoA, helping distinguish

novel compounds from those with known activities. Early work in this area by Nonejuie et al. (2013) demonstrated the effectiveness of bacterial image-based profiling for MoA prediction using a small set of engineered features—including area, perimeter, length, width, circularity, and fluorescent pixel intensity of both the membrane and nucleoid. Their study successfully revealed the mode of action of spirohexenolide in methicillin-resistant Staphylococcus aureus [140]. More recently, convolutional neural networks (CNNs) have significantly enhanced the accuracy and efficiency of mode-of-action prediction. One notable example is MycoBCP, a CNN-based adaptation of BCP developed for Mycobacterium tuberculosis, which achieved an impressive 96% accuracy in identifying the mechanisms of action of various antimicrobial compounds [141]. Furthermore, a supervised deep learning approach utilizing CNNs was applied to brightfield microscopy images of E. coli, demonstrating remarkable accuracy in predicting antibiotic modes of action even at subinhibitory concentrations and successfully identifying novel modes of action, thus paving the way for automated antibiotic discovery [142]. These developments mark a significant step toward more data-driven, automated strategies for antibiotic discovery and characterization.

Functional Genomics

Beyond mode-of-action prediction, bacterial image-based profiling is emerging as a powerful and relatively novel approach in functional genomics. It enables researchers to systematically probe gene function in bacterial species by capturing high-content phenotypic changes in response to genetic perturbations. Central to this strategy are genome-wide gene knockout libraries, such as the E. coli Keio collection, which facilitate the systematic deletion of individual genes to observe resulting phenotypic effects. A compelling example of this approach is provided by Sondervorst et al. (2025), who used cross-condition image-based profiling of E. coli deletion strains to investigate the roles of genes with previously unknown functions [143]. Their study identified five genes whose deletions produced nutrient-independent or functionally informative phenotypes, such as changes in cell size, growth rate, or subcellular protein localization. These findings underscore how image-based profiling can bridge the genotype-phenotype gap and accelerate bacterial gene function annotation, ultimately advancing our understanding of bacterial physiology, pathogenesis, and environmental adaptation.

Understanding Evolutionary Adaptations

Bacterial populations undergo continuous adaptation to environmental pressures, leading to changes in cell morphology that enhance survival [144]. Profiling bacterial cells over time can reveal evolutionary trends, such as increased resistance to desiccation, adaptation to extreme temperatures, or enhanced motility mechanisms [145, 146]. These adaptations provide valuable insights into microbial ecology and evolutionary biology, helping scientists understand how bacteria persist in diverse habitats, including deep-sea vents, arid deserts, and the human microbiome [147, 148]. Furthermore, studying bacterial morphological evolution can inform synthetic biology and bioengineering efforts, where understanding natural adaptation strategies can inspire the design of robust, engineered microbial systems [149].

3 Materials and Methods

The source code for all scripts used in this study is publicly available on GitHub. Scripts for training the Brightfield Segmentation models are hosted at https://github.com/tgaspe/Bacterial_Brightfield_Segmentation. Scripts for training and data analysis of the EfficientNet-based feature extraction model are available at https://github.com/tgaspe/Bacterial_EfficientNet_Feature_Extraction. These repositories contain further documentation and instructions for reproducing the experiments described in this thesis.

3.1 Brightfield Segmentation Model

To train a bacteria brightfield image segmentation model, a machine learning workflow was employed, as shown in Figure 11. The dataset consisted of paired phase contrast and brightfield images of bacterial species with diverse morphologies, including *Phocaeicola vulgatus*, *Leuconostoc mesenteroides*, *Escherichia coli*, *Bacillus subtilis*, and *Bacillus thuringiensis*. The process began with segmenting the phase-contrast images to create ground-truth masks. The dataset was then split into three subsets: training, validation, and test sets. The training set images, originally 2720x2720 pixels, were tiled into smaller 512x512-pixel squares. Tiles with fewer than 10 cells were filtered out to prevent training errors. Next, three different training sets were created with 1000, 2000, and 2501 images to study the effect of dataset size on model performance. The models were trained on these sets, and the validation and test set brightfield images were fed to the trained models to generate predicted masks. These predicted masks were compared to their counterpart ground-truth masks using the Jaccard index metric to evaluate their performance.

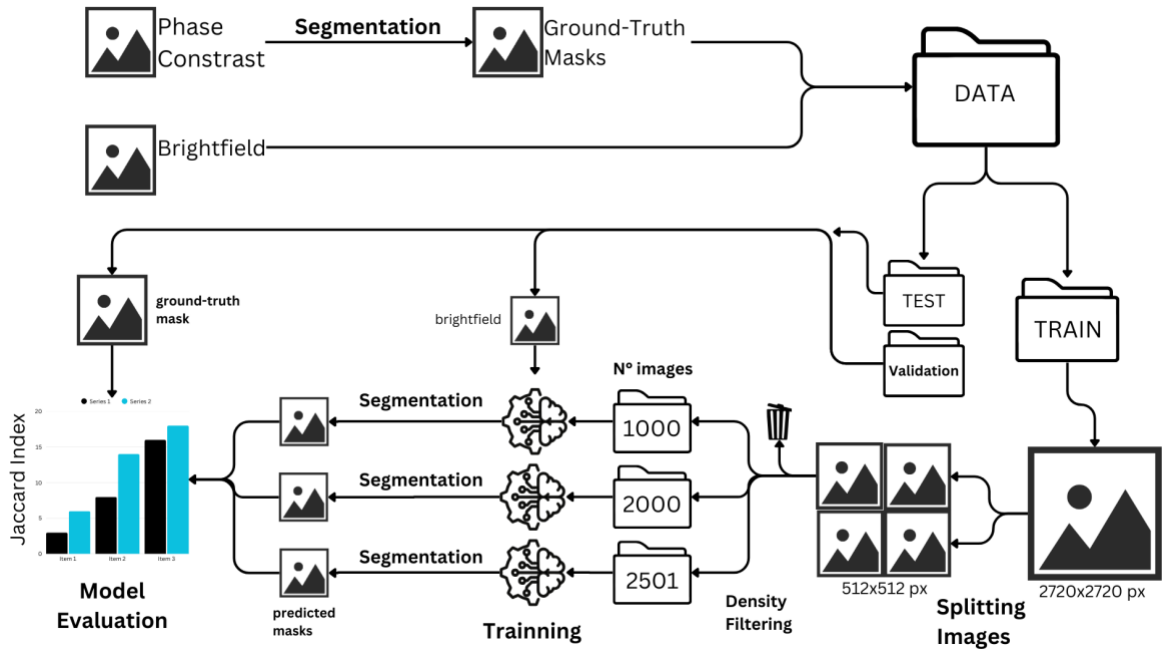


Figure 11: Brightfield Segmentation Model Training Workflow.

3.1.1 Ground Truth Generation by Segmenting Phase Contrast Images

Phase-contrast images were segmented using Omnipose to produce ground-truth masks for training our segmentation models. Omnipose is an advanced deep learning-based tool designed for cell segmentation, extending the capabilities of Cellpose by improving accuracy for densely packed or non-standard cell shapes, such as bacterial cells, through enhanced mask reconstruction dynamics and flow field predictions [11]. The Omnipose algorithm was configured with default parameters, except for the mask threshold, set to 1 to require stronger evidence for seeding cell masks, and the flow threshold, set to 0 to disable flow error checking and accelerate processing. The pretrained bact_phase_omni model, provided by Omnipose, was utilized for this segmentation step.

3.1.2 Dataset Split

The brightfield images and their corresponding ground-truth masks were divided into training, validation, and test sets with approximate proportions of 75%, 10%, and 15%, respectively. This split resulted in 123 images for the training set, 15 for the validation set, and 25 for the test set. To ensure unbiased evaluation of the trained segmentation models, the validation and test sets included distinct bacterial species not present in the training set. The training set included cells of *Escherichia coli*, *Bacillus subtilis*, and *Bacillus thuringiensis* species. The validation set consisted of *Phocaeicola vulgatus* cells, while the test set comprised *Leuconostoc mesenteroides* cells.

3.1.3 Image Tiling

Following the data split, the brightfield training set images and their associated ground-truth masks, originally sized at 2720 × 2720 pixels, were tiled into smaller 512 × 512 pixel squares using the Pillow Python package. This tile size adheres to Omnipose documentation recommendations for optimal model training. The 512 × 512 pixel tiles are memory-balanced, as they are small enough to fit efficiently within the memory constraints of most modern GPUs during training, while still being large enough to capture sufficient contextual information about multiple cells or objects in each tile. This balance ensures stable training without excessive memory demands, allowing the model to process batches effectively. The tiling process generated a total of 3075 brightfield tiles from the 123 training images.

3.1.4 Cell Density Filtering

To mitigate sparse density errors during training, tiles containing fewer than 10 cells were excluded. This filtering process involved loading the mask tiles with Pillow, converting them to NumPy arrays, and counting the unique cell instances in each mask. As a result, 574 tiles and their corresponding masks were removed, leaving 2501 tiles for the final training dataset.

3.1.5 Training and Hyperparameter Tuning

To assess the impact of training dataset size on segmentation model performance, the training set was divided into three subsets containing 1000, 2000, and 2501 images, respectively. The Python random package was used to randomly sample images for each subset, ensuring an unbiased selection process. These subsets were then used to train segmentation models under two conditions: (1) transfer learning, leveraging the pretrained bact_phase_omni model from Omnipose [11], and (2) training from scratch without transfer learning. This comparison aimed to evaluate the effectiveness of pretrained weights in improving model accuracy and convergence.

Additionally, the 2501-image subset with transfer learning was used to investigate the effect of batch size on training dynamics. Three batch sizes—50, 100, and 200—were tested to determine their influence on model performance and stability. Each model configuration was trained for multiple epochs, with checkpoints saved every 20 epochs to monitor progress and enable subsequent analysis.

3.1.6 Brightfield Segmentation Masks Generation

Each trained model was applied to segment the brightfield validation set, facilitating performance evaluation across various training configurations. Initially, segmentation masks were generated using Omnipose's default parameters (mask threshold = 0.0, flow threshold = 0.4).

To optimize segmentation quality for bacterial cells in brightfield images, the mask threshold and flow threshold parameters were tuned using the transfer learning model trained on the 2501-image subset, with weights saved at epoch 220. The mask threshold, which determines the seeding of cell masks by thresholding the distance transform output (a map of pixel distances from cell boundaries), was tested over a range of -2 to 2 with increments of 1. Lower values increase sensitivity by allowing more pixels to initiate masks, potentially capturing faint or incomplete cells, while higher values require stronger evidence, reducing over-segmentation. The flow threshold, which filters masks by comparing the predicted flow field (a pixel-wise map guiding mask boundary formation) to the "true" flow derived from the masks, was varied from 0.0 to 0.6 with increments of 0.1. This parameter sets the maximum allowable average flow error per pixel; a value of 0.0 disables the check for speed, accepting all masks, while higher values (e.g., 0.6) relax the consistency requirement, retaining masks with greater flow discrepancies. These ranges were explored to increase segmentation accuracy.

3.1.7 Segmentation Evaluation Metrics

Model performance was assessed using the Intersection over Union (IoU), also known as the Jaccard Index, a widely used metric for evaluating segmentation accuracy. The IoU is defined as the ratio of the overlapping area between the predicted and ground-truth regions to their combined area, expressed as:

$$IoU = \left|\frac{A \cap B}{A \cup B}\right| \quad (1)$$

where **A** represents the predicted segmentation mask, **B** is the ground-truth mask, $|A \cap B|$ is the number of intersecting pixels, and $|A \cup B|$ is the union of the pixels in both masks.

The segmentation masks generated by each model for the brightfield validation and test sets were compared to their corresponding ground-truth masks, derived from phase-contrast image segmentation. For each cell in the predicted masks, the IoU was calculated based on their pixel overlap and union with their ground truth mask. To classify a segmentation as a true positive, IoU thresholds ranging from 0.5 to 1.0 (in increments of 0.01) were applied. Cells meeting or exceeding the threshold were considered successful detections and included in the evaluation. Figure 12 depicts the overlap between predicted and ground truth masks at increasing IoU thresholds, highlighting the enhancement in segmentation quality as IoU values increase. For each image in the validation and test sets, the IoU scores of all qualifying cells were averaged to compute a per-image IoU. Finally, the overall model performance was determined by averaging these per-image IoU values across all images in each set, referred here as the average Jaccard Index for clarity and simplicity. Using this method, the validation set was utilized to identify optimal parameters for the segmentation models. Subsequently, the best-performing models, configured with these optimized parameters, were evaluated on the test set to provide a final assessment of their segmentation performance.

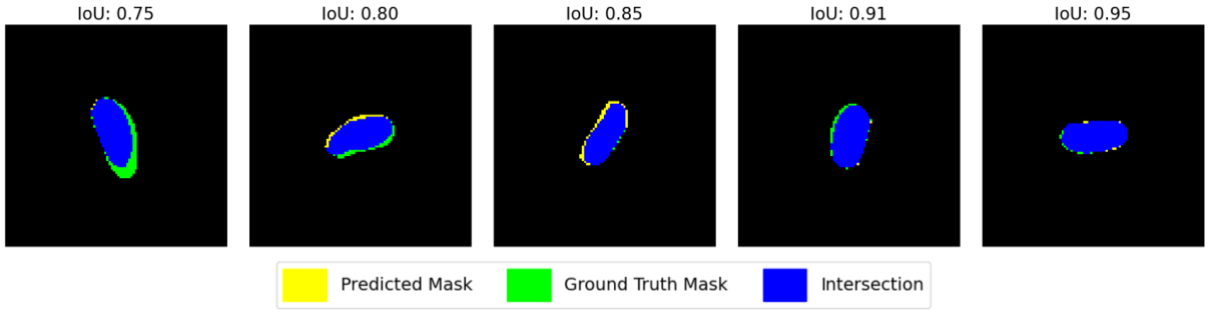


Figure 12: Comparison of Cell Segmentation at Varying IoU Values. Yellow indicates predicted mask pixels, green represents ground truth pixels, and blue highlights intersection pixels, illustrating the overlap between predicted and ground truth segmentations.

3.2 EfficientNet-Based Cell Feature Extraction Model

To extract unbiased features from bacterial images for image-based profiling, an EfficientNet-based model was trained. The dataset used consisted of multi-channel images from Govers et al. 2024 [109], featuring E. coli with 800 unique gene deletions, cultured in M9 minimal medium with L-alanine as the carbon source. The images consisted of four channels: c1 (phase-contrast), c2 (DAPI staining nucleoids), c3 (FtsZ labeled with Venus sandwich fusion fluorescent marker), and c4 (SeqA labeled with mCherry fluorescent marker).

The workflow, illustrated in Figure 13, begins with stacking the single-channel *E. coli* images to form multi-channel TIFF images. These images undergo phase-contrast segmentation using Omnipose, followed by the generation of 76x76-pixel cell patches. The dataset is then balanced

to ensure equal representation of mutant and wild-type samples, split into training (~65%), validation (~15%), and test (~20%) sets, and used to train an EfficientNet-B0 model for binary classification. Finally, the trained model extracts 1280-dimensional feature vectors from the patches, saved as CSV files for downstream analysis.

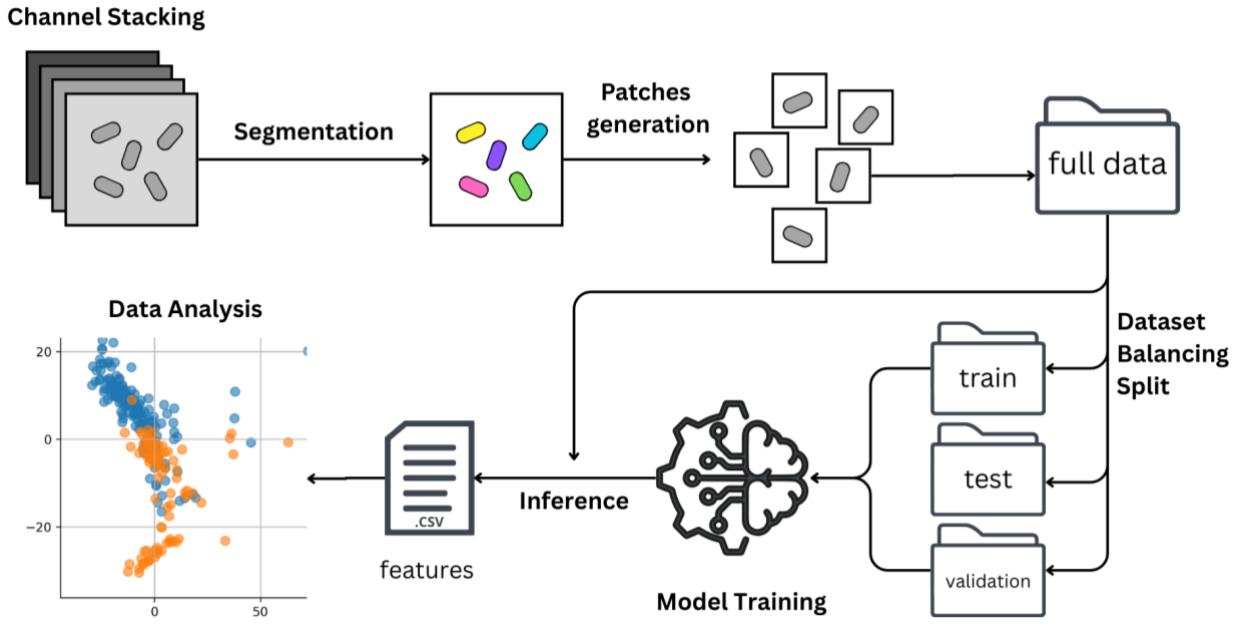


Figure 13: Feature Extraction Model Training Workflow.

3.2.1 Stacking Channels

The dataset was composed of wild type and mutant single channel images stored in four subdirectories labeled c1, c2, c3, and c4, each corresponding to a distinct imaging channel (phase-contrast, DAPI, VenusSW, mCherry), and required the creation of multi-channel TIFF images for the model training. For each image set, identified by a common base name (e.g., image), the TIFF files (e.g., c1/image_c1.tif, c2/image_c2.tif, etc.) were loaded using the Python Imaging Library (PIL). These images were converted to NumPy arrays, and their dimensions were verified to ensure consistency across channels. The arrays were then combined into a single four-channel array of shape (height, width, 4), representing the stacked channels. The resulting multi-channel image was saved as a TIFF file in a designated output directory using the tifffile library.

3.2.2 Phase Contrast Segmentation

After composing multi-channel images, phase-contrast segmentation was performed using the Omnipose software to generate masks. The same parameters as in Section 3.1.1 were applied, with the only modification being the specification of the c1 (phase-contrast) channel for segmentation due to the multi-channel nature of the images.

3.2.3 Patches Generation

Once the masks were generated, the individual cell patches were extracted from the multi-channel TIFF images. Each input image, a multi-channel TIFF file, was paired with a corresponding labeled mask image in TIFF format, where each non-zero pixel value represented a unique cell identifier, and zero denoted the background. The mask image was loaded using PIL and converted to a NumPy array, while the multi-channel image was read using the tifffile library.

For each image-mask pair, the unique non-zero labels in the mask were identified to isolate individual cells. A patch size of 76x76 pixels was defined, centered on each cell's centroid. To compute the centroid, the moments of a binary mask for each cell label were calculated using OpenCV. Moments are mathematical values that describe the distribution of pixels in a region, like a cell, capturing properties such as its area and center. Specifically, the zeroth-order moment represents the cell's area (total non-zero pixels), while first-order moments indicate the pixel distribution along the x- and y-axes. The centroid coordinates were determined as the ratio of the first-order moments to the zeroth-order moment, ensuring the cell had a non-zero area. Patches were extracted only if the full 75x75-pixel region fit within the image boundaries, skipping cells whose patches would extend beyond the image edges.

For each valid cell, a patch was extracted from the multi-channel image by cropping a 76x76-pixel region around the centroid. Background pixels (where the mask did not match the current cell's label) were set to zero to isolate the cell's signal. The resulting patches, retaining all channels, were saved as multi-channel TIFF files named using the input image's base name and the cell's label. The process was repeated for all cells in the mask, and the patches were saved in multiple subdirectories each containing 5000 patches for easier data manipulation.

3.2.4 Dataset Balancing & Dataset Split

To prepare a balanced dataset for model training, a subset of the full dataset from Govers et al. (2024) was used. While the original dataset included 800 unique gene deletion (mutant) strains, only a selected subset of 107 mutant strains was used in this project. In contrast, all available wild-type batches from the dataset were included. This selection was made to ensure manageable dataset size while maintaining representative diversity across mutant phenotypes.

The wild-type and mutant images were divided into training, validation, and test sets using a roughly 65/15/20% split. Class balance was maintained by ensuring each set contained a proportionate number of wild-type and mutant images. Specifically, the dataset consisted of 8 wild-type batches and 107 mutant groups. To avoid data leakage and promote generalization, wild-type images were grouped by experimental batch—ensuring all images from the same batch were assigned to a single set—while mutant images were grouped by deletion gene, so all images from a specific mutant strain remained together.

The training set comprised 65.04% of the total data and included 5 wild-type batches with 249,378 wild-type images (47.09% of the training set) and 74 mutant strains with 280,182 mutant images (52.91%). The validation set accounted for 14.56% of the data, containing 1

wild-type batch with 54,005 wild-type images (45.55% of validation) and 16 mutant strains with 64,553 mutant images (54.45%). Finally, the test set made up 20.40% of the dataset and consisted of 2 wild-type batches with 102,110 wild-type images (61.46% of the test set) and 17 mutant strains with 64,030 mutant images (38.54%).

A custom Python script using scikit-learn's train_test_split was employed to partition the grouped data randomly, while balancing the number of images per class based on the smaller class's count. The selected images were then organized into patches/train, patches/val, and patches/test directories, with grouping integrity preserved to support robust and unbiased model evaluation.

3.2.5 EfficientNet Model Training

An EfficientNet [12] model was then trained on the training set image patches with the task of classifying the patches as mutant or wild-type using the train.py script.

Training set patches (originally 76 × 76 pixels) were resized to 224 × 224 pixels to match the EfficientNet-B0 model's input requirements, randomly flipped horizontally with a 0.5 probability, rotated within ±15 degrees. Next, each channel was normalized using the mean and standard deviation pixel values of the entire dataset for the respective channel. Validation and test set patches were only resized and normalized with identical parameters. These steps used torchvision.transforms, and the PyTorch framework.

The EfficientNet-B0 model, pretrained on ImageNet from the efficientnet_pytorch library, had its initial convolutional layer modified to accept four input channels, using a kernel size of 3, stride of 2, padding of 1, and producing 32 output channels. The output layer was set for binary classification of mutant and wild-type cells. PyTorch's DataLoader was utilized to handle data loading for the training and validation sets, processing batches of 64 patches.

Training minimized cross-entropy loss using the AdamW optimizer with a learning rate of 3×10^{-3} and weight decay of 1×10^{-5}. A ReduceLROnPlateau scheduler adjusted the learning rate upon validation loss plateaus. The model trained for up to 200 epochs, stopping early if validation loss did not improve for 15 epochs. The best model's weights were then saved on a .pth file based on the lowest validation loss across all epochs.

In addition to this baseline configuration, multiple combinations of learning rates, optimizers, and weight decay values were explored to optimize model performance. The following configurations were tested as shown in the following table:

Table 1: Different EfficientNet Model Configurations Trained.

Model	Optimizer	Learning Rate	Weight Decay
model_lr_1e5_adamw_wd_5e2	AdamW	1×10^{-5}	5×10^{-2}
model_lr_1e4_sgd_wd_5e3	SGD	1×10^{-4}	5×10^{-3}
model_lr_5e4_radam_wd_1e4	RAdam	5×10^{-4}	1×10^{-4}
model_lr_3e4_rmsprop_wd_1e3	RMSprop	3×10^{-4}	1×10^{-3}
model_lr_5e3_sgd_wd_1e2	SGD	5×10^{-3}	1×10^{-2}

These variants allowed for assessing the sensitivity of the model's performance to different training dynamics, including convergence behavior and generalization capacity.

Test set evaluation computed accuracy, sensitivity (correctly classified mutant patches), specificity (correctly classified wild-type patches), and area under the curve (AUC) from the Receiver Operator Curve (ROC). Training and validation accuracy and loss curves were saved for analysis. The software was created in Python and used PyTorch, torchvision, argparse, tqdm, and wandb, with tifffile handling TIFF files.

3.2.6 Single Cell Images Feature Extraction

The trained EfficientNet-B0 model was subsequently used to extract features from the entire patches/cells dataset, enabling downstream analysis. The single cell images underwent the same preprocessing (resizing and normalization) as described in Section 3.2.5 ensuring consistency with the training pipeline. A Python script, feature_extraction.py, orchestrated the feature extraction process.

The model, loaded with weights from the checkpoint file (*model_lr_3e3_adamw_wd_1e5.pth*) generated during training, was set to evaluation mode to disable gradient computation. To ensure compatibility, the checkpoint was processed to remove DataParallel prefixes. The entire dataset was loaded using PyTorch's DataLoader method with a batch size of 64 patches, configured for sequential loading without shuffling. The model's extract_features method generated feature maps for each batch, which were then reduced to a 1280-dimensional feature vector per patch using adaptive average pooling. These vectors were detached from the computational graph and converted to NumPy arrays for further processing.

The extracted features were organized into a tabular format using the pandas library. Each cell was represented as a row, with 1280 columns labeled feature_0 through feature_1279 corresponding to the dimensions of the EfficientNet-B0 feature space. An additional column, image_path, was included to record the file path of each image patch for traceability. The resulting dataset was saved into a CSV file (features_model_lr_3e3_adamw_wd_1e5.csv), with the image path as the first column followed by the feature columns, facilitating subsequent analysis.

This process leveraged the same software environment as the training phase, including Python, PyTorch, torchvision, numpy, pandas, efficientnet_pytorch, and tifffile for TIFF file handling. Execution occurred in a GPU-enabled environment to optimize computational efficiency, with the output CSV stored in a designated directory for use in further classification or visualization tasks.

3.2.7 Cell Area Extraction

The cell area was extracted from the patch images, each containing a single cell with background pixels set to zero across all channels. The extraction of the cell area served two primary purposes: first, to identify and exclude poorly segmented images that could compromise the accuracy of the analysis, and second, to provide a means to verify the correctness of the inferred cell cycle trajectories. The first channel (phase-contrast) of each multi-channel TIFF patch was used to calculate the area by counting non-zero pixels, which represent the cell. The area_extraction.py script processed all TIFF files in the 'wild_type' and 'mutants' directories, loading each image with the tifffile library, verifying its four channels, extracting the first channel, and computing the area using NumPy's count_nonzero function. The results, including the computed area and image path, were stored in a pandas DataFrame and saved as a CSV file named 'patches_cell_areas.csv' for subsequent analysis.

3.3 Cell Cycle Inference

The data analysis pipeline was designed to elucidate cell cycle progression differences between wild-type and mutant *E. coli* cells by leveraging EfficientNet-derived features and cell area measurements. The pipeline, implemented in a Jupyter notebook called cell_cycle_trajectory_analysis.ipynb, encompassed data preprocessing, subset selection, normalization, dimensionality reduction, and visualization of cell cycle trajectories. The following figure (14) and subsections detail the key steps taken, which were executed using Python with libraries including pandas, NumPy, scikit-learn, PHATE, and Matplotlib.

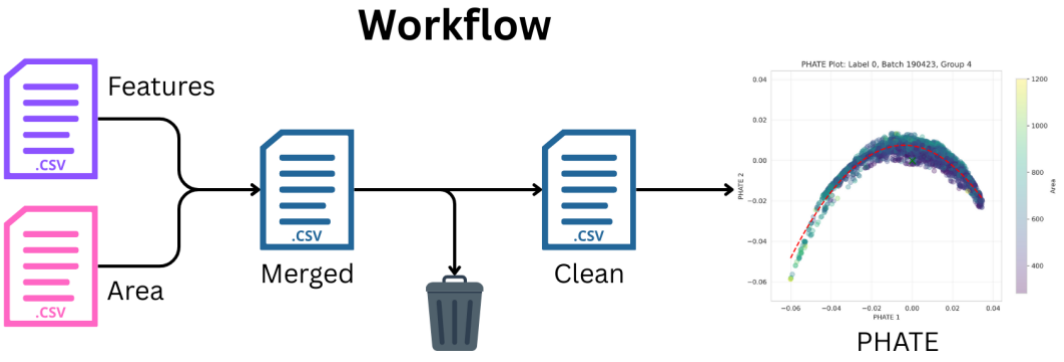


Figure 14: Cell cycle trajectory inference workflow. First each sample in feature csv file is merge with its respective area. Then, samples below a certain area threshold are discarded. Finaly, PHATE dimensionality reduction is applied to the datasubset.

3.3.1 Preprocessing of EfficientNet Extracted Features and Area

Preprocessing of the EfficientNet-extracted features and cell area data was performed to generate a cleaned dataset for subsequent analysis. The feature data were loaded sequentially from a CSV file (features_model_lr_3e3_adamw_wd_1e5.csv) in chunks of 20,000 rows to manage memory efficiently. Concurrently, a dictionary mapping normalized image paths to cell areas was constructed from area_cells.csv.

Image paths in both datasets were normalized to ensure consistency by prepending the base directory to relative paths. Each cell was labeled as wild-type (0) or mutant (1) based on the presence of "mutant" in the lowercase image path. Batch and group identifiers were extracted from the image paths using string parsing. Rows with missing values were removed to ensure data integrity. Cells with areas ≤ 270 pixels, indicative of segmentation errors or non-cellular artifacts, were filtered out, and their corresponding images were copied to a designated garbage directory for inspection. For mutant cells (label = 1), gene annotations were added by mapping group identifiers to gene names using a dictionary constructed from mutant_names.csv. The processed data were grouped by label, batch, and group, and each group was saved as a separate CSV file in an output directory, with filenames formatted as clean_label_batch_group.csv. This grouping and chunk-wise processing ensured efficient handling of large datasets while preserving data organization for downstream analysis.

3.3.2 Processing of Subset Files

Each preprocessed CSV file, previously grouped by unique combinations of label (wild-type or mutant), batch, and group during preprocessing, was loaded individually from the output directory. A filter was applied to retain only those subsets containing at least 100 cells, ensuring statistical robustness for downstream analysis, while subsets with fewer cells were excluded.

3.3.3 Outlier Removal and Feature Standardization

Within each retained subset, cells were filtered to remove outliers by excluding those with any feature value exceeding 12 standard deviations from the feature's mean across the subset. After filtering, the remaining cells' feature data were standardized using scikit-learn's StandardScaler. This process, applied independently to each subset, transformed the features to have a mean of zero and a standard deviation of one, ensuring uniformity for downstream analysis.

3.3.4 Dimensionality Reduction Using PHATE

The standardized feature data for each subset were then processed using the Potential of Heat-diffusion for Affinity-based Transition Embedding (PHATE) algorithm. PHATE reduced the high-dimensional feature space to two dimensions, producing embeddings that preserved both local and global data structures. This step enabled visualization of complex patterns, such as cell cycle trajectories, in a simplified 2D space.

3.3.5 Parabolic Fitting and Geometric Quantification

For each subset's 2D PHATE embeddings, a parabolic arc was fitted using a least-squares optimization method to capture the parabola-like shape typical of cell cycle data. The code computed several geometric metrics from these fits and the data distribution, including: curvature of the fitted parabola, arc length along the parabola, width and height of the data spread, centroid of the PHATE points.

PHATE plots were generated, where data points were colored by cell area. The fitted parabola and centroid were overlaid on these plots to visualize the trajectory and its central point.

3.3.6 Composite Visualization

Additionally, a single figure was produced displaying all fitted parabolas together—wild-type in green, mutants in blue—annotated with group identifiers and marked with centroids, providing a collective view of trajectory variations across the dataset.

3.4 Phenotypic Analysis: Differential Features, Clustering, and Dimension Reduction Visualization

The source code for the methods described in this section can be found in the Jupyter notebooks differential_feature_means_analysis.ipynb and enrichment_analysis.ipynb. Specifically, the code for the differential mean features analysis (section 3.4.1) is available in differential_feature_means_analysis.ipynb, while the code for the clustering (section 3.4.2) and dimensionality reduction visualization (section 3.4.3) is implemented in enrichment_analysis.ipynb. These notebooks contain the complete workflows, including data processing, statistical testing, clustering, and visualization, as detailed in the respective subsections.

3.4.1 Differential Mean Features Analysis

For each wild-type replicate and mutant deletion strain, the mean of every feature was calculated across all samples using pandas. The EfficientNet feature columns were selected, their means computed per each group, and results compiled into a DataFrame with metadata (label, batch, group, gene), saved as a CSV for further analysis.

To compare wild-type (label=0) and mutant (label=1) groups, t-tests and Mann-Whitney U tests were performed on each feature column using scipy.stats, with results (test statistics and p-values) saved to a CSV. Manhattan plots of -log10(p-values) for both tests were generated using matplotlib, including a p=0.05 significance threshold, and QQ plots were created to evaluate p-value distributions, all saved as high-resolution images.

3.4.2 Clustering to Identify Phenotypically Similar Mutant Strains

K-means, Gaussian Mixture Model (GMM), and HDBSCAN were applied to cluster features from the group feature means dataset (samples_features_means_updated.csv) containing

sample feature means and metadata (labels, batch, group, filename, gene), using the notebook enrichment_analysis.ipynb. Numeric features were standardized using StandardScaler. For K-means, the elbow method (1–15 clusters) identified 7 as a good number of clusters, with mutant gene assignments to each cluster and saved to kmeans_cluster_genes.csv. For GMM, BIC scores (1–12 components) selected 7 components, similarly with results saved to gmm_cluster_genes.csv. HDBSCAN used a minimum cluster size and samples of 5, with assignments saved to hdbscan_cluster_genes.csv. Gene names per cluster were extracted to identify phenotypically similar mutants for functional analysis.

3.4.3 Dimensionality Reduction Visualization

PCA, t-SNE, and UMAP reduced numeric features (excluding metadata) to two dimensions after standardization. PCA projected data onto two components, with scatter plots colored by wild-type/mutant labels or cluster assignments (K-means, GMM, HDBSCAN) and mutant points annotated with gene names. Variance explained was indicated on axes. t-SNE and UMAP (random state 42) produced similar plots, with HDBSCAN noise points labeled as "Noise." Plots were saved as PNG files with 10x8-inch size, 0.6 transparency, and 50-point size, using the tab10 colormap.

4 Results

4.1 Retraining a CNN to Segment Brightfield Images

This section details the effectiveness of retraining a Convolutional Neural Network (CNN) for segmenting brightfield images of bacteria. The primary objective was to ascertain if brightfield microscopy, a simpler and more accessible technique, could achieve accurate cell segmentation, a task traditionally challenging due to its lower contrast compared to other imaging methods like phase contrast. To optimize performance, various factors were systematically evaluated, including the impact of transfer learning, training set size, batch size, and segmentation thresholds (flow and mask).

4.1.1 Transfer Vs No-Transfer Learning and Training Set Size Performance Effect

The performance of six models was evaluated using the average Jaccard Index (JI) metric at different IoU thresholds. The models were divided into two groups: those trained with transfer learning (denoted by a "_T" suffix in the figure below) and those trained from scratch without transfer learning (denoted by an "_NT" suffix). The numbers 1000, 2000, and 2501 represent the size of the training set in images. The number of epochs required for convergence varied with training set size for both groups.

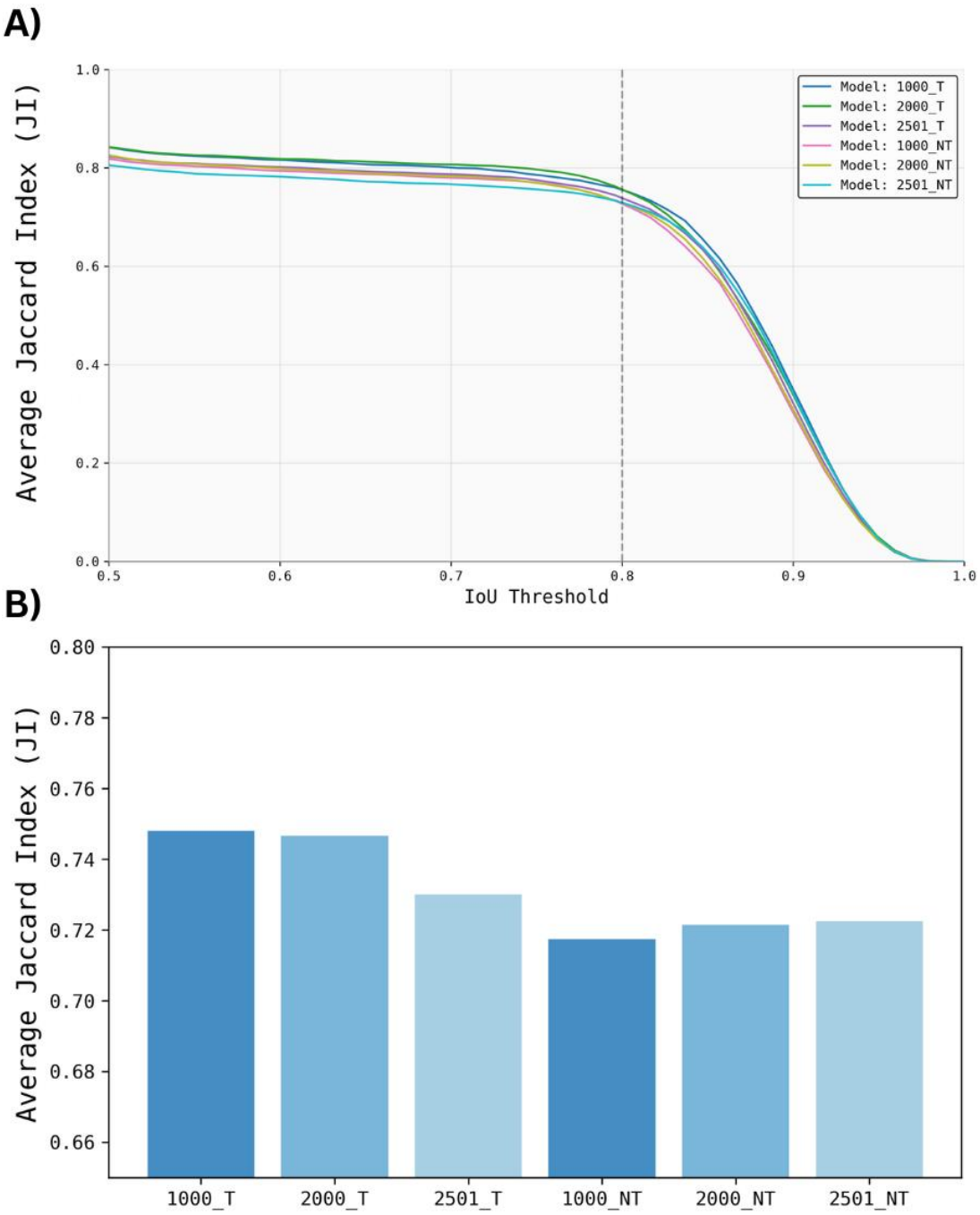


Figure 15: Transfer Vs No-Transfer JI. A) Plot of average Jaccard index versus varying IoU thresholds for the six models, with a gray line indicating the 0.8 IoU threshold. B) Histogram of average Jaccard index values at a fixed 0.8 IoU threshold across the six models.

At the 0.8 IoU threshold, transfer learning models consistently achieved higher JI scores compared to their no-transfer learning counterparts across all training set sizes. Specifically, the 1000_T model, trained on 1000 images, recorded the highest JI within the transfer learning group at 0.748. As the training set size increased for transfer learning models (2000_T and 2501_T), the JI values showed a slight, progressive decrease (0.747 and 0.730, respectively). A notable trend for transfer learning models was the significant reduction in the number of epochs required for convergence as the training set size increased, dropping from 740 epochs for 1000_T to 220 epochs for 2501_T.

For models trained without transfer learning, the average JI values at 0.8 IoU were generally lower, ranging from 0.717 (1000_NT) to 0.723 (2501_NT). Unlike the transfer learning group, the JI values for no-transfer learning models showed a slight increase with larger training sets. The number of epochs for convergence in this group initially peaked at 1500 epochs for the 2000_NT model, before decreasing to 700 epochs for the 2501_NT model.

Overall, the 1000_T model demonstrated the highest average JI across all evaluated models, underscoring the benefit of transfer learning, even with a smaller training dataset, in achieving superior segmentation performance.

4.1.3 Effect of the Batch Size on Segmentation Performance

The influence of the batch size parameter on segmentation performance was evaluated using a transfer learning model trained on 2501 images. Models were assessed at batch sizes of 50, 100, and 200. Convergence times were largely similar across these batch sizes, requiring 220, 220, and 280 epochs, respectively. The corresponding average Jaccard Index values at a fixed 0.8 IoU threshold were 0.732 (batch size 50), 0.730 (batch size 100), and 0.740 (batch size 200), as depicted in Figure 16B. While the JI values remained relatively close, a batch size of 200 yielded a marginally higher performance. This suggests that within the tested range, batch size has a limited impact on final segmentation quality, with larger batch sizes potentially offering a small slight advantage.

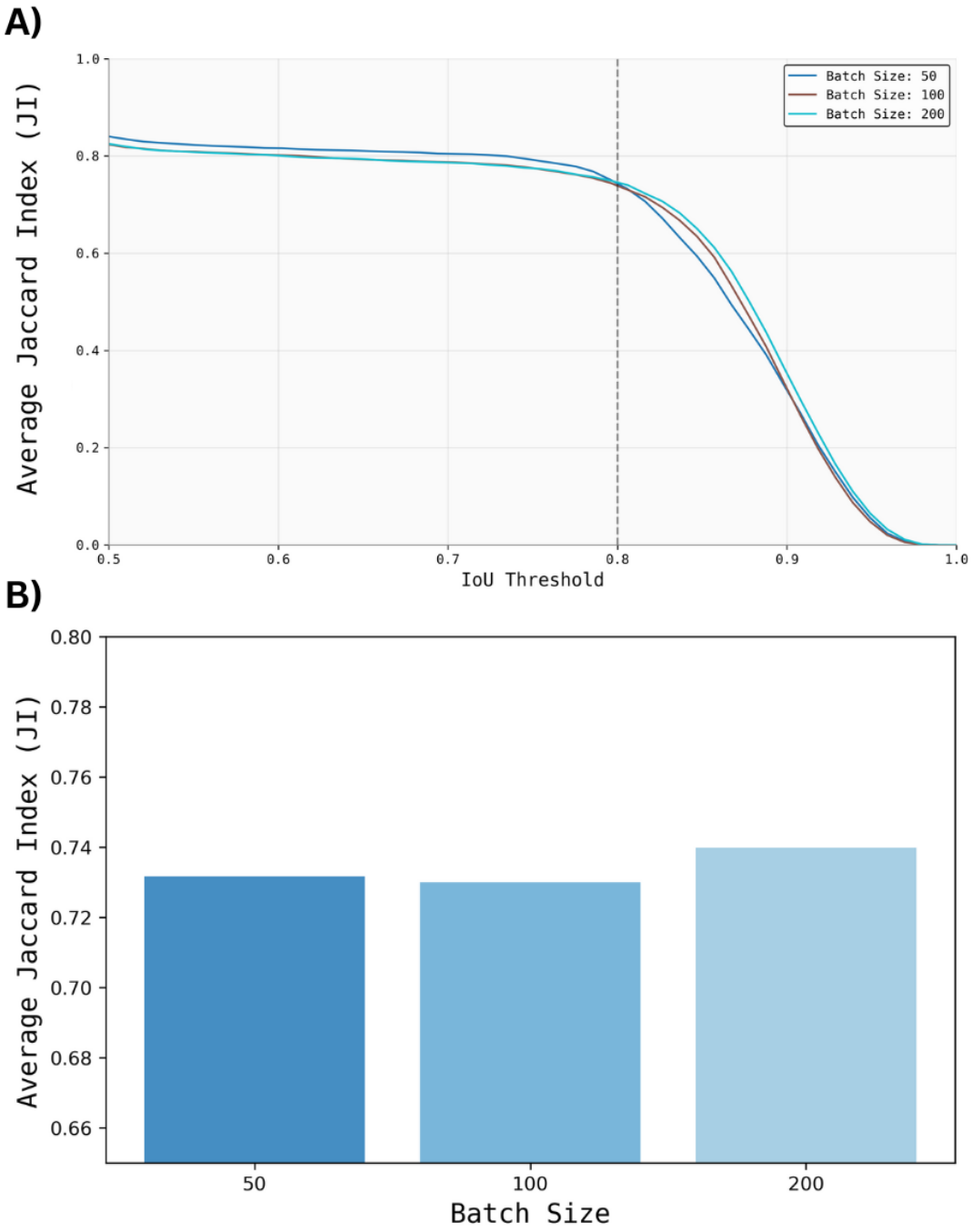


Figure 16: Batch Size Effect on JI. A) Plot of average Jaccard index versus varying IoU thresholds for different batch size models, with a gray line indicating the 0.8 IoU threshold. B) Histogram of average Jaccard index values at a fixed 0.8 IoU threshold across the models.

4.1.4 Effect of the Flow Threshold on Segmentation Performance

After a segmentation model has been trained, certain parameters can still be varied during the employment phase, and these "post-processing" parameters—namely the flow threshold (which filters masks by comparing the predicted flow field, a pixel-wise map guiding mask boundary formation, to the "true" flow derived from the masks) and the mask threshold (which determines the seeding of cell masks by thresholding the distance transform output, a map of pixel distances

from cell boundaries)—can significantly influence the final segmentation quality. The impact of the flow threshold parameter was investigated using the 2501_T transfer learning model. This parameter was varied across values of 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, and 0.6. Figure 17B illustrates the average Jaccard Index values at a fixed 0.8 IoU threshold for each flow threshold. The results showed a clear optimal point at a flow threshold of 0.1, which yielded the highest average JI of 0.755. Performance generally decreased as the flow threshold deviated from this optimum, with values ranging from 0.728 to 0.742 for other thresholds. This highlights the critical importance of optimizing the flow threshold as a hyperparameter to maximize segmentation accuracy.

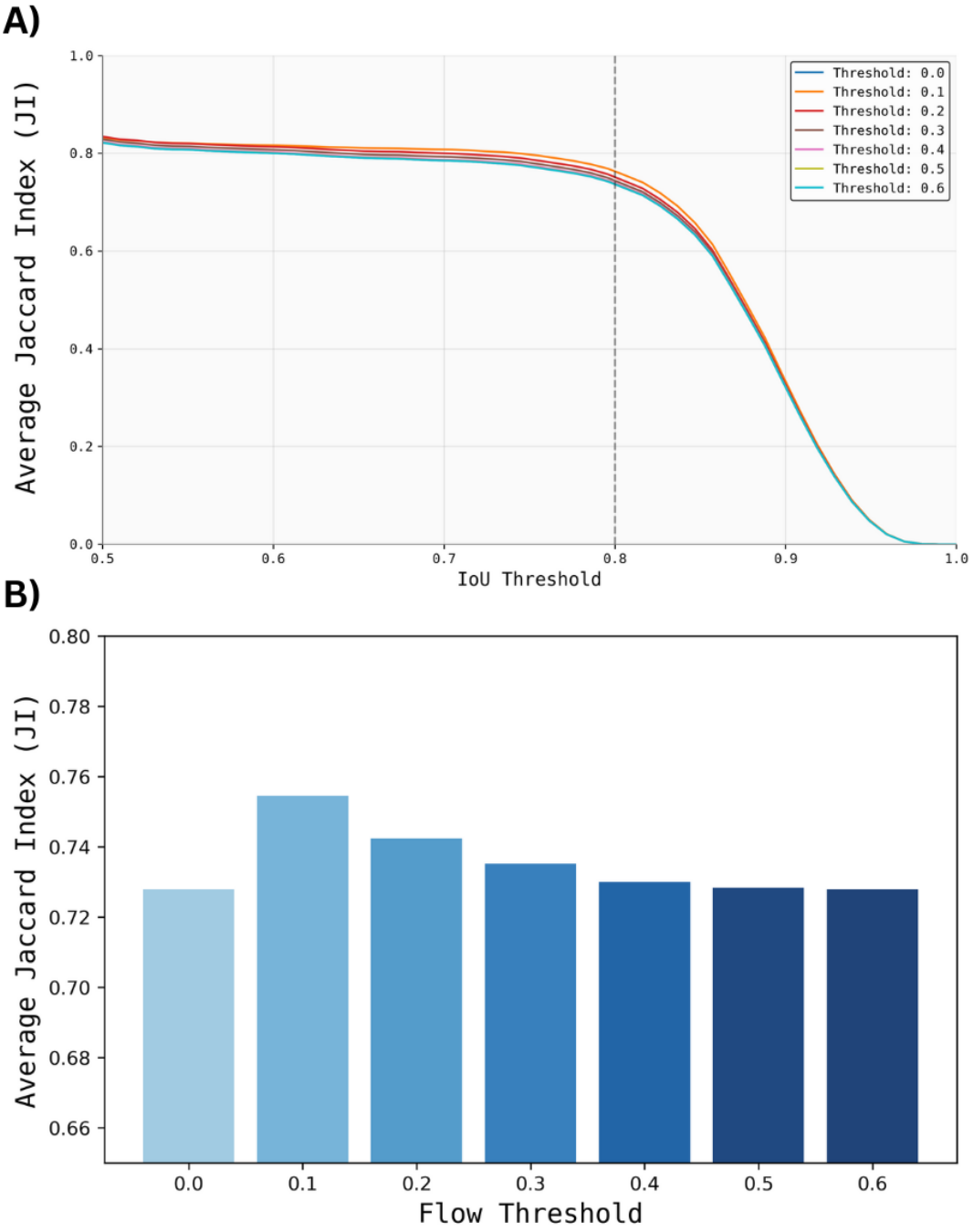


Figure 17: Flow Threshold Effect on JI. A) Plot of average Jaccard index versus varying IoU thresholds for 2501_T model at different flow thresholds, with a gray line indicating the 0.8

IoU threshold. B) Histogram of average Jaccard index values at a fixed 0.8 IoU threshold across the flow values.

4.1.5 Effect of the Mask Threshold on Segmentation Performance

This section examines the effect of varying the mask threshold parameter on segmentation results. The 2501_T transfer learning model was used, and the mask threshold was varied across values of -2, -1, 0, 1, and 2. Figure 18B presents the average Jaccard Index values at a fixed 0.8 IoU threshold for each mask threshold.

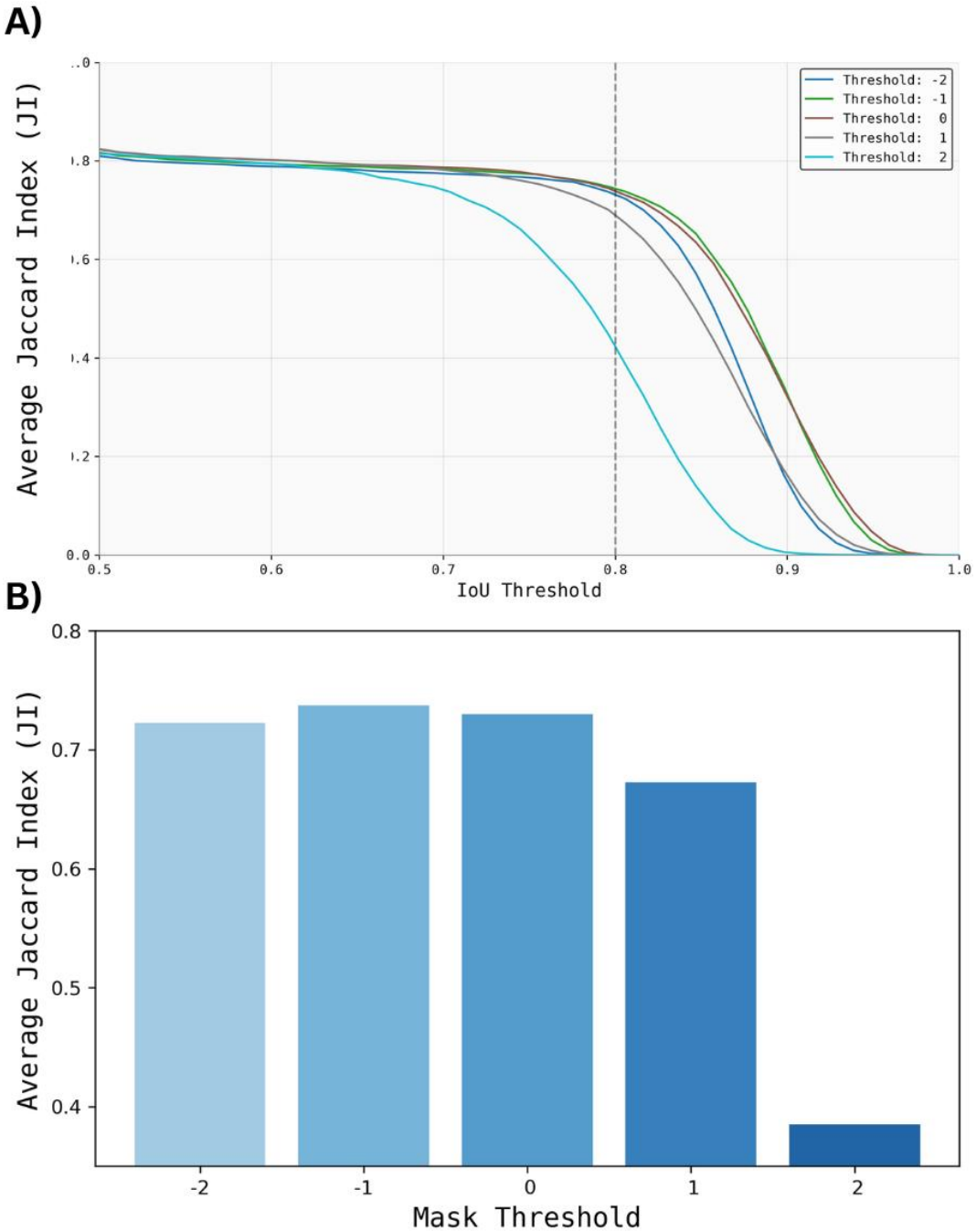


Figure 18: Mask Threshold Effect on JI. A) Plot of average Jaccard index versus varying IoU thresholds for 2501_T model at different mask thresholds, with a gray line indicating the 0.8

IoU threshold. B) Histogram of average Jaccard index values at a fixed 0.8 IoU threshold across the mask values.

The results indicate that the highest performance was observed at a mask threshold of -1, achieving an average JI of 0.737. As the mask threshold increased beyond 0, a notable decline in performance was observed, with the JI dropping significantly to 0.673 at a threshold of 1 and further plummeting to 0.385 at a threshold of 2. This strong inverse relationship at higher thresholds underscores the sensitivity of segmentation quality to this parameter and the necessity of careful selection to avoid substantial degradation in results.

4.1.6 Test Set Performance Evaluation of Transfer Learning Models

In machine learning, it is crucial to evaluate a model's ability to generalize to unseen data. This is typically achieved by splitting the dataset into training, validation, and test sets. The training set is used to train the model, allowing it to learn patterns and relationships. The validation set is used during the training process to tune hyperparameters and prevent overfitting, providing an unbiased estimate of model performance on new data during development. Finally, the test set is a completely independent dataset used only once, after the model is fully developed and optimized, to provide a final, unbiased evaluation of its generalization capability.

The performance of the best transfer learning models, previously trained on datasets of 1000, 2000, and 2501 images, was rigorously evaluated on an independent test set to determine their final segmentation effectiveness. At a fixed 0.8 IoU threshold, the average Jaccard Index values for these models were 0.858 (1000 images), 0.825 (2000 images), and 0.841 (2501 images), respectively. As shown in Figure 19, the model trained on 1000 images achieved the highest Jaccard Index on the test set, indicating its superior generalization performance among the final models.

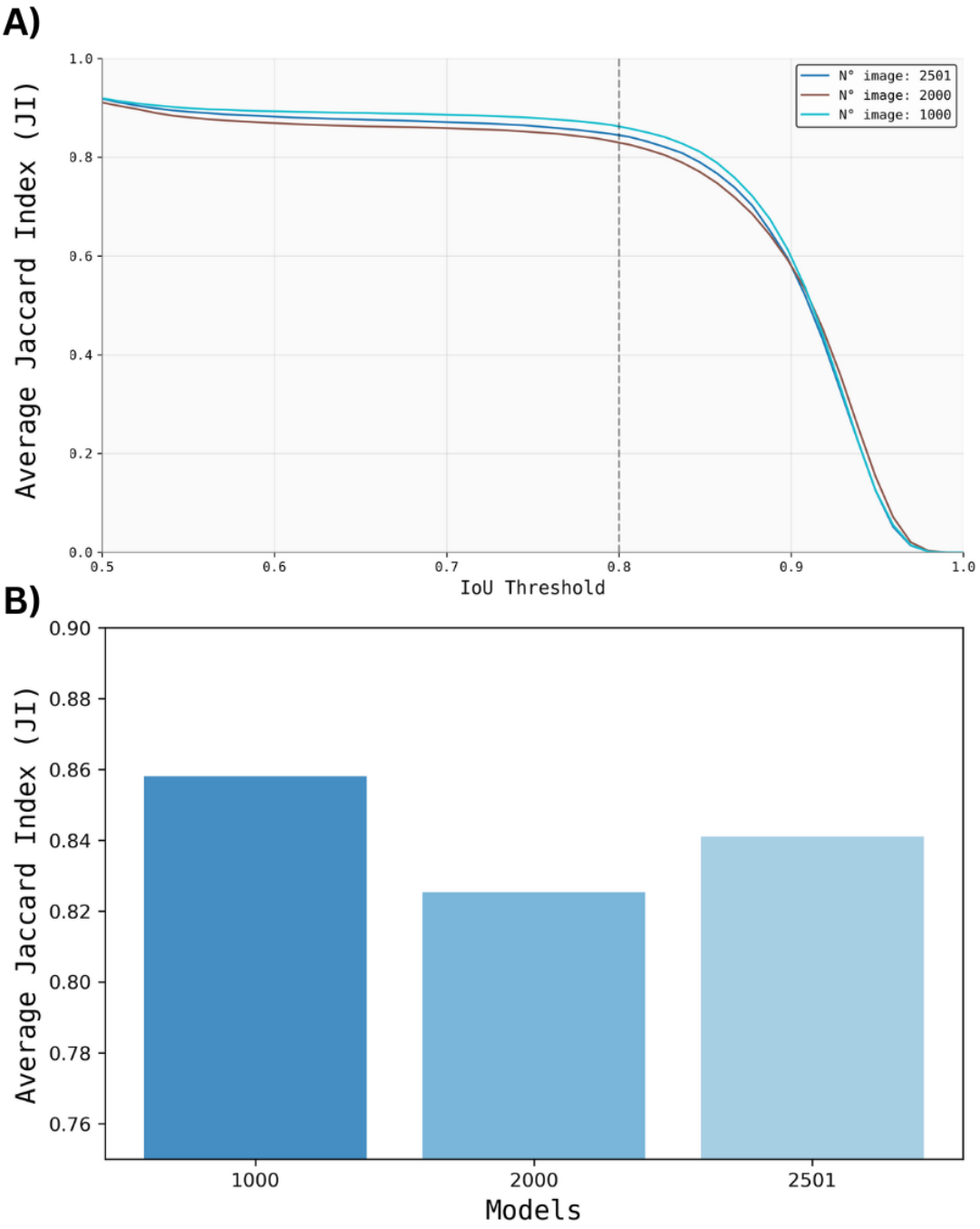


Figure 19: Test Set Model JI Evaluation. A) Plot of average Jaccard index versus varying IoU thresholds for the best transfer learning models, with a gray line indicating the 0.8 IoU threshold. B) Histogram of average Jaccard index values at a fixed 0.8 IoU threshold across the models.

Additionally, a visual inspection was conducted to qualitatively assess the segmentation quality. Figure 20 displays a predicted mask from the 1000_T model's test set alongside its corresponding phase contrast generated ground truth mask and the original brightfield image. Through visual analysis, no significant issues were identified in the segmentation output,

affirming the model's robustness in accurately delineating bacterial cells in brightfield images.

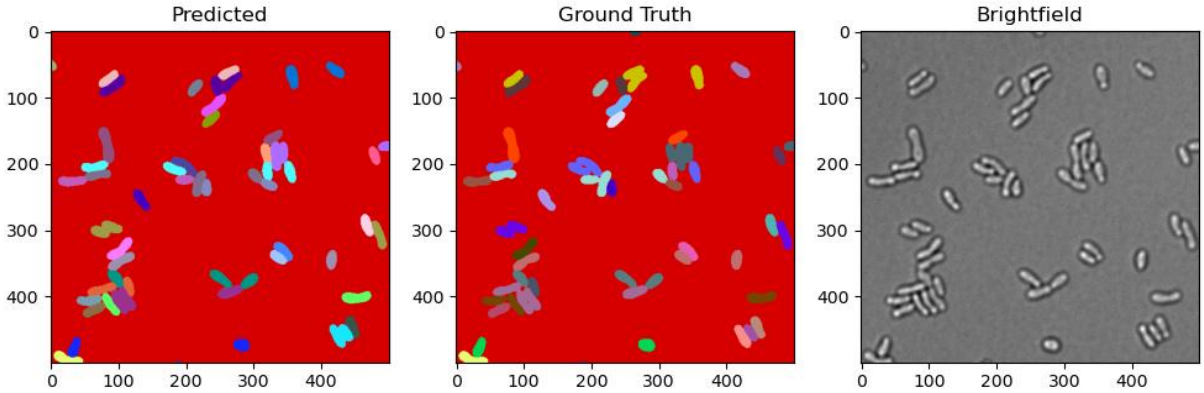


Figure 20: Masks Visualization. Left: Predicted 1000 images transfer learning model mask. Middle: phase contrast generated ground truth mask. Right: Brightfield image.

4.2 Feature Extraction Model Results

To extract unbiased, high-dimensional features from multi-channel bacterial images, an Efficient Models were trained to classify mutant versus wild-type *E. coli* cells, and their performance was assessed using metrics such as accuracy, sensitivity, specificity, and AUC, to determine their ability to differentiate between the two classes.

4.2.1 EfficientNet Model Evaluation

Six EfficientNet models were trained to classify mutant versus wild-type bacterial cells, using different optimizers (AdamW, SGD, RAdam, RMSProp), learning rates (1×10^{-5}) to (5×10^{-3}), and weight decay parameters (1×10^{-5}) to (5×10^{-2}). Model performance was evaluated on a validation (Appendix A1) and test set using accuracy, sensitivity, specificity, and area under the curve (AUC) from the Receiver Operating Characteristic (ROC) curve (Table 1). These metrics were chosen to assess the models' ability to correctly distinguish between classes, with sensitivity and specificity highlighting performance on mutant and wild-type cells, respectively, and AUC indicating overall discriminative power.

The model with a learning rate of (3×10^{-3}), AdamW optimizer, and weight decay of (1×10^{-5}) (*model_lr_3e3_adamw_wd_1e5*) achieved the best performance. It recorded the lowest validation loss of 0.431 and a validation accuracy of 82.46% at epoch 14, with training halted by early stopping at epoch 29 after no improvement in validation loss for fifteen epochs. Figure 21 illustrates this model's training dynamics over its 30 epochs. The left panel shows training and validation loss curves, demonstrating stable convergence. The right panel displays corresponding accuracy curves, indicating consistent improvement in classification performance.

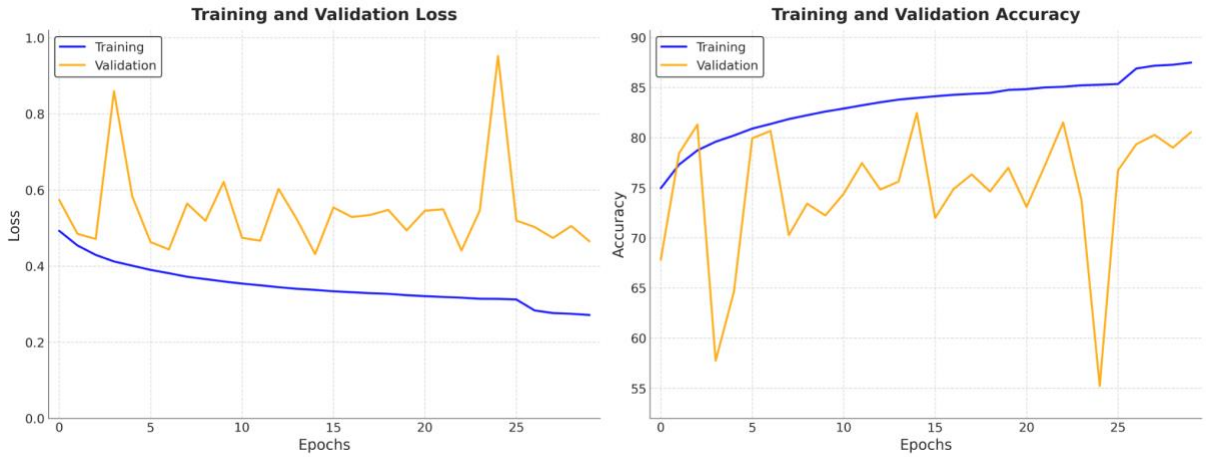


Figure 21: Training and validation performance for *model_lr_3e3_adamw_wd_1e5*. Left: Loss curves for training and validation sets across epochs. Right: Accuracy curves for training and validation sets across epochs.

Test set performance for all models is summarized in Table 1. Accuracies ranged from 83.4% to 90.8%, with AUC values between 0.903 and 0.970. The top-performing model (*model_lr_3e3_adamw_wd_1e5*) achieved 90.8% accuracy, 94.1% sensitivity, 85.7% specificity, and an AUC of 0.970, indicating excellent classification performance. A comparable model with a lower learning rate (1×10^{-5}) and higher weight decay (5×10^{-2}) reached 89.4% accuracy and an AUC of 0.958. Models using SGD, RAdam, or RMSProp optimizers performed less effectively, with the SGD-based model (learning rate (5×10^{-3}), weight decay (1×10^{-2}) yielding the lowest performance: 83.4% accuracy, 86.6% sensitivity, 78.3% specificity, and an AUC of 0.903.

Table 2: Test set evaluation parameters for all trained models.

Model	Accuracy	Sensitivity	Specificity	AUC
model_lr_3e3_adamw_wd_1e5	90.828%	94.055%	85.680%	0.970
model_lr_1e5_adamw_wd_5e2	89.405%	90.917%	86.994%	0.958
model_lr_1e4_sgd_wd_5e3	86.786%	92.723%	77.320%	0.945
model_lr_5e4_radam_wd_1e4	86.324%	90.184%	80.169%	0.938
model_lr_3e4_rmsprop_wd_1e3	85.661%	93.865%	72.578%	0.922
model_lr_5e3_sgd_wd_1e2	83.404%	86.626%	78.265%	0.903

Overall, all models achieved high AUC scores, reflecting robust discriminative ability. However, AdamW-based configurations consistently outperformed others, likely due to better handling of weight updates in the presence of sparse gradients. These results highlight the importance of optimizer and hyperparameter tuning in optimizing deep learning models for bacterial cell classification.

4.3 Differential Mean Features Analysis and Phenotypic Divergence

This part focused on analyzing the extracted features to identify significant differences between wild-type and mutant strains and to visualize their phenotypic relationships. Statistical tests (t-test and Mann-Whitney U test) were used to confirm significant differences in feature means, and dimensionality reduction techniques like PCA, t-SNE, and UMAP were employed to visualize the data and identify distinct groupings among samples.

4.4.1 Differential Mean Features Analysis

The average values of several features differed significantly between wild-type replicates and mutant deletion strains. The t-test analysis identified 879 EfficientNet features out of the 1280 with p-values below the 0.05 significance threshold, with 9 features achieving p<1e-40, indicating extreme differences between the two groups (Figure 22A). The Mann-Whitney U test confirmed these findings, identifying 1110 significant features (p<0.05), with 873 overlapping with the t-test results at p<0.05, suggesting consistency across statistical approaches (Figure 22B). The Manhattan plots of -log10(p-values) for both tests displayed a very high amount of prominent peaks highlighting key differences between the means of features in the two populations (Figure 22A, B).

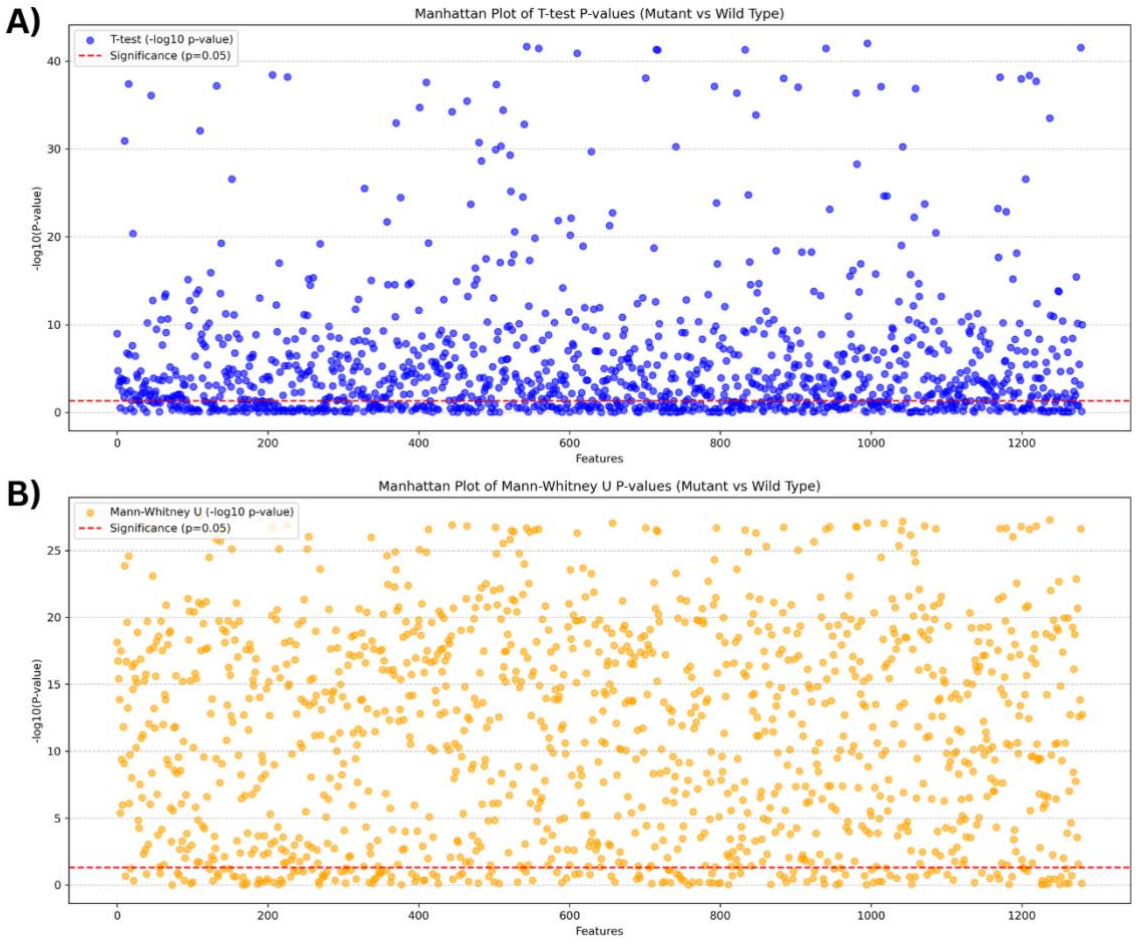


Figure 22: Manhattan plot illustrating significant differences in features between wild-type and mutant groups. Displaying the negative log10 p-values in the y-axis, and the 1280 EfficientNet features in the x-axis. Features above the red dotted line are statistically significant. A) t-test results, B) Mann-Whitney U test results.

Quantile-quantile (QQ) plots showed that p-values for the t-test generally aligned with the expected uniform distribution, with deviations in the upper tail (Figure 23A). However, the Mann-Whitney U test displayed the shape of an arc with most values positioned above their expected distribution (Figure 23B). Indicating widespread significant differences in most features between the two groups. Both tests reflect a large subset of features with strong evidence of differential means (Figure 23).

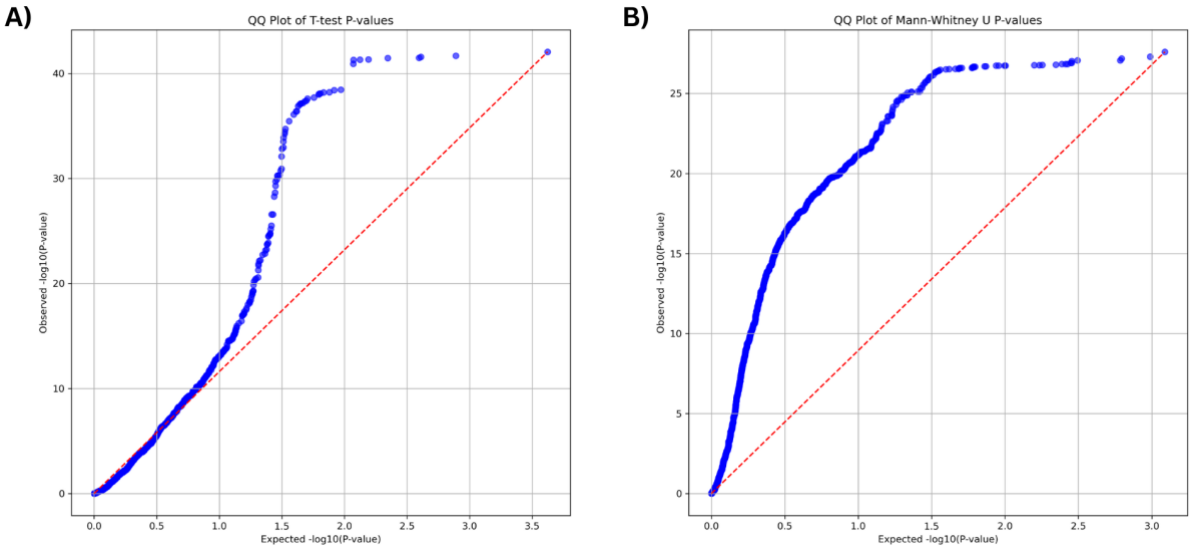


Figure 23. QQ plots comparing observed and expected distributions under the null hypothesis. A) QQ plot for the t-test. B) QQ plot for the Mann–Whitney U test. Deviations from the diagonal line indicate departures from the expected distribution under the null hypothesis.

4.4.2 Phenotypic Divergence Among Samples

Dimensionality reduction and clustering techniques helped visualize phenotypic relationships among wild-type and mutant strains in two dimensions (Appendix A2). Principal Component Analysis (PCA) captured 86.63% of the total variance in the first two components (PC1: 73.72%, PC2: 12.91%). PCA scatter plots showed some overlap between wild-type and mutant strains, with both wild-type and mutant samples clustering tightly in the lower-left quadrant, but with a distinct separation into two clusters at line -20 from PC2 (Figure 24).

Looking at the PCA scatter plot there are notable outliers. For the wild types, points like the one around 150 on PC1, 80 on PC2 and a point near 250 on PC1 and 20 on PC2 deviate significantly. For the mutants, samples such as *"ΔygeG"* and *"ΔycgV"*, *"ΔdamX"*, *"ΔsucB"*, and *"Δrep"* stand out as outliers, indicating potential phenotypic divergence from the main groups. These outliers suggest that certain mutants and wild-type replicates exhibit unique feature profiles despite the overall clustering.

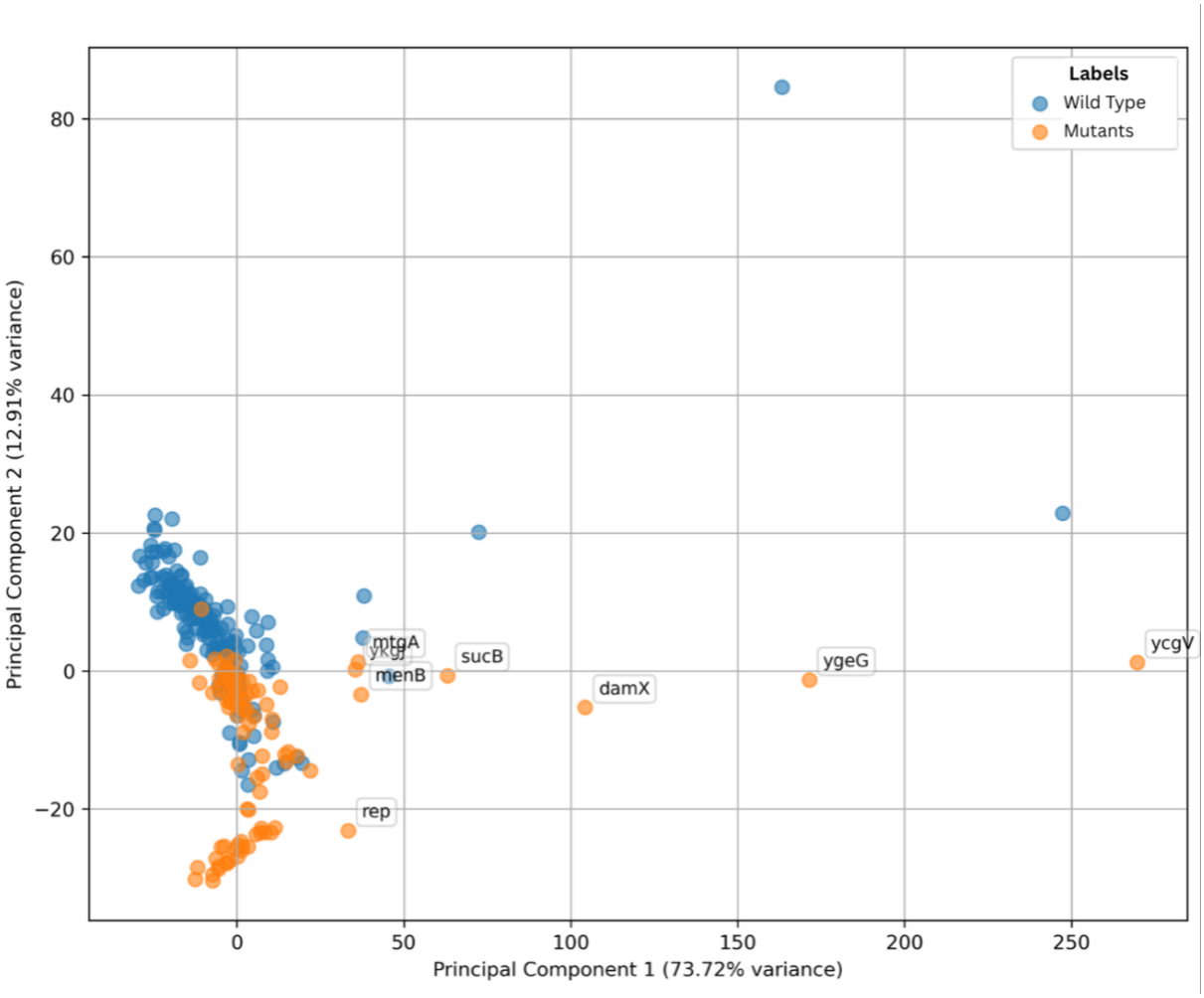


Figure 24. PCA plot showing phenotypic divergence of mutant deletion strains from the wild type. Wild type in blue, and mutant deletion strain in orange. Mutant samples are annotated with their deletion gene. PC1 captured 73.72% of the explained variance, and PC2 12.91%.

4.4 Inferring Cell Cycle Progression from Deep Feature Embeddings

To infer cell cycle progression in wild-type and mutant *E. coli* cells PHATE dimensionality reduction on the EfficientNet-derived features and cell area measurements was performed. The analysis aimed to capture continuous trajectories indicative of cell cycle progression and highlight differences between wild-type and mutant strains in terms of trajectory shape, cell area distribution, and other geometric variables.

4.4.1 Comparative PHATE Trajectory Mapping of Wild-Type and Mutant Strains

The following analysis presents the outcomes of applying PHATE dimensionality reduction to high-dimensional EfficientNet-extracted feature data from wild-type and mutant deletion strains, with the goal of visualizing continuous, arc-shaped trajectories reflective of cell cycle progression. The analysis resulted in a dataset comprising 162 PHATE plots for wild-type

replicates (Appendix A.3) and 107 plots for mutant deletion strains (Appendix A.4), totaling 269 plots that collectively illustrate continuous, arc-shaped trajectories indicative of cell cycle progression as shown in Figures 27 and 28.

The plots display variations in shape and cell area distribution across replicates and deletion strains, with most exhibiting a characteristic arc-shaped trajectory, though some deviate from this pattern and do not form a clear arc (Appendix A.3 and A.4). Additionally, differences in arc length are observed, and in several plots, a distinct distribution is noted where smaller-area cells (purplish hues) are concentrated on the left leg of the arc and larger-area cells (greenish to yellowish hues) on the right leg, a trend evident in Figure 25 but not consistently observed across all visualizations. A composite overlay of fitted parabolic arcs from all PHATE plots, presented in Figure 27, highlights differences between wild-type (green arcs) and mutant (blue arcs) strains, while a pairwise analysis of extracted metrics—curvature, arc length, width, and height—revealed in Figure 28 shows distinct distributions, with wild-type replicates demonstrating greater variability compared to the more uniform profiles of mutant strains.

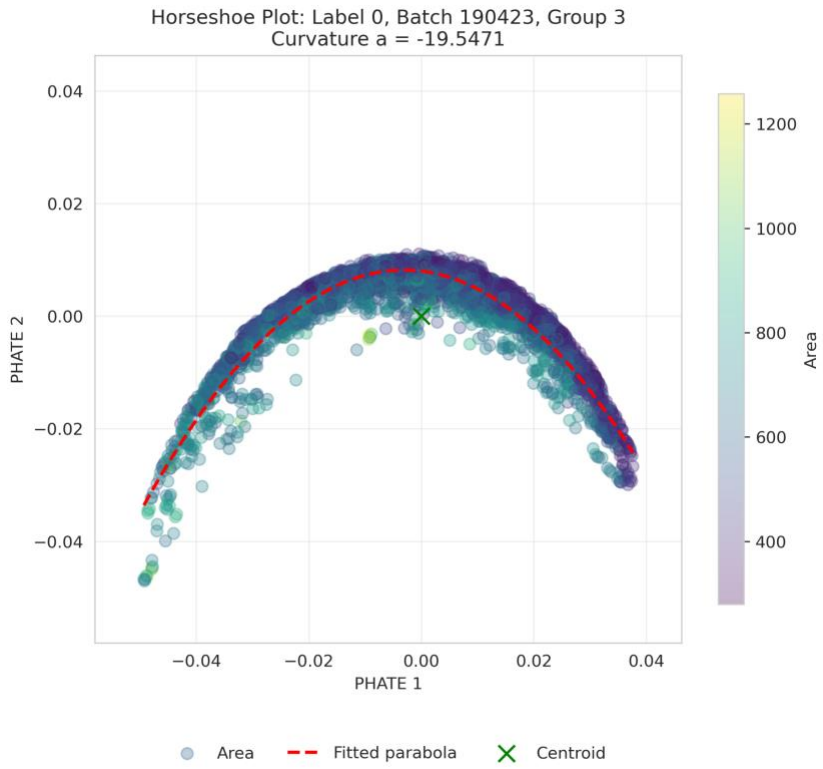


Figure 25: Representative PHATE Plot of a Wild-Type Replicate. This two-dimensional PHATE plot (Batch 190425, Group 111) shows data points colored by cell area, ranging from ~400 units (purple) to ~1200 units (yellow). The plot exhibits an arc-shaped trajectory, with smaller-area cells mostly on the left leg and larger-area cells on the right leg. A fitted parabola (red dashed line) traces the trajectory, and the centroid (green "X") marks the data's geometric center.

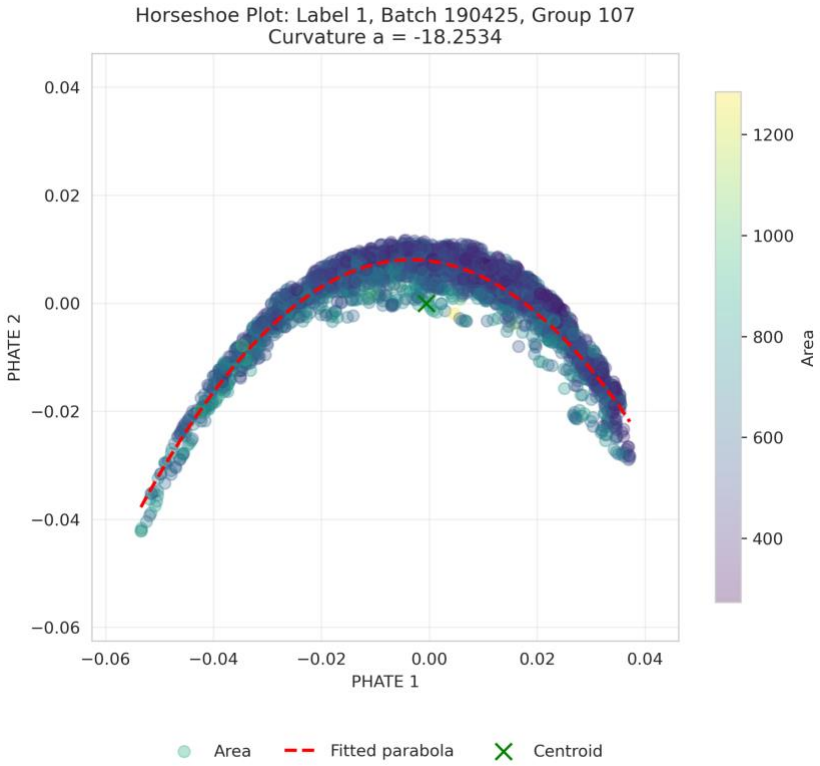


Figure 26: Representative PHATE Plot of a Mutant Deletion Strain. This PHATE plot for a mutant deletion strain shows data points colored by cell area. An arc-shaped trajectory is present, differing in shape and cell area distribution from the wild-type. A fitted parabola (red dashed line) and centroid (green "X") highlight the trajectory's structure and central position.

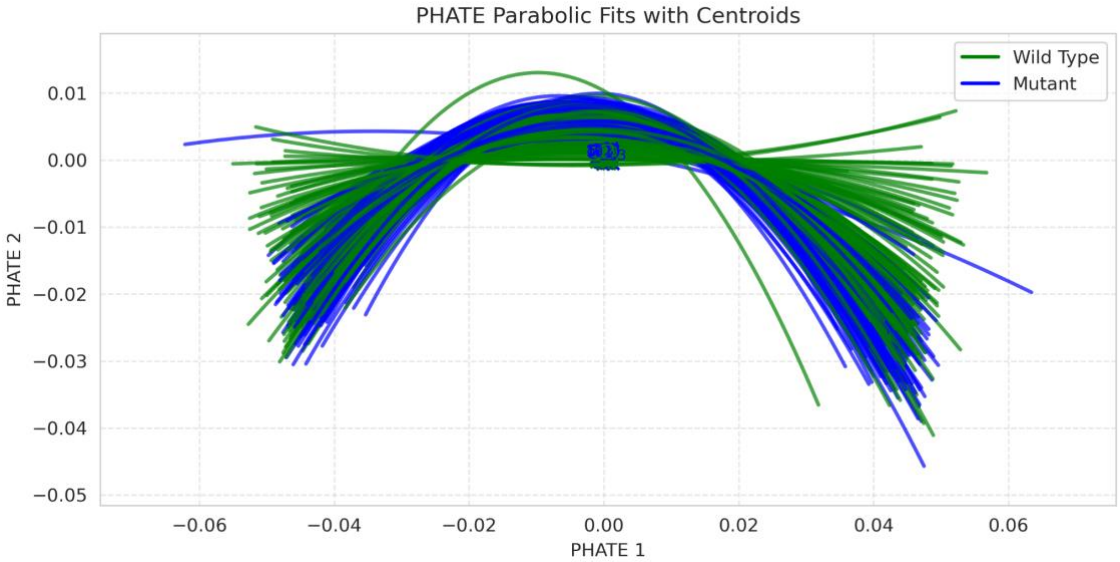


Figure 27: Overlay trajectory plot showing distinct parabolas for wild-type and mutant deletion strains. This composite plot includes fitted parabolic arcs from all 269 PHATE plots, with green arcs for wild-type replicates and blue arcs for mutant strains.

4.3.2 Extracted Trajectory Metrics Analysis

The quantitative analysis of extracted metrics, shown in Figure 28, highlights distinct differences between wild-type and mutant samples. Wild-type replicates exhibit a broader curvature range with a higher mean curvature, a similar arc length span to mutants but with greater variability, and a larger mean width with a smaller mean height. In contrast, mutant strains show a more constrained profile, with more negative curvature values, a narrower arc length distribution, a smaller mean width, and a greater mean height with less variability. These findings indicate significant variability in wild-type trajectory metrics and a more uniform progression in the mutant dataset. Overall, there is a clear difference in the means and ranges of these metrics between the two groups.

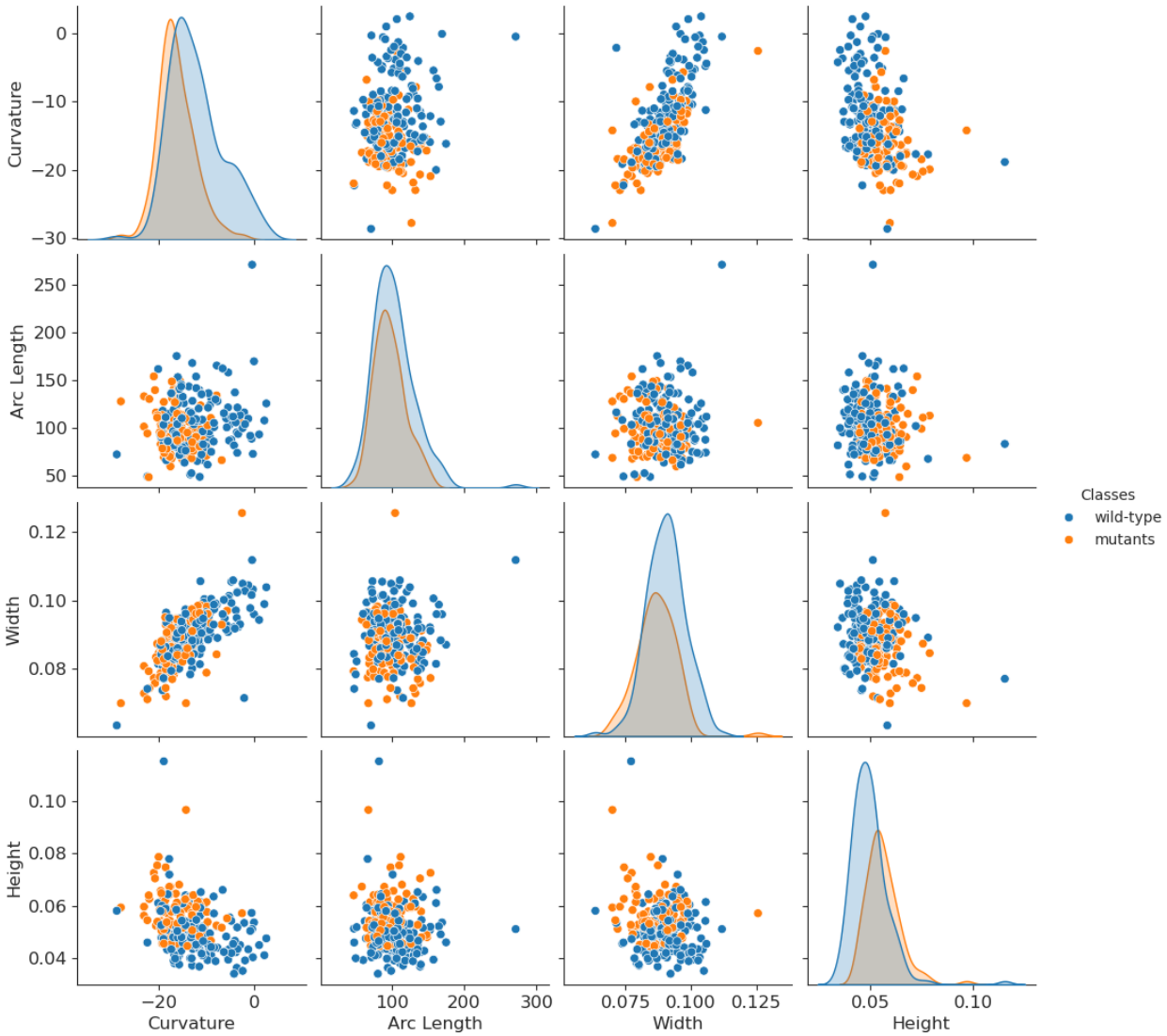


Figure 28: Pairwise Analysis of Extracted Trajectory Metrics. This 4x4 pair plot displays four metrics—curvature, arc length, width, and height—from the PHATE plots. Histograms on the diagonal show each metric's density, and scatter plots show pairwise comparisons. Blue points (wild-type) have a wider spread, while orange points (mutant) are more clustered, indicating greater uniformity in mutant trajectories.

5 Discussion

5.1 Brightfield Segmentation Model

This part evaluated the performance of Omnipose convolutional neural network (CNN) models for segmenting brightfield images, focusing on the effects of transfer learning, training set size, batch size, and flow and mask thresholds. The results provide insights into optimizing brightfield segmentation and highlight the practical utility of transfer learning for bacterial image-based profiling.

5.1.1 Superiority of Transfer Learning

Transfer learning models consistently outperformed their no-transfer-learning counterparts across all training set sizes, as evidenced by higher average Jaccard Index (JI) values at a 0.8 IoU threshold (e.g., 0.748 for 1000_T vs. 0.717 for 1000_NT). This superiority likely stems from the use of pre-trained weights, which initialize the model with robust feature extractors learned from large, diverse datasets. These features, such as edges and textures, are particularly relevant for segmenting complex brightfield images, enabling faster convergence and improved generalization. Additionally, transfer learning models required fewer epochs to converge (e.g., 220 epochs for 2501_T vs. 700 for 2501_NT), reducing computational costs and making them more practical for resource-constrained settings.

5.1.2 Effect of Training Set Size

The impact of training set size varied between transfer and no-transfer-learning models. For transfer learning models, the 1000-image model (1000_T) unexpectedly achieved the highest JI in the final evaluation (0.858), surpassing models trained on 2000 (0.825) and 2501 images (0.841) (Figure 15). This suggests that 1000 images may have been sufficient to capture the variability in brightfield images, while larger datasets could have introduced redundant or noisy data, potentially diluting model performance. Alternatively, the 1000_T model may have benefited from longer training (740 epochs vs. 220 for 2501_T), allowing finer optimization. Overall, the transfer learning models achieved relatively good performances, with segmentation masks very similar to their ground truth counterparts. For no-transfer-learning models, JI slightly increased with training set size (0.717 for 1000_NT to 0.722 for 2501_NT), but required more epochs, peaking at 1500 epochs for 2000_NT. This indicates that models trained from scratch rely more heavily on larger datasets to learn relevant features, yet still underperform compared to transfer learning models. This observation is consistent with prior research, such as Luo et al. (2018), which showed that larger training datasets generally enhance the performance of CNN-based models trained from scratch [150], underscoring their reliance on sufficient data to learn effective features, while transfer learning models can achieve strong results with smaller datasets [151].

5.1.3 Negligible Impact of Batch Size

Batch size had a minimal effect on the performance of the 2501-image transfer learning model, with JI values ranging from 0.730 (batch size 100) to 0.739 (batch size 200) at a 0.8 IoU threshold. Convergence times were also similar (220–280 epochs). This stability may reflect the robustness of the transfer learning model, where pre-trained weights reduce sensitivity to batch size variations. Additionally, the dataset size (2501 images) likely provided sufficient gradient information across batch sizes, minimizing differences in optimization. These findings suggest that batch size tuning may not be critical for similar tasks, allowing flexibility in computational resource allocation. This is likely due to the combined strength of a robust transfer learning approach and a sufficiently sized dataset that appears to have mitigated the typical sensitivity to batch size variations often seen in training models from scratch or with smaller datasets.

5.1.4 Optimization of Segmentation Flow and Mask Thresholds

The flow and mask threshold parameters significantly influenced segmentation accuracy. The highest JI (0.754) was observed at a flow threshold of 0.1, indicating an optimal balance between under- and over-segmentation. Lower thresholds may have produced overly coarse segmentations, while higher thresholds (e.g., 0.6) could have fragmented objects, reducing JI. Similarly, a mask threshold of -1 yielded the best JI (0.737), closely aligning predicted masks with phase contrast ground truth masks. Higher mask thresholds, particularly 2 (JI of 0.385), likely imposed overly stringent criteria, discarding valid segmentations. These results underscore the importance of fine-tuning flow and mask thresholds to match the characteristics of brightfield images and ground truth masks.

5.1.5 Final Model Performance

The final evaluation of transfer learning models demonstrated high segmentation accuracy, with JI values of 0.858, 0.825, and 0.841 for the transfer 1000, 2000, and 2501 image models, respectively. The superior performance of the 1000_T model suggests that transfer learning can achieve excellent results even with moderate dataset sizes, making it a cost-effective approach for brightfield segmentation. Visual inspection of predicted masks further confirmed quantitative results, with the 1000_T model's masks closely resembling phase contrast ground truth masks, validating its practical utility.

5.1.6 Implications for Bacterial Image-Based Profiling

These findings have significant implications for bacterial image-based profiling. Traditionally, phase contrast imaging is the preferred method for bacterial segmentation due to its high contrast between cells and the background [152, 153]. However, this study demonstrates that brightfield images, despite their lower contrast at cell boundaries, can achieve comparable segmentation accuracy. This demonstrates brightfield imaging could be a viable alternative for segmentation in bacterial image-based profiling, offering researchers greater flexibility in their imaging choices. Additionally, the ability to use small datasets, combined with transfer

learning, enables the training of effective brightfield segmentation models in fewer epochs, reducing the need for extensive annotated data and computational resources.

The high segmentation accuracy achieved by transfer learning models facilitates precise quantification of bacterial morphology, growth patterns, and spatial distributions, which are essential for applications such as antibiotic resistance studies and microbial ecology. The efficiency of transfer learning, requiring fewer images and training epochs, supports its integration into high-throughput imaging pipelines where resources and annotated data may be constrained. Furthermore, the robustness of the 1000_T model indicates that transfer learning can generalize across diverse bacterial species datasets, enhancing its practical utility in real-world microbiological research.

5.1.7 Limitations of Using the Omnipose Algorithm

Despite the promising results, this study has several limitations. The Omnipose framework lacks an early stopping mechanism based on a validation set, which may have led to overtraining, particularly for models with high epoch counts (e.g., 1500 for 2000_NT). Additionally, saving the model every 20 epochs may have missed the point of optimal model convergence, potentially contributing to the plateaued performance observed in no-transfer-learning models. The quality of ground truth masks, derived from phase contrast images, also posed challenges. Artifacts such as bubbles in brightfield images, combined with poor segmentations and hallucinations in phase contrast masks, most likely reduced segmentation accuracy, as the model's predictions are limited by the quality of the ground truth. Consequently, brightfield segmentation masks are only as accurate as their phase contrast ground truth counterparts. Furthermore, the model's performance is constrained by the diversity of the dataset; variations in image focus or other imaging parameters could degrade performance.

Future work should address these limitations by implementing early stopping mechanisms to optimize training efficiency and mitigate overfitting. Enhancing ground truth quality, potentially through advanced preprocessing to remove artifacts or by incorporating multi-modal imaging (e.g., fluorescence), could improve segmentation accuracy. Expanding the dataset to include more diverse images with varying imaging conditions will help ensure robustness and generalizability. Additionally, exploring a wider range of hyperparameters, such as learning rates or alternative optimizers, may further enhance model performance. Testing these models on diverse bacterial datasets or other brightfield imaging contexts will be essential to evaluate their applicability. Finally, integrating these models into automated profiling pipelines could facilitate scalable, high-throughput bacterial analysis, advancing their practical utility in microbiology research.

5.2 Evaluation Metrics for the EfficientNet Feature Extraction Model

The best feature extraction model, *model_lr_3e3_adamw_wd_1e5*, exhibits strong performance in differentiating wild-type and mutant *E. coli* cells, as evidenced by its comprehensive

performance metrics (Table 1). Achieving an overall accuracy of 90.828%, the model demonstrates a high level of correctness in classifying samples, reflecting its ability to effectively capture and utilize discriminative phenotypic features present in the images. Furthermore, the model obtained a sensitivity of 94.055%, indicating exceptional proficiency in correctly identifying mutant cells. Complementing this, the specificity of 85.680% highlights the model's good capability of also accurately identifying wild-type cells, though it suggests a slightly higher rate of false positives compared to sensitivity, pointing to a potential area for further optimization. The Area Under the Curve (AUC) of 0.970 further reinforces the model's outstanding discriminatory power, as it effectively distinguishes between wild-type and mutant deletion strains across various classification thresholds. This near-perfect AUC value confirms the model's robustness and its ability to maintain performance consistency, making it a reliable tool for bacteria image-based profiling.

5.3 Discriminative Features and Phenotypic Variability

5.3.1 Statistical Validation of Discriminative Features

The differential analysis between the averaged mutant and wild-type samples revealed that the majority of features extracted by our EfficientNet model were statistically significant. As the model was trained for a classification task to distinguish wild-type versus mutant cell populations, the extracted features were expected to be statistically different between the two classes, as confirmed by statistical tests, including the t-test and Mann-Whitney U test. Visual analyses further supported these findings, with the Manhattan plot illustrating that most features exhibited significant differences between the two groups (Figure 22), and the QQ plot reinforcing the statistical robustness of these distinctions (Figure 23). These results collectively underscore the model's effectiveness in identifying key distinguishing features between mutant and wild-type populations. With their significance confirmed by statistical tests and visual analyses, the most statistically different features are crucial to explore, as they likely capture the average phenotype of the wild-type and mutant classes. Future works should plan to extract established morphological features, such as width, length, and nucleoid size, to serve as reference for comparison with the EfficientNet-derived features. This approach could help elucidate biological insights of the model abstract features. By analyzing potential correlations between the two embedding sets, biological features could be tied to complex EfficientNet-derived ones helping with the interpretability of downstream analysis.

5.3.2 Captured Phenotypic Divergence

Dimensionality reduction techniques (UMAP, PCA, t-SNE) and clustering of sample mean features revealed that certain mutant deletion strains, such as ΔycgV and ΔsucB, exhibit phenotypes markedly distinct from the wild type. For example, ycgV is homologous to the flu gene, which encodes an adhesin involved in bacterial adhesion and biofilm formation, suggesting its deletion may alter cell surface properties [154]. Similarly, sucB gene encodes a component of the 2-oxoglutarate dehydrogenase complex in the TCA cycle, and its deletion may disrupt energy metabolism, leading to phenotypic changes [155]. These captured phenotypic features may not necessarily align with the cell cycle trajectory, indicating that some

of the visual differences identified by the EfficientNet model are likely driven by traits unrelated to cell cycle dynamics. In contrast, mutants like ΔdamX, which encodes a cell division protein that interacts with FtsQ and FtsN [156], and Δrep, which encodes an accessory replicative helicase essential for aiding fork progression and resolving protein-DNA conflicts during replication [157], also appeared as outliers in the PCA plot (Figure 24), demonstrating that the model can detect changes related to cell cycle dynamics.

The model demonstrated high accuracy in distinguishing these mutants from the wild type, as shown by clear separation in clustering groups and dimensionality reduction visualizations (Figure 24, 25, 26), underscoring its ability to detect and differentiate a wide range of phenotypic traits effectively. Additionally, outliers within the wild-type group were observed, which could reflect natural genetic variation, experimental variability, or environmental stress responses, highlighting the model's sensitivity to subtle phenotypic deviations. However, while the model excels at identifying phenotypic differences, including those related to the cell cycle, its primary strength lies in broader phenotypic differentiation rather than specifically inferring cell cycle trajectories.

Nevertheless, the model is valuable for exploring genes with unknown functions, such as ygeG, which is located in a remnant pathogenicity island and may influence pathogenesis-related traits [158]. This deletion mutant appears as an outlier in PCA, indicating significant phenotypic changes. Future studies should focus on investigating these outlier mutants, particularly those with unknown or cell cycle-related functions, and exploring whether genes within identified clusters share related roles, potentially uncovering novel biological connections.

5.4 Assessing the Ability of EfficientNet Features to Infer the Cell Cycle Trajectory of Bacteria

The analysis was unable to confirm that the PHATE dimensionality reduction applied to EfficientNet-extracted features reliably captured bacterial cell cycle trajectories. This is likely due to a mismatch between the model's feature prioritization and cell cycle dynamics. Analysis of 269 PHATE plots (162 wild-type, 107 mutant) showed arc-shaped trajectories in many cases, with cell areas transitioning from approximately 400 pixel units (purple) to 1200 pixel units (yellow), suggesting progression through binary fission (Figure 25, 26). However, inconsistent arc patterns and deviations in area transitions in some plots indicate that PHATE struggled to align with bacterial cell cycle trajectories (Appendix A3, A4). PHATE is designed to visualize continuous transition states by preserving local and global data relationships, but the EfficientNet features, optimized for distinguishing wild-type from mutant cells, likely underrepresented critical cell cycle indicators such as cell length, cell constriction, nucleoid separation, DNA replication (SeqA foci), divisome formation, or septum formation. Additionally, reliance on default PHATE parameters may have limited its ability to capture the data's underlying structure.

Overlay trajectory and pair plots further confirmed distinct ranges for wild-type and mutant cells (Figure 27, 28), reinforcing the model's ability to separate these groups. However, the arc-

shaped trajectories lacked consistency across the dataset. According to Govers et al., 2024 approximately 20% of the mutant deletion strains in E. coli deviating from the wild-type cell cycle trajectory phenotype in a specific growth condition [109]. This suggests that the EfficientNet features did not adequately capture morphological variations essential for precise cell cycle inference, particularly in wild-type samples where consistent trajectories were expected. Consequently, PHATE could not fully model the intrinsic geometry of the cell cycle.

To improve cell cycle trajectory inference, future studies could focus on extracting cell cycle-related features from the EfficientNet embeddings—such as cell length, cell constriction, nucleoid separation, DNA replication (e.g., SeqA foci), divisome formation, and septum formation. Since these features vary continuously throughout the cell cycle, selecting high-variance features within the wild-type population may help isolate those most relevant to cell cycle progression. In contrast, features related to class differentiation are likely to show lower variance, as they are potentially more consistent across cells. Additionally, tuning PHATE parameters—such as the number of nearest neighbors or the number of diffusion steps—could further improve its ability to capture and visualize dynamic cell cycle trajectories.

6 Conclusion

Image-based profiling refers to the extraction of rich quantitative features from microscopy images to systematically capture and compare cellular morphology and phenotypes [1]. Over the past decade, this approach has advanced considerably, propelled by developments in deep learning and high-throughput imaging technologies. While eukaryotic cell profiling has seen widespread application and methodological development, bacterial image-based profiling remains comparatively underexplored. This thesis set out to bridge that gap by investigating whether modern deep learning techniques can be effectively adapted for bacterial imaging applications, with particular emphasis on brightfield microscopy segmentation and phenotypic profiling through automated feature extraction using large pretrained models.

The first major aim of this thesis was to validate the viability of brightfield microscopy—a low-cost and widely accessible imaging modality [159]—for accurate bacterial cell segmentation. To address this, we retrained a deep learning-based Omnipose segmentation model using brightfield images with ground-truth masks generated from phase contrast images. Our experiments demonstrated that segmentation performance improves significantly when leveraging pretrained weights via transfer learning, even when the training dataset is relatively small. Additionally, through systematic evaluation of model and segmentation hyperparameters—including batch size, mask threshold, and flow threshold—we optimized segmentation quality, achieving high Jaccard Index scores on both validation and test sets. These results establish that brightfield images can be a robust foundation for bacterial image segmentation when paired with an appropriate deep learning pipeline. However, this approach is not without limitations: the segmentation model was trained and evaluated on a limited number of bacterial species, and performance may vary for species with more complex or atypical morphologies. As such, generalization to other datasets or imaging conditions may require additional tuning, retraining, or domain adaptation strategies.

The second major aim of this thesis was to investigate whether convolutional neural networks could extract meaningful, unbiased features from bacterial images to enable downstream phenotypic analyses. To this end, we developed a CNN-based feature extraction pipeline using EfficientNet-B0, trained on multi-channel fluorescence images incorporating phase contrast, nucleoid (DAPI), FtsZ (VenusSW), and SeqA (mCherry) markers. By generating cell-centered patches and training the model in a supervised manner, we obtained 1280-dimensional embeddings for each cell, which were then subjected to dimensionality reduction techniques. This approach allowed us to identify phenotypically distinct subpopulations among mutant strains and infer putative cell cycle trajectories. Although a complete reconstruction of the cell cycle remained elusive, the deep feature embeddings proved highly informative, capturing biologically relevant signals such as morphological transitions related to DNA replication and cell division and effectively discriminating between mutants. Differential analysis of averaged wild-type and mutant samples further confirmed the biological relevance of these embeddings, with most features showing statistically significant differences as validated by t-tests, Mann-Whitney U tests, and visualizations including Manhattan and QQ plots. These results

underscore the model's ability to capture distinguishing phenotypic traits, yet the largely abstract nature of the extracted features limits direct biological interpretability. Future studies should therefore aim to correlate these deep embeddings with established morphological metrics—such as cell width, length, and nucleoid size—to enhance their interpretability. While dimensionality reduction and clustering analyses successfully revealed global phenotypic patterns, the absence of inherently interpretable features currently constrains deeper biological insights into the samples analyzed.

Nevertheless, the contributions of this thesis are twofold. First, it demonstrates that brightfield microscopy can be successfully harnessed for bacterial image-based profiling through deep learning. Second, it presents a complete computational pipeline—from segmentation to feature extraction to phenotypic analysis—that is adaptable, reproducible, and publicly available via GitHub. In conclusion, this thesis underscores the transformative potential of deep learning in bacterial image-based profiling. It lays the groundwork for future studies seeking to scale up bacterial phenotyping efforts, integrate multi-omics data, or apply these methods in different contexts such as antibiotic screening or functional genomics. Future research should enhance brightfield segmentation by leveraging transfer learning, incorporating early stopping strategies, improving ground truth annotations, and expanding datasets to include a wider range of bacterial types and experimental conditions. For feature extraction, linking EfficientNet-embeddings to morphological metrics will aid interpretation. Exploring outliers and refining PHATE could reveal new biological insights into cell cycle progression and advancing high-throughput bacterial phenotyping. Ultimately, this work advances the field of bacterial image analysis by demonstrating scalable methods for high-throughput, high-resolution image-based profiling.

7 References

1. Bray, M. A., Singh, S., Han, H., Davis, C. T., Borgeson, B., Hartland, C., ... & Carpenter, A. E. (2016). Cell Painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. Nature protocols, 11(9), 1757-1774.
2. Chandrasekaran, S. N., Ceulemans, H., Boyd, J. D., & Carpenter, A. E. (2021). Image-based profiling for drug discovery: due for a machine-learning upgrade?. Nature Reviews Drug Discovery, 20(2), 145-159.
3. Caicedo, J. C., Cooper, S., & Carpenter, A. E. (2017). Data-analysis strategies for image-based cell profiling. Nature Methods, 14(9), 849–863.
4. Kamble, N. S., Bera, S., Bhedase, S. A., Gaur, V., & Chowdhury, D. (2024). Review on applied applications of microbiome on human lives. Bacteria, 3(3), 141-159.
5. Huang, K. C. (2015). Applications of imaging for bacterial systems biology. Current Opinion in Microbiology, 27, 114-120.
6. Danielsen, J., & Nordenfelt, P. (2016). Computer vision-based image analysis of bacteria. In Bacterial Pathogenesis: Methods and Protocols (pp. 161-172). New York, NY: Springer New York.
7. Li, C., Tebo, A. G., & Gautier, A. (2017). Fluorogenic labeling strategies for biological imaging. International Journal of Molecular Sciences, 18(7), 1473.
8. Millard, T. P., Endrizzi, M., Ignatyev, K., Hagen, C. K., Munro, P. R. T., Speller, R. D., & Olivo, A. (2013). Method for automatization of the alignment of a laboratory based x-ray phase contrast edge illumination system. Review of Scientific Instruments, 84(8).
9. Spahn, C., Gómez-de-Mariscal, E., Laine, R. F., Pereira, P. M., von Chamier, L., Conduit, M., ... & Henriques, R. (2022). DeepBacs for multi-task bacterial image analysis using open-source deep learning approaches. Communications Biology, 5(1), 688.
10. Karkra, S., Singh, P., & Kaur, K. (2019). Convolution neural network: A shallow dive in to deep neural net technology. Int. J. Recent Technol. Eng, 8(2), 487-495.
11. Cutler, K. J., Stringer, C., Lo, T. W., Rappez, L., Stroustrup, N., Brook Peterson, S., ... & Mougous, J. D. (2022). Omnipose: a high-precision morphology-independent solution for bacterial cell segmentation. Nature methods, 19(11), 1438-1448.
12. Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning (pp. 6105-6114). PMLR.
13. Emanuel, G., Moffitt, J. R., & Zhuang, X. (2017). High-throughput, image-based screening of pooled genetic-variant libraries. Nature Methods, 14(9), 883–886.
14. Chandrasekaran, S. N., Ackerman, J., Alix, E., Ando, D. M., Arevalo, J., Bennion, M., ... & Carpenter, A. E. (2023). JUMP Cell Painting dataset: morphological impact of 136,000 chemical and genetic perturbations. BioRxiv, 2023-03.
15. Chandrasekaran, S. N., Cimini, B. A., Goodale, A., Miller, L., Kost-Alimova, M., Jamali, N., ... & Carpenter, A. E. (2024). Three million images and morphological

profiles of cells treated with matched chemical and genetic perturbations. Nature Methods, 1-8.
16. Seal, S., Trapotsi, M. A., Spjuth, O., Singh, S., Carreras-Puigvert, J., Greene, N., ... & Carpenter, A. E. (2024). A Decade in a Systematic Review: The Evolution and Impact of Cell Painting. ArXiv.
17. Gustafsdottir, S. M., Ljosa, V., Sokolnicki, K. L., Anthony Wilson, J., Walpita, D., Kemp, M. M., ... & Shamji, A. F. (2013). Multiplex cytological profiling assay to measure diverse cellular states. PloS one, 8(12), e80999.
18. Lakowicz, J. R., & Masters, B. R. (2008). Principles of fluorescence spectroscopy. Journal of Biomedical Optics, 13(2), 029901.
19. Bucevičius, J., Lukinavičius, G., & Gerasimaitė, R. (2018). The use of hoechst dyes for DNA staining and beyond. Chemosensors, 6(2), 18.
20. Wulf, E., Deboben, A., Bautz, F. A., Faulstich, H., & Wieland, T. (1979). Fluorescent phallotoxin, a tool for the visualization of cellular actin. Proceedings of the national academy of sciences, 76(9), 4498-4502.
21. Pendergrass, W., Wolf, N., & Poot, M. (2004). Efficacy of MitoTracker Green™ and CMXrosamine to measure changes in mitochondrial membrane potentials in living cells and tissues. Cytometry Part A: the journal of the International Society for Analytical Cytology, 61(2), 162-169.
22. Thermo Fisher. (2004). SYTO® RNASelect™ green fluorescent cell stain (S32703). Thermo Fisher Scientific. https://assets.thermofisher.com/TFS-Assets/LSG/manuals/mp32703.pdf
23. Wu, Y., Liu, Y., Lu, C., Lei, S., Li, J., & Du, G. (2020). Quantitation of RNA by a fluorometric method using the SYTO RNASelect stain. Analytical biochemistry, 606, 113857.
24. Vida, T. A., & Emr, S. D. (1995). A new vital stain for visualizing vacuolar membrane dynamics and endocytosis in yeast. The Journal of cell biology, 128(5), 779-792.
25. Kuhry, J. G., Fonteneau, P., Duportail, G., Maechling, C., & Laustriat, G. (1983). TMA-DPH: a suitable fluorescence polarization probe for specific plasma membrane fluidity studies in intact living cells. Cell biophysics, 5(2), 129-140.
26. Michael, S., Auld, D., Klumpp, C., Jadhav, A., Zheng, W., Thorne, N., Austin, C. P., Inglese, J., & Simeonov, A. (2008). A robotic platform for quantitative high-throughput screening. Assay and drug development technologies, 6(5), 637–657. https://doi.org/10.1089/adt.2008.150.
27. Oreopoulos, J., Berman, R., & Browne, M. (2014). Spinning-disk confocal microscopy: present technology and future trends. Methods in cell biology, 123, 153-175.
28. Li, C., Moatti, A., Zhang, X., Troy Ghashghaei, H., & Greenbaum, A. (2021). Deep learning-based autofocus method enhances image quality in light-sheet fluorescence microscopy. Biomedical Optics Express, 12(8), 5214-5226.
29. Forte, P. M., Felgueiras, P. E. R., Ferreira, F. P., Sousa, M. A., Nunes-Pereira, E. J., Bret, B. P., & Belsley, M. S. (2017). Exploring combined dark and bright field illumination to improve the detection of defects on specular surfaces. Optics and Lasers in Engineering, 88, 120-128.
30. Zernike, F. (1935). Phase contrast. Z Tech Physik, 16, 454.

31. Rao, J., Dragulescu-Andrasi, A., & Yao, H. (2007). Fluorescence imaging in vivo: recent advances. Current opinion in biotechnology, 18(1), 17-25.
32. Kirillov, A., He, K., Girshick, R., Rother, C., & Dollár, P. (2019). Panoptic segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 9404-9413).
33. Xu, R., Li, Y., Wang, C., Xu, S., Meng, W., & Zhang, X. (2022). Instance segmentation of biological images using graph convolutional network. Engineering Applications of Artificial Intelligence, 110, 104739.
34. Asgari Taghanaki, S., Abhishek, K., Cohen, J. P., Cohen-Adad, J., & Hamarneh, G. (2021). Deep semantic segmentation of natural and medical images: a review. Artificial Intelligence Review, 54, 137-178.
35. Carpenter, A. E., Jones, T. R., Lamprecht, M. R., Clarke, C., Kang, I. H., Friman, O., ... & Sabatini, D. M. (2006). CellProfiler: image analysis software for identifying and quantifying cell phenotypes. Genome biology, 7, 1-11.
36. Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., ... & Cardona, A. (2012). Fiji: an open-source platform for biological-image analysis. Nature methods, 9(7), 676-682.
37. Legland, D., Arganda-Carreras, I., & Andrey, P. (2016). MorphoLibJ: integrated library and plugins for mathematical morphology with ImageJ. Bioinformatics, 32(22), 3532-3534.
38. McQuin, C., Goodman, A., Chernyshev, V., Kamentsky, L., Cimini, B. A., Karhohs, K. W., ... & Carpenter, A. E. (2018). CellProfiler 3.0: Next-generation image processing for biology. PLoS biology, 16(7), e2005970.
39. Bannon, D., Moen, E., Schwartz, M., Borba, E., Kudo, T., Greenwald, N., ... & Van Valen, D. (2021). DeepCell Kiosk: scaling deep learning–enabled cellular image analysis with Kubernetes. Nature methods, 18(1), 43-45.
40. Greenwald, N. F., Miller, G., Moen, E., Kong, A., Kagel, A., Dougherty, T., ... & Van Valen, D. (2022). Whole-cell segmentation of tissue images with human-level performance using large-scale data annotation and deep learning. Nature biotechnology, 40(4), 555-565.
41. Bannon, D., Moen, E., Schwartz, M., Borba, E., Kudo, T., Greenwald, N., ... & Van Valen, D. (2021). DeepCell Kiosk: scaling deep learning–enabled cellular image analysis with Kubernetes. Nature methods, 18(1), 43-45.
42. Greenwald, N. F., Miller, G., Moen, E., Kong, A., Kagel, A., Dougherty, T., ... & Van Valen, D. (2022). Whole-cell segmentation of tissue images with human-level performance using large-scale data annotation and deep learning. Nature biotechnology, 40(4), 555-565.
43. Van Valen, D. A., Kudo, T., Lane, K. M., Macklin, D. N., Quach, N. T., DeFelice, M. M., ... & Covert, M. W. (2016). Deep learning automates the quantitative analysis of individual cells in live-cell imaging experiments. PLoS computational biology, 12(11), e1005177.
44. Huang, K., Xu, Y., Feng, T., Lan, H., Ling, F., Xiang, H., & Liu, Q. (2024). The Advancement and Application of the Single-Cell Transcriptome in Biological and Medical Research. Biology, 13(6), 451. https://doi.org/10.3390/biology13060451

45. Mao, X., Xia, D., Xu, M., Gao, Y., Tong, L., Lu, C., ... & Yuan, S. (2024). Single-Cell Simultaneous Metabolome and Transcriptome Profiling Revealing Metabolite-Gene Correlation Network. Advanced Science, 2411276.
46. Huang, K., Xu, Y., Feng, T., Lan, H., Ling, F., Xiang, H., & Liu, Q. (2024). The Advancement and Application of the Single-Cell Transcriptome in Biological and Medical Research. Biology, 13(6), 451. https://doi.org/10.3390/biology13060451
47. Williams, C.G., Lee, H.J., Asatsuma, T. et al. An introduction to spatial transcriptomics for biomedical research. Genome Med 14, 68 (2022). https://doi.org/10.1186/s13073-022-01075-1
48. Jiang, X., Wang, S., Guo, L. et al. iIMPACT: integrating image and molecular profiles for spatial transcriptomics analysis. Genome Biol 25, 147 (2024). https://doi.org/10.1186/s13059-024-03289-5
49. Scheeder, C., Heigwer, F., & Boutros, M. (2018). Machine learning and image-based profiling in drug discovery. Current opinion in systems biology, 10, 43–52. https://doi.org/10.1016/j.coisb.2018.05.004
50. Gurusamy, V., Kannan, S., & Nalini, G. (2013). Review on image segmentation techniques. J Pharm Res, 20125, 4548-4553.
51. Xu, Y., Quan, R., Xu, W., Huang, Y., Chen, X., & Liu, F. (2024). Advances in medical image segmentation: a comprehensive review of traditional, deep learning and hybrid approaches. Bioengineering, 11(10), 1034.
52. Vohra, S. K., & Prodanov, D. (2021). The active segmentation platform for microscopic image classification and segmentation. Brain Sciences, 11(12), 1645.
53. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18 (pp. 234-241). Springer International Publishing.
54. He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).
55. Zou, J., Han, Y., & So, S. S. (2009). Overview of artificial neural networks. Artificial neural networks: methods and applications, 14-22.
56. Hijazi, S., Kumar, R., & Rowen, C. (2015). Using convolutional neural networks for image recognition. Cadence Design Systems Inc.: San Jose, CA, USA, 9(1).
57. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.
58. Hollandi, R., Szkalisity, A., Toth, T., Tasnadi, E., Molnar, C., Mathe, B., ... & Horvath, P. (2019). A deep learning framework for nucleus segmentation using image style transfer. Biorxiv, 580605.
59. N. Siddique, S. Paheding, C. P. Elkin and V. Devabhaktuni, "U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications," in IEEE Access, vol. 9, pp. 82031-82057, 2021, doi: 10.1109/ACCESS.2021.3086020.
60. Johnson, J. W. (2018). Adapting mask-rcnn for automatic nucleus segmentation. arXiv preprint arXiv:1805.00500.

61. Alzubaidi, L., Zhang, J., Humaidi, A.J. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J Big Data 8, 53 (2021). https://doi.org/10.1186/s40537-021-00444-8
62. Moshkov, N., Bornholdt, M., Benoit, S., Smith, M., McQuin, C., Goodman, A., ... & Caicedo, J. C. (2024). Learning representations for image-based profiling of perturbations. Nature communications, 15(1), 1594.
63. Caicedo, J. C., McQuin, C., Goodman, A., Singh, S., & Carpenter, A. E. (2018). Weakly supervised learning of single-cell feature embeddings. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 9309-9318).
64. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.
65. Kim, H.E., Cosa-Linan, A., Santhanam, N. et al. Transfer learning for medical image classification: a literature review. BMC Med Imaging 22, 69 (2022). https://doi.org/10.1186/s12880-022-00793-7
66. Guan, H., & Liu, M. (2022). Domain Adaptation for Medical Image Analysis: A Survey. IEEE transactions on bio-medical engineering, 69(3), 1173–1185. https://doi.org/10.1109/TBME.2021.3117407
67. Moen, E., Bannon, D., Kudo, T., Graf, W., Covert, M., & Van Valen, D. (2019). Deep learning for cellular image analysis. Nature methods, 16(12), 1233-1246.
68. Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016). 3D U-Net: learning dense volumetric segmentation from sparse annotation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19 (pp. 424-432). Springer International Publishing.
69. Zhang, C., Gai, K., & Zhang, S. (2019). Matrix normal pca for interpretable dimension reduction and graphical noise modeling. arXiv preprint arXiv:1911.10796.
70. Simmons, S., Peng, J., Bienkowska, J., & Berger, B. (2015). Discovering What Dimensionality Reduction Really Tells Us About RNA-Seq Data. Journal of computational biology : a journal of computational molecular cell biology, 22(8), 715–728. https://doi.org/10.1089/cmb.2015.0085
71. Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).
72. Kobak, D., & Berens, P. (2019). The art of using t-SNE for single-cell transcriptomics. Nature communications, 10(1), 5416.
73. McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426.
74. Becht, E., McInnes, L., Healy, J., Dutertre, C. A., Kwok, I. W., Ng, L. G., ... & Newell, E. W. (2019). Dimensionality reduction for visualizing single-cell data using UMAP. Nature biotechnology, 37(1), 38-44.
75. Moon, K. R., Van Dijk, D., Wang, Z., Gigante, S., Burkhardt, D. B., Chen, W. S., ... & Krishnaswamy, S. (2019). Visualizing structure and transitions in high-dimensional biological data. Nature biotechnology, 37(12), 1482-1492.

76. Ando, D. M., McLean, C. Y., & Berndl, M. (2017). Improving phenotypic measurements in high-content imaging screens. BioRxiv, 161422.
77. Liaw, A. (2002). Classification and regression by randomForest. R news.
78. Jin, M., Govindarajan, L. N., & Cheng, L. (2014, April). A random-forest random field approach for cellular image segmentation. In 2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI) (pp. 1251-1254). IEEE.
79. Nguyen, T. T., Huang, J. Z., & Nguyen, T. T. (2015). Unbiased Feature Selection in Learning Random Forests for High-Dimensional Data. The Scientific World Journal, 2015(1), 471371.
80. Cortes, C. (1995). Support-Vector Networks. Machine Learning.
81. Jones, T. R., Carpenter, A. E., Lamprecht, M. R., Moffat, J., Silver, S. J., Grenier, J. K., ... & Sabatini, D. M. (2009). Scoring diverse cellular morphologies in image-based screens with iterative feedback and machine learning. Proceedings of the National Academy of Sciences, 106(6), 1826-1831.
82. Lopez-Martinez, D. (2017). Regularization approaches for support vector machines with applications to biomedical data. arXiv preprint arXiv:1710.10600.
83. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
84. Zhao, W., Guo, Y., Yang, S., Chen, M., & Chen, H. (2020). Fast intelligent cell phenotyping for high-throughput optofluidic time-stretch microscopy based on the XGBoost algorithm. Journal of biomedical optics, 25(6), 066001-066001.
85. van Hoof, J., & Vanschoren, J. (2021). Hyperboost: Hyperparameter optimization by gradient boosting surrogate models. arXiv preprint arXiv:2101.02289.
86. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.
87. Pawlowski, N., Caicedo, J. C., Singh, S., Carpenter, A. E., & Storkey, A. (2016). Automating morphological profiling with generic deep convolutional networks. BioRxiv, 085118.
88. Liu, K., Kang, G., Zhang, N., & Hou, B. (2018). Breast cancer classification based on fully-connected layer first convolutional neural networks. IEEE Access, 6, 23722-23732.
89. Xu, Rui, and Donald Wunsch. "Survey of clustering algorithms." IEEE Transactions on neural networks 16.3 (2005): 645-678.
90. Lloyd, S. (1982). Least squares quantization in PCM. IEEE transactions on information theory, 28(2), 129-137.
91. Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. Journal of the royal statistical society. series c (applied statistics), 28(1), 100-108.
92. Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226-231).
93. Reynolds, D. A. (2009). Gaussian mixture models. Encyclopedia of biometrics, 741(659-663).

94. Santoso, H. A., & Haw, S. C. (2023). Improvement of k-Means Clustering Performance on Disease Clustering using Gaussian Mixture Model. Journal of System and Management Sciences, 13(5), 169-179.
95. Tran, D. T., & Pham, T. D. (2006, October). Relaxation labeling for cell phase identification. In 2006 IEEE International Conference on Systems, Man and Cybernetics (Vol. 2, pp. 1275-1280). IEEE.
96. Murtagh, F. (1983). A survey of recent advances in hierarchical clustering algorithms. The computer journal, 26(4), 354-359.
97. Jeon, Y., Yoo, J., Lee, J., & Yoon, S. (2017). Nc-link: A new linkage method for efficient hierarchical clustering of large-scale data. IEEE Access, 5, 5594-5608.
98. Bayer, M. E., & Remsen, C. C. (1970). Structure of Escherichia coli after freeze-etching. Journal of Bacteriology, 101(1), 304-313.
99. Cesar, S., & Huang, K. C. (2017). Thinking big: the tunability of bacterial cell size. FEMS microbiology reviews, 41(5), 672-678.
100. Stanislav, K. (2005). Light microscopy in biological research. Biophysical journal, 88(6), 3741.
101. Valli, J., Garcia-Burgos, A., Rooney, L. M., e Oliveira, B. V. D. M., Duncan, R. R., & Rickman, C. (2021). Seeing beyond the limit: A guide to choosing the right super-resolution microscopy technique. Journal of Biological Chemistry, 297(1).
102. Chong, T. N., & Shapiro, L. (2024). Bacterial cell differentiation enables population level survival strategies. Mbio, 15(6), e00758-24.
103. Jaramillo-Riveri, S., Broughton, J., McVey, A., Pilizota, T., Scott, M., & El Karoui, M. (2022). Growth-dependent heterogeneity in the DNA damage response in Escherichia coli. Molecular Systems Biology, 18(5), e10441.
104. Govers, S. K., & Jacobs-Wagner, C. (2020). Caulobacter crescentus: model system extraordinaire. Current Biology, 30(19), R1151-R1158.
105. van Teeseling, M. C., de Pedro, M. A., & Cava, F. (2017). Determinants of bacterial morphology: from fundamentals to possibilities for antimicrobial targeting. Frontiers in microbiology, 8, 1264.
106. Cellini, L., Allocati, N., Angelucci, D., Iezzi, T., Campli, E. D., Marzio, L., & Dainelli, B. (1994). Coccoid Helicobacter pylori not culturable in vitro reverts in mice. Microbiology and immunology, 38(11), 843-850.
107. Karandikar, A., Sharples, G. P., & Hobbs, G. (1997). Differentiation of Streptomyces coelicolor A3 (2) under nitrate-limited conditions. Microbiology, 143(11), 3581-3590.
108. Kratz, J. C., & Banerjee, S. (2023). Dynamic proteome trade-offs regulate bacterial cell size and growth in fluctuating nutrient environments. Communications Biology, 6(1), 486.
109. Govers, S. K., Campos, M., Tyagi, B., Laloux, G., & Jacobs-Wagner, C. (2024). Apparent simplicity and emergent robustness in the control of the Escherichia coli cell cycle. Cell Systems, 15(1), 19-36.
110. Giavazzi, F., Brogioli, D., Trappe, V., Bellini, T., & Cerbino, R. (2009). Scattering information obtained by optical microscopy: differential dynamic

microscopy and beyond. Physical Review E—Statistical, Nonlinear, and Soft Matter Physics, 80(3), 031403.

111. Ali, M. A., Hollo, K., Laasfeld, T., Torp, J., Tahk, M. J., Rinken, A., ... & Fishman, D. (2022). ArtSeg—Artifact segmentation and removal in brightfield cell microscopy images without manual pixel-level annotations. Scientific Reports, 12(1), 11404.

112. Buggenthin, F., Marr, C., Schwarzfischer, M., Hoppe, P. S., Hilsenbeck, O., Schroeder, T., & Theis, F. J. (2013). An automatic method for robust and fast cell detection in bright field images from high-throughput microscopy. BMC bioinformatics, 14, 1-12.

113. Yin, Z., Kanade, T., & Chen, M. (2012). Understanding the phase contrast optics to restore artifact-free microscopy images for segmentation. Medical image analysis, 16(5), 1047-1062.

114. Campos, M., & Jacobs-Wagner, C. (2013). Cellular organization of the transfer of genetic information. Current opinion in microbiology, 16(2), 171-176.

115. Longin, A., Souchier, C., Ffrench, M., & Bryon, P. A. (1993). Comparison of anti-fading agents used in fluorescence microscopy: image analysis and laser confocal microscopy study. Journal of Histochemistry & Cytochemistry, 41(12), 1833-1840.

116. Cormack, B. P., Valdivia, R. H., & Falkow, S. (1996). FACS-optimized mutants of the green fluorescent protein (GFP). Gene, 173(1), 33-38.

117. Glick, B. R. (1995). Metabolic load and heterologous gene expression. Biotechnology advances, 13(2), 247-261.

118. Schirripa Spagnolo, C., Moscardini, A., Amodeo, R., Beltram, F., & Luin, S. (2023). Quantitative determination of fluorescence labeling implemented in cell cultures. BMC biology, 21(1), 190.

119. Smith, T. C., Pullen, K. M., Olson, M. C., McNellis, M. E., Richardson, I., Hu, S., ... & Aldridge, B. B. (2020). Morphological profiling of tubercle bacilli identifies drug pathways of action. Proceedings of the National Academy of Sciences, 117(31), 18744-18753.

120. Martínez-Antonio, A., Medina-Rivera, A., & Collado-Vides, J. (2009). Structural and functional map of a bacterial nucleoid. Genome biology, 10, 1-4.

121. Wu, L. J., & Errington, J. (2012). Nucleoid occlusion and bacterial cell division. Nature Reviews Microbiology, 10(1), 8-12.

122. Floc'h, K., Lacroix, F., Servant, P., Wong, Y. S., Kleman, J. P., Bourgeois, D., & Timmins, J. (2019). Cell morphology and nucleoid dynamics in dividing Deinococcus radiodurans. Nature communications, 10(1), 3815.

123. Zaritsky, A., Woldringh, C. L., & Männik, J. (Eds.). (2016). The bacterial cell: coupling between growth, nucleoid replication, cell division and shape. Frontiers Media SA.

124. Adams, D. W., Wu, L. J., & Errington, J. (2015). Nucleoid occlusion protein N oc recruits DNA to the bacterial cell membrane. The EMBO journal, 34(4), 491-501.

125. Dillon, S. C., & Dorman, C. J. (2010). Bacterial nucleoid-associated proteins, nucleoid structure and gene expression. Nature Reviews Microbiology, 8(3), 185-195.

126. Mott, M. L., & Berger, J. M. (2007). DNA replication initiation: mechanisms and regulation in bacteria. Nature Reviews Microbiology, 5(5), 343-354.
127. Loose, M., & Mitchison, T. J. (2014). The bacterial cell division proteins FtsA and FtsZ self-organize into dynamic cytoskeletal patterns. Nature cell biology, 16(1), 38-46.
128. Squyres, G. R., Holmes, M. J., Barger, S. R., Pennycook, B. R., Ryan, J., Yan, V. T., & Garner, E. C. (2021). Single-molecule imaging reveals that Z-ring condensation is essential for cell division in Bacillus subtilis. Nature microbiology, 6(5), 553-562.
129. Adams, D. W., Wu, L. J., Czaplewski, L. G., & Errington, J. (2011). Multiple effects of benzamide antibiotics on FtsZ function. Molecular microbiology, 80(1), 68-84.
130. Dörr, T., Moynihan, P. J., & Mayer, C. (2019). Bacterial cell wall structure and dynamics. Frontiers in microbiology, 10, 2051.
131. Coyette, J., & Van Der Ende, A. (2008). Peptidoglycan: the bacterial Achilles heel. FEMS microbiology reviews, 32(2), 147-148.
132. Maldonado, R. F., Sá-Correia, I., & Valvano, M. A. (2016). Lipopolysaccharide modification in Gram-negative bacteria during chronic infection. FEMS microbiology reviews, 40(4), 480-493.
133. Claessen, D., & Errington, J. (2019). Cell wall deficiency as a coping strategy for stress. Trends in microbiology, 27(12), 1025-1033.
134. Lichius, A., & Zeilinger, S. (2019). Application of membrane and cell wall selective fluorescent dyes for live-cell imaging of filamentous fungi. J. Vis. Exp, 153(7).
135. Harris, L. G. (2022). Microbial Cell Structure and Organization: Bacteria.
136. Vinella, D., & D'Ari, R. (1995). Overview of controls in the Escherichia coli cell cycle. Bioessays, 17(6), 527-536.
137. Wang, J. D., & Levin, P. A. (2009). Metabolism, cell growth and the bacterial cell cycle. Nature Reviews Microbiology, 7(11), 822-827.
138. Lutkenhaus, J., & Addinall, S. G. (1997). Bacterial cell division and the Z ring. Annual review of biochemistry, 66(1), 93-116.
139. Haeusser, D. P., Schwartz, R. L., Smith, A. M., Oates, M. E., & Levin, P. A. (2004). EzrA prevents aberrant cell division by modulating assembly of the cytoskeletal protein FtsZ. Molecular microbiology, 52(3), 801-814.
140. Nonejuie, P., Burkart, M., Pogliano, K., & Pogliano, J. (2013). Bacterial cytological profiling rapidly identifies the cellular pathways targeted by antibacterial molecules. Proceedings of the National Academy of Sciences, 110(40), 16169-16174.
141. Quach, D., Sharp, M., Ahmed, S., Ames, L., Bhagwat, A., Deshpande, A., ... & Sugie, J. (2025). Deep learning–driven bacterial cytological profiling to determine antimicrobial mechanisms in Mycobacterium tuberculosis. Proceedings of the National Academy of Sciences, 122(6), e2419813122.
142. Krentzel, D., Kho, K., Petit, J., Mahtal, N., Shorte, S. L., Wehenkel, A. M., ... & Zimmer, C. (2025). Deep learning recognises antibiotic modes of action from brightfield images. bioRxiv, 2025-03.

143. Sondervorst, K., Nesporova, K., Herdman, M., Steemans, B., Rosseels, J., & Govers, S. K. (2025). Complex interplay between gene deletions and the environment uncovers cellular roles for genes of unknown function in Escherichia coli. bioRxiv, 2025-02.
144. Shi, Aiqin, Feiyu Fan, and James R. Broach. "Microbial adaptive evolution." Journal of Industrial Microbiology and Biotechnology 49.2 (2022): kuab076.
145. Laskowska, E., & Kuczyńska-Wiśnik, D. (2020). New insight into the mechanisms protecting bacteria during desiccation. Current genetics, 66(2), 313-318.
146. Ramoneda, J., Fan, K., Lucas, J. M., Chu, H., Bissett, A., Strickland, M. S., & Fierer, N. (2024). Ecological relevance of flagellar motility in soil bacterial communities. The ISME Journal, 18(1), wrae067.
147. Minic, Z., & Thongbam, P. D. (2011). The biological deep sea hydrothermal vent as a model to study carbon dioxide capturing enzymes. Marine drugs, 9(5), 719-738.
148. Osborne, P., Hall, L. J., Kronfeld-Schor, N., Thybert, D., & Haerty, W. (2020). A rather dry subject; investigating the study of arid-associated microbial communities. Environmental Microbiome, 15, 1-14.
149. Leggieri, P. A., Liu, Y., Hayes, M., Connors, B., Seppälä, S., O'Malley, M. A., & Venturelli, O. S. (2021). Integrating systems and synthetic biology to understand and engineer microbiomes. Annual Review of Biomedical Engineering, 23(1), 169-201.
150. Luo, C., Li, X., Wang, L., He, J., Li, D., & Zhou, J. (2018, November). How does the data set affect CNN-based image classification performance?. In 2018 5th international conference on systems and informatics (ICSAI) (pp. 361-366). IEEE.
151. Samala, R. K., Chan, H. P., Hadjiiski, L., Helvie, M. A., Richter, C. D., & Cha, K. H. (2018). Breast cancer diagnosis in digital breast tomosynthesis: effects of training sample size on multi-stage transfer learning using deep neural nets. IEEE transactions on medical imaging, 38(3), 686-696.
152. Jaccard, N., Szita, N., & Griffin, L. D. (2017). Segmentation of phase contrast microscopy images based on multi-scale local basic image features histograms. Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization, 5(5), 359-367.
153. Binici, R. C., Şahin, U., Ayanzadeh, A., Töreyin, B. U., Önal, S., Okvur, D. P., ... & Ünay, D. (2019, October). Automated segmentation of cells in phase contrast optical microscopy time series images. In 2019 Medical Technologies Congress (TIPTEKNO) (pp. 1-4). IEEE.
154. Roux, A., Beloin, C., & Ghigo, J. M. (2005). Combined inactivation and expression strategy to study gene function under physiological conditions: application to identification of new Escherichia coli adhesins. Journal of bacteriology, 187(3), 1001–1013. https://doi.org/10.1128/JB.187.3.1001-1013.2005
155. Bunik, V. I., & Degtyarev, D. (2008). Structure-function relationships in the 2-oxo acid dehydrogenase family: substrate-specific signatures and functional predictions for the 2-oxoglutarate dehydrogenase-like proteins. Proteins, 71(2), 874–890. https://doi.org/10.1002/prot.21766

156. Gerding, M. A., Liu, B., Bendezú, F. O., Hale, C. A., Bernhardt, T. G., & de Boer, P. A. (2009). Self-enhanced accumulation of FtsN at Division Sites and Roles for Other Proteins with a SPOR domain (DamX, DedD, and RlpA) in Escherichia coli cell constriction. Journal of bacteriology, 191(24), 7383–7401. https://doi.org/10.1128/JB.00811-09

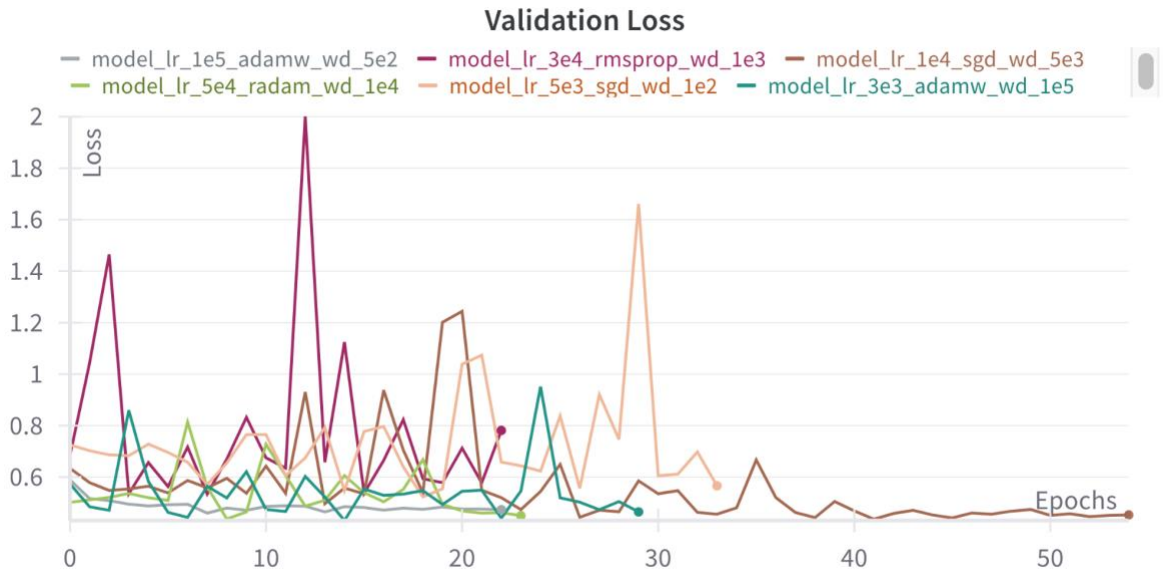
157. Boubakri, H., De Septenville, A. L., Viguera, E., & Michel, B. (2010). The helicases DinG, Rep and UvrD cooperate to promote replication across transcription units in vivo. The EMBO journal, 29(1), 145-157.

158. Ren, C. P., Chaudhuri, R. R., Fivian, A., Bailey, C. M., Antonio, M., Barnes, W. M., & Pallen, M. J. (2004). The ETT2 gene cluster, encoding a second type III secretion system from Escherichia coli, is present in the majority of strains but has undergone widespread mutational attrition. Journal of bacteriology, 186(11), 3547–3560. https://doi.org/10.1128/JB.186.11.3547-3560.2004

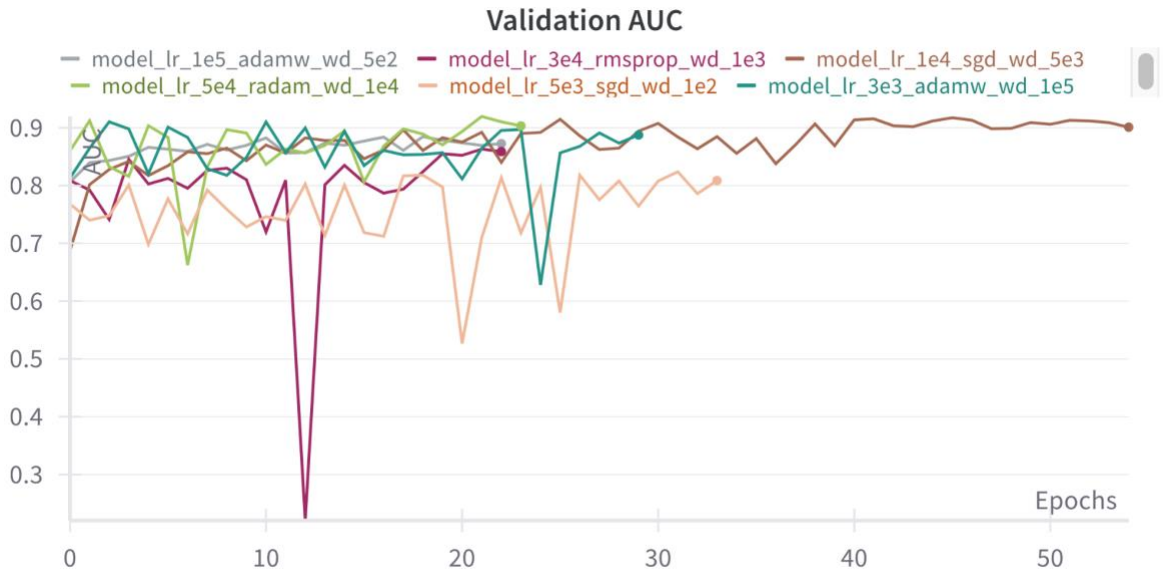
159. Emeigh, C., & Ryu, S. (2024, July). Using Brightfield Microscopy to Assess the Balloon Expansion Performance of a Microfluidic Cell Compression Device. In Fluids Engineering Division Summer Meeting (Vol. 88131, p. V002T06A002). American Society of Mechanical Engineers.

Appendix

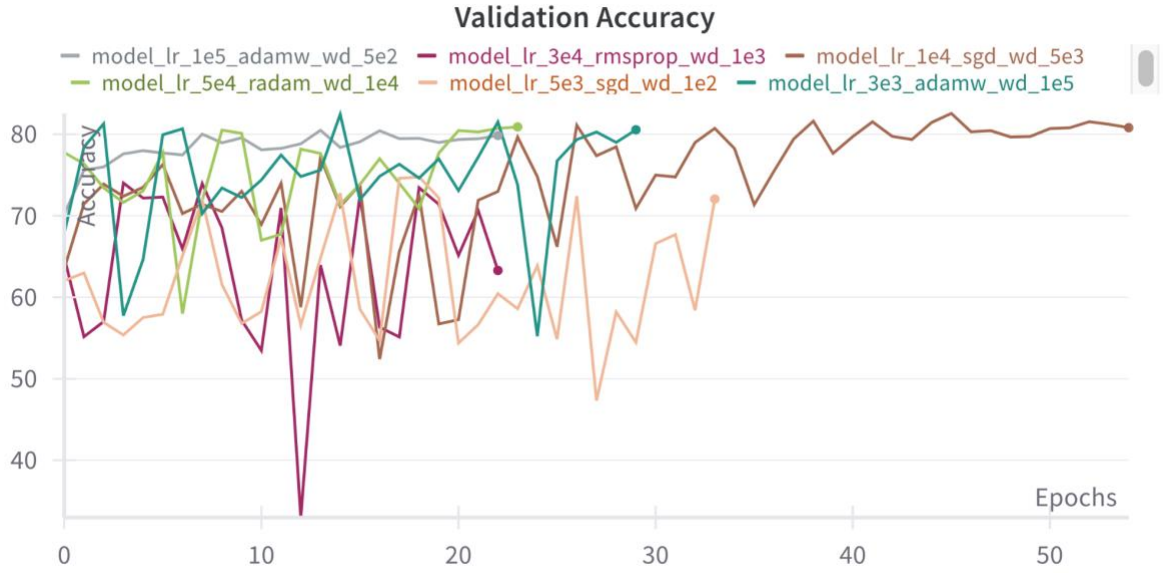
A.1 EfficientNet Performance Evaluation



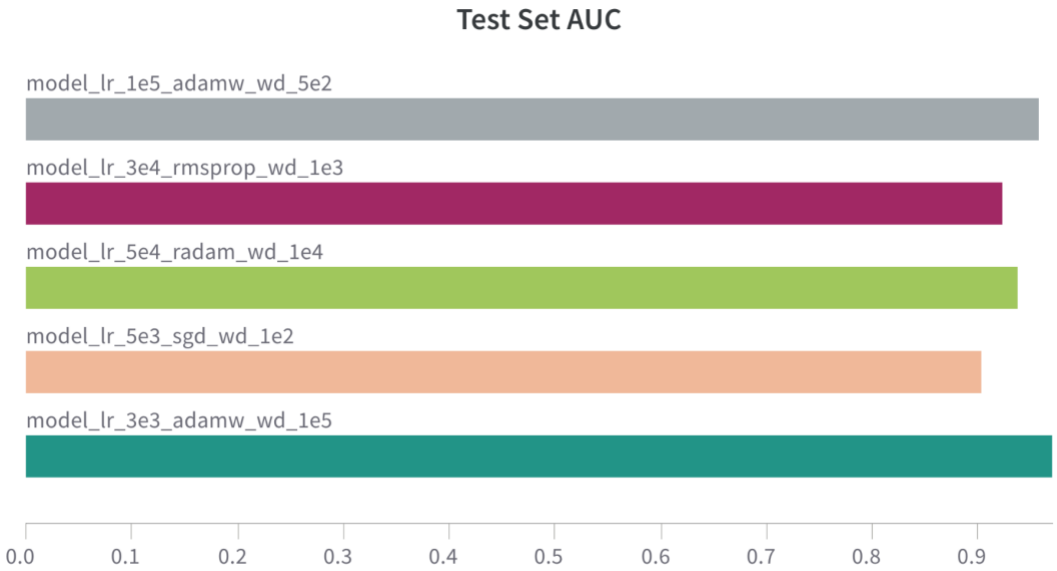
A.1.1: Validation Loss of all EfficientNet models trained.



A.1.2: Validation AUC of all EfficientNet models trained.

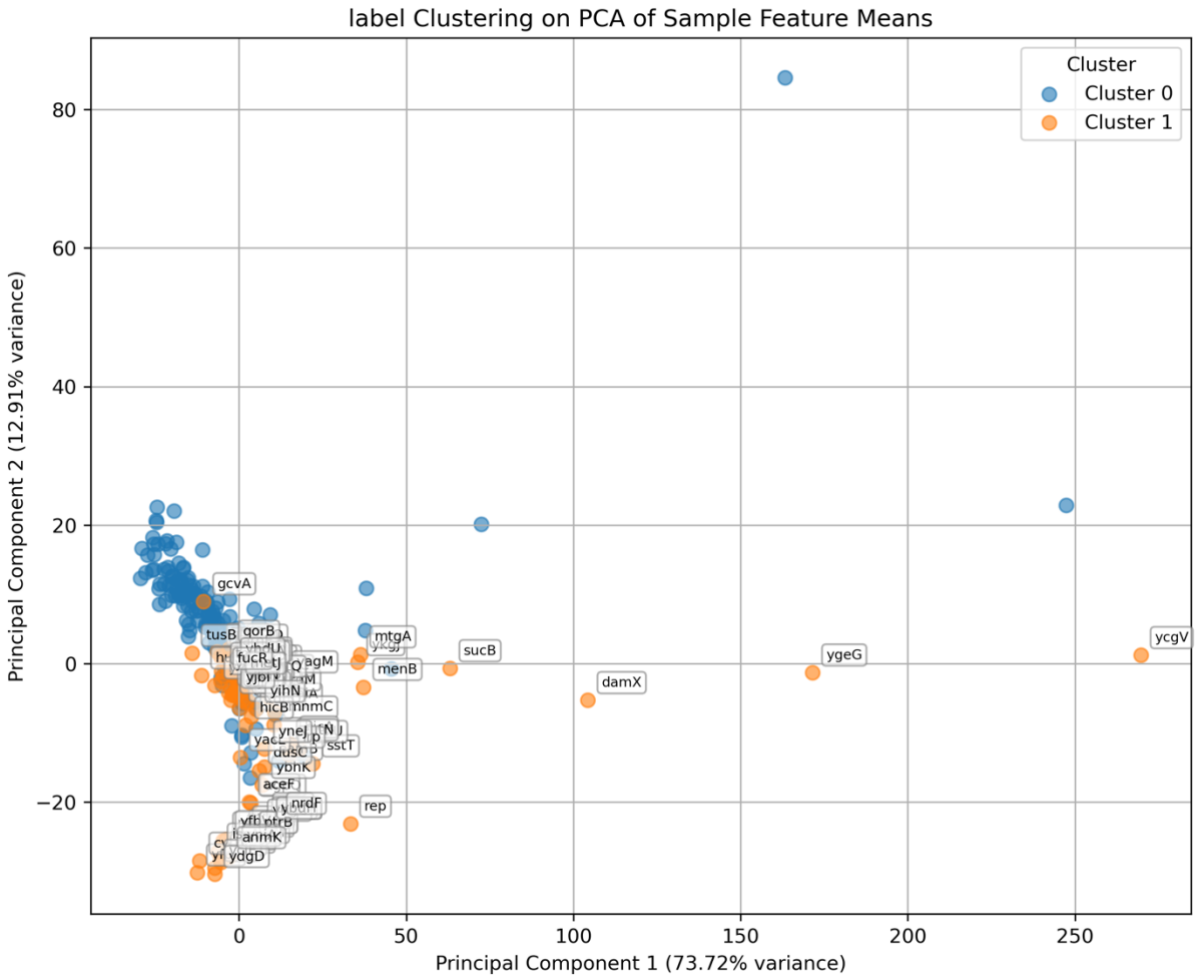


A.1.3: Validation Accuracy of all EfficientNet models trained.

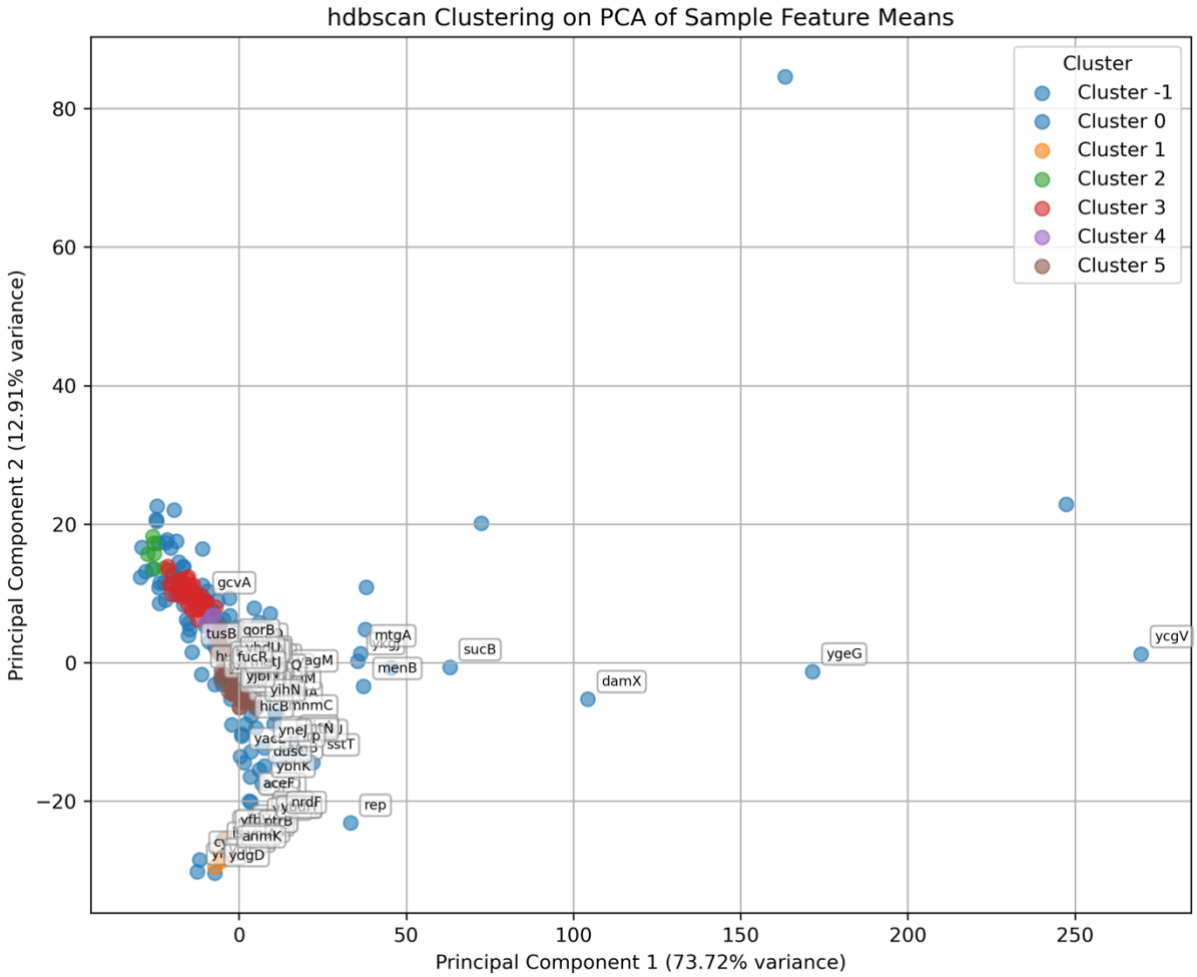


A.1.4: Test set AUC of all EfficientNet models trained.

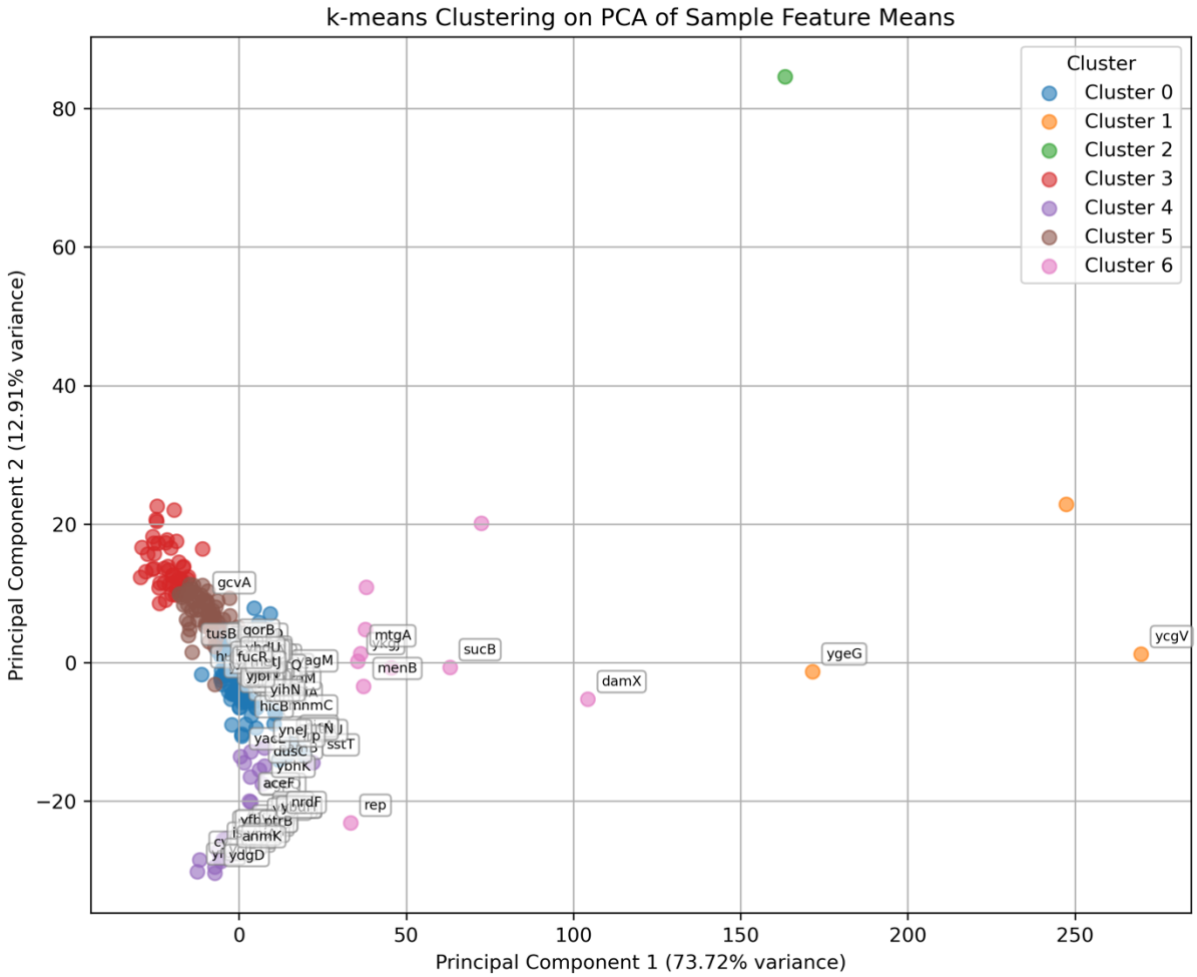
A.2 Dimension Reduction and Clustering



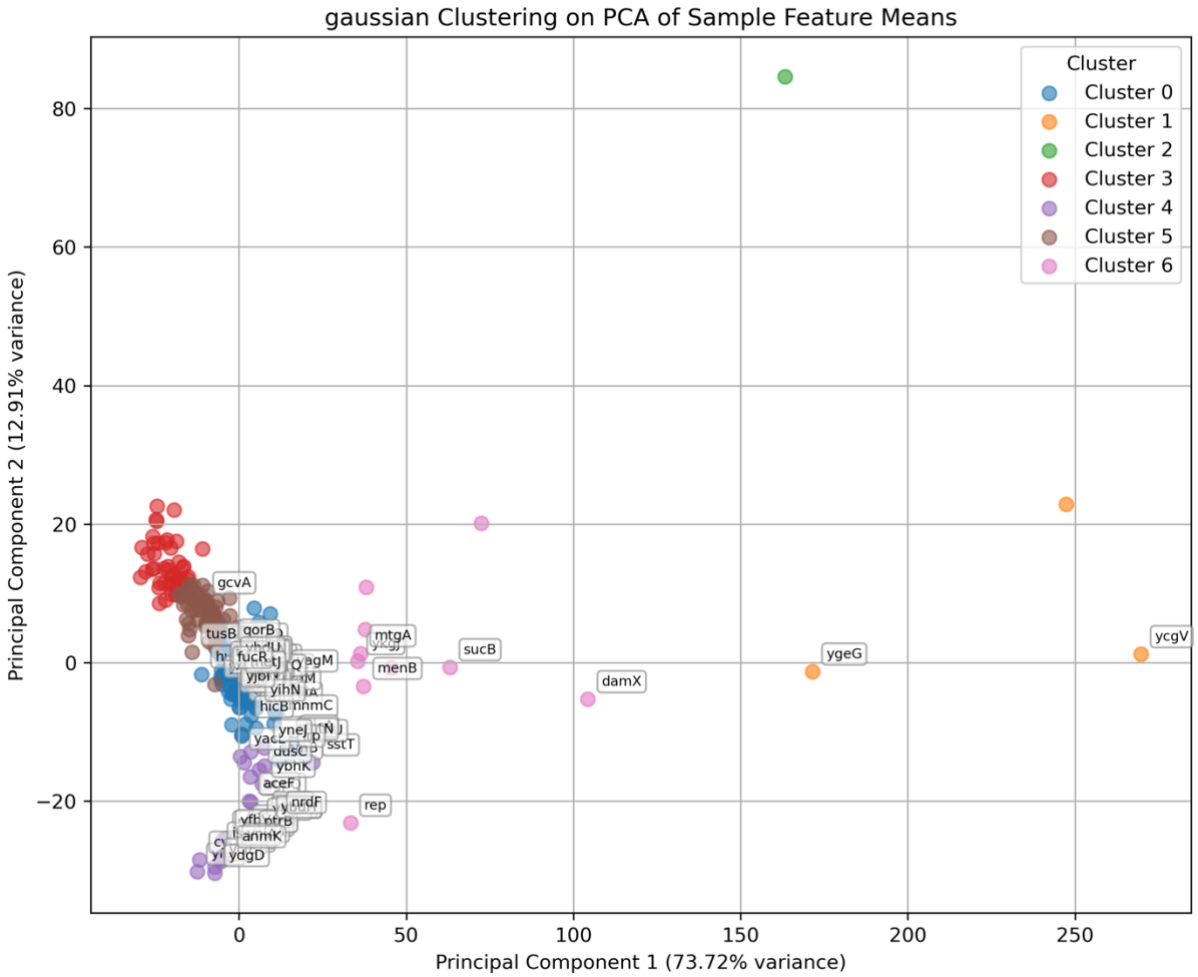
A.2.1: PCA plot showing phenotypic divergence of mutant deletion strains from the wild type. Wild type in blue, and mutant deletion strain in orange. Mutant samples are annotated with their deletion gene. PC1 captured 73.72% of the explained variance, and PC2 12.91%.



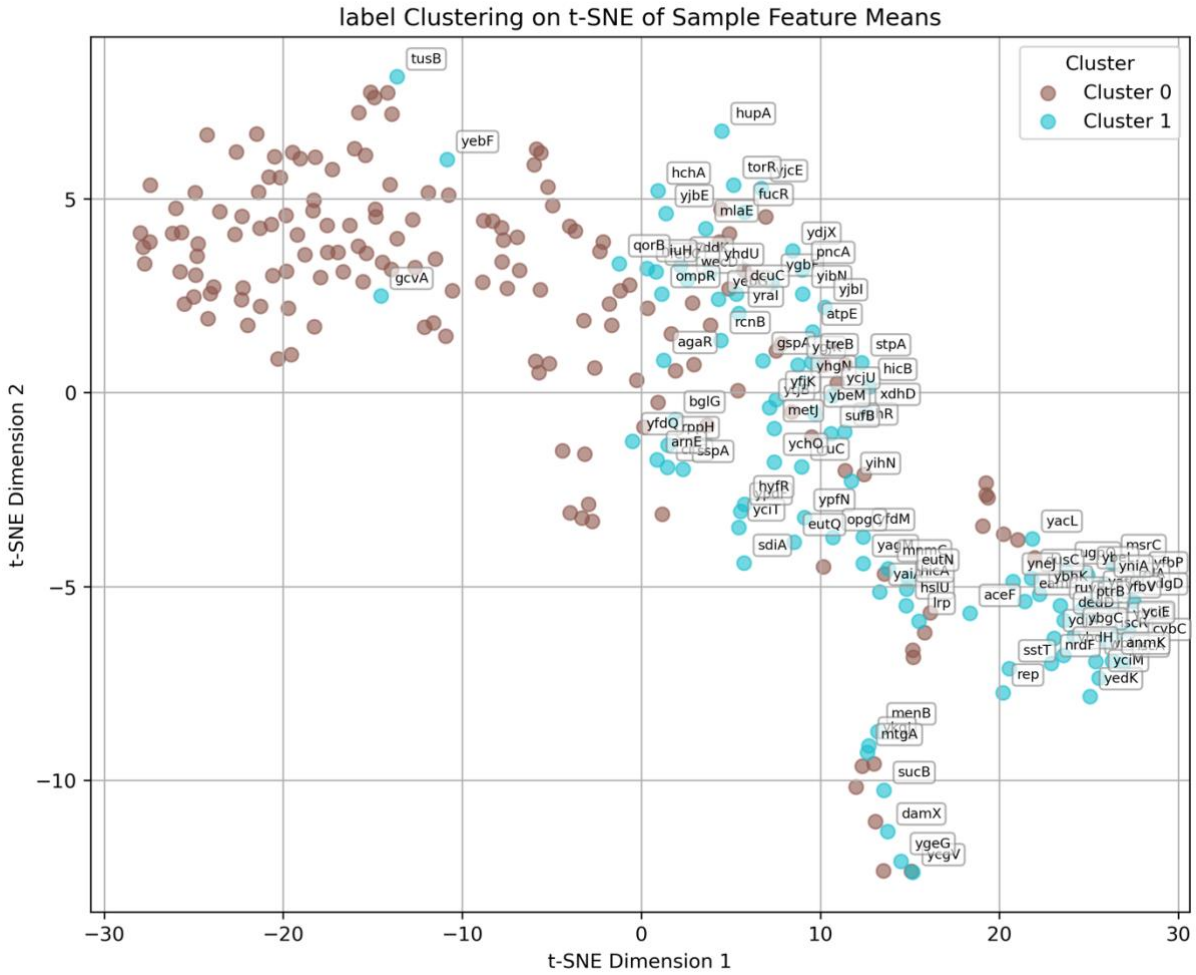
A.2.2: PCA plot showing HDBSCAN Clustering groups. Mutant samples are annotated with their deletion gene.



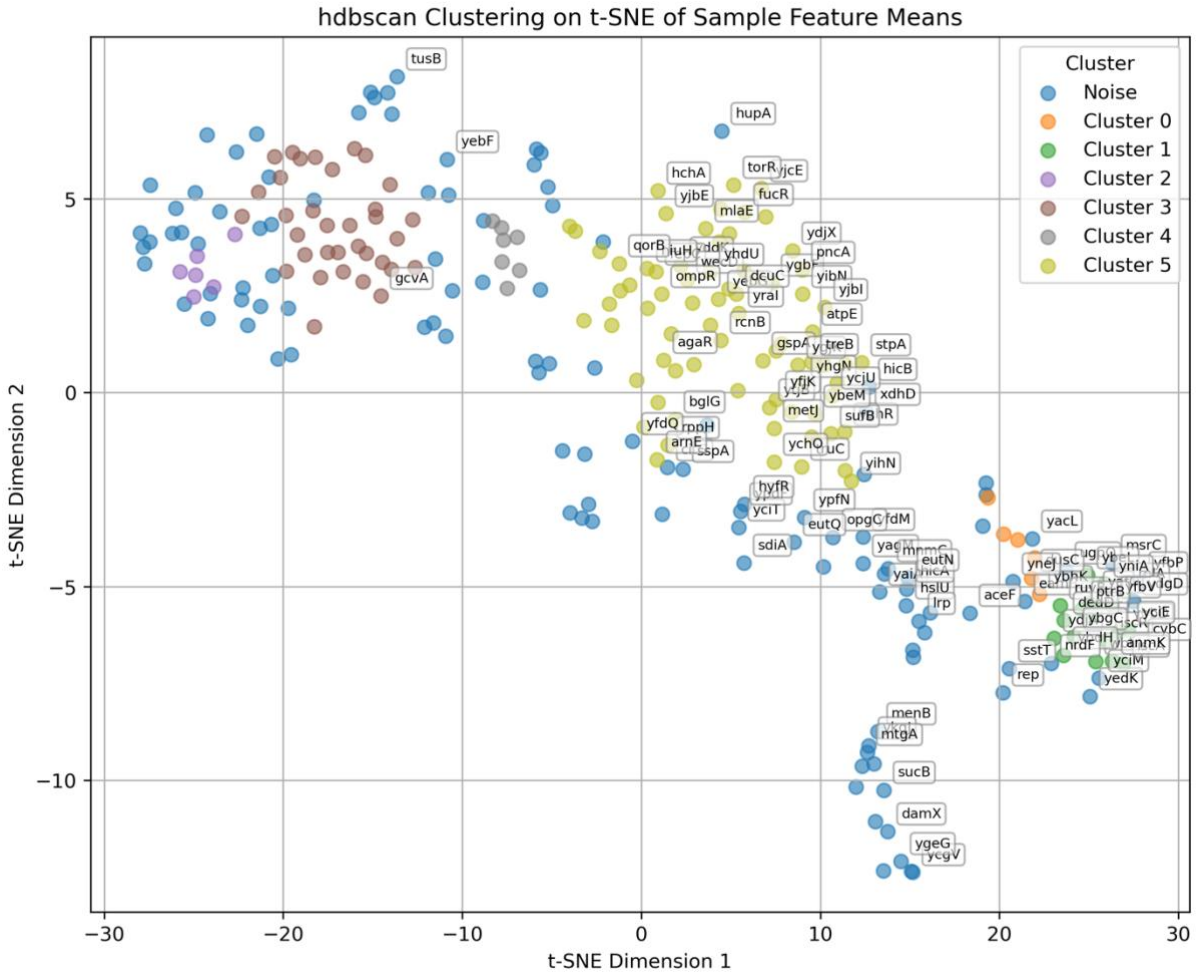
A.2.3: PCA plot showing k-Means Clustering groups. Mutant samples are annotated with their deletion gene.



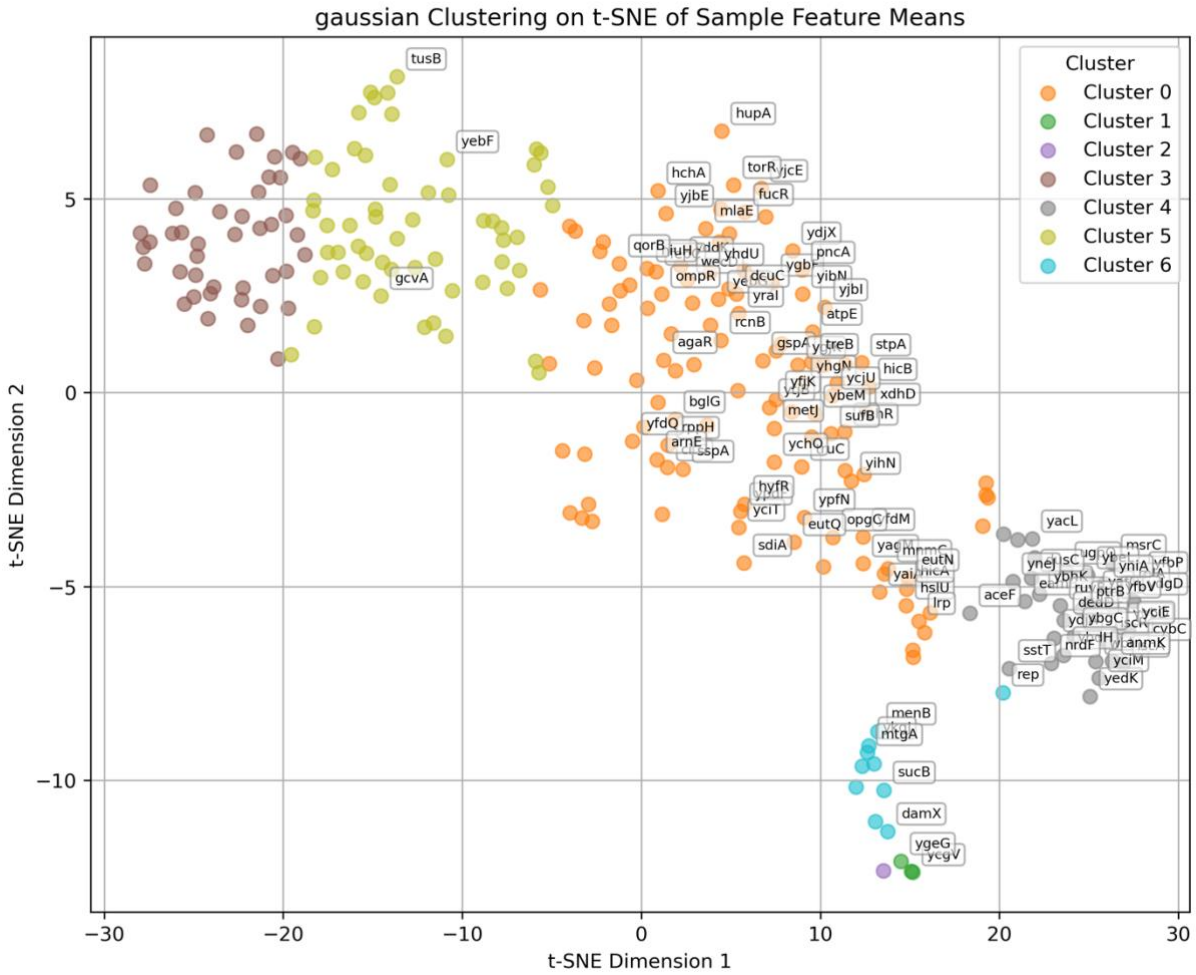
A.2.4: PCA plot showing Gaussian Mixture Models Clustering groups. Mutant samples are annotated with their deletion gene.



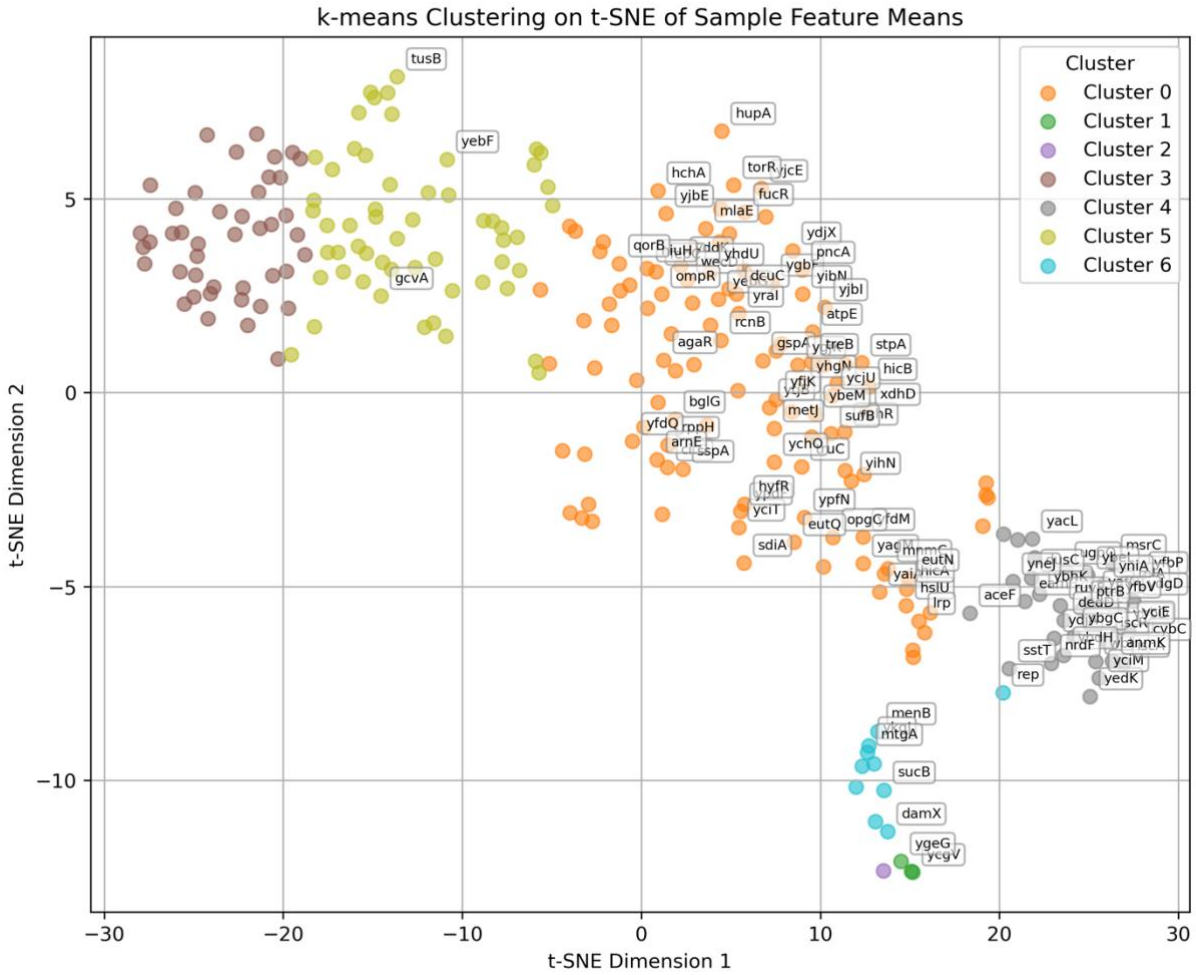
A.2.5: t-SNE plot showing wild-type and mutant deletion samples. Mutant samples are blue and annotated. Whereas wild type samples are brown.



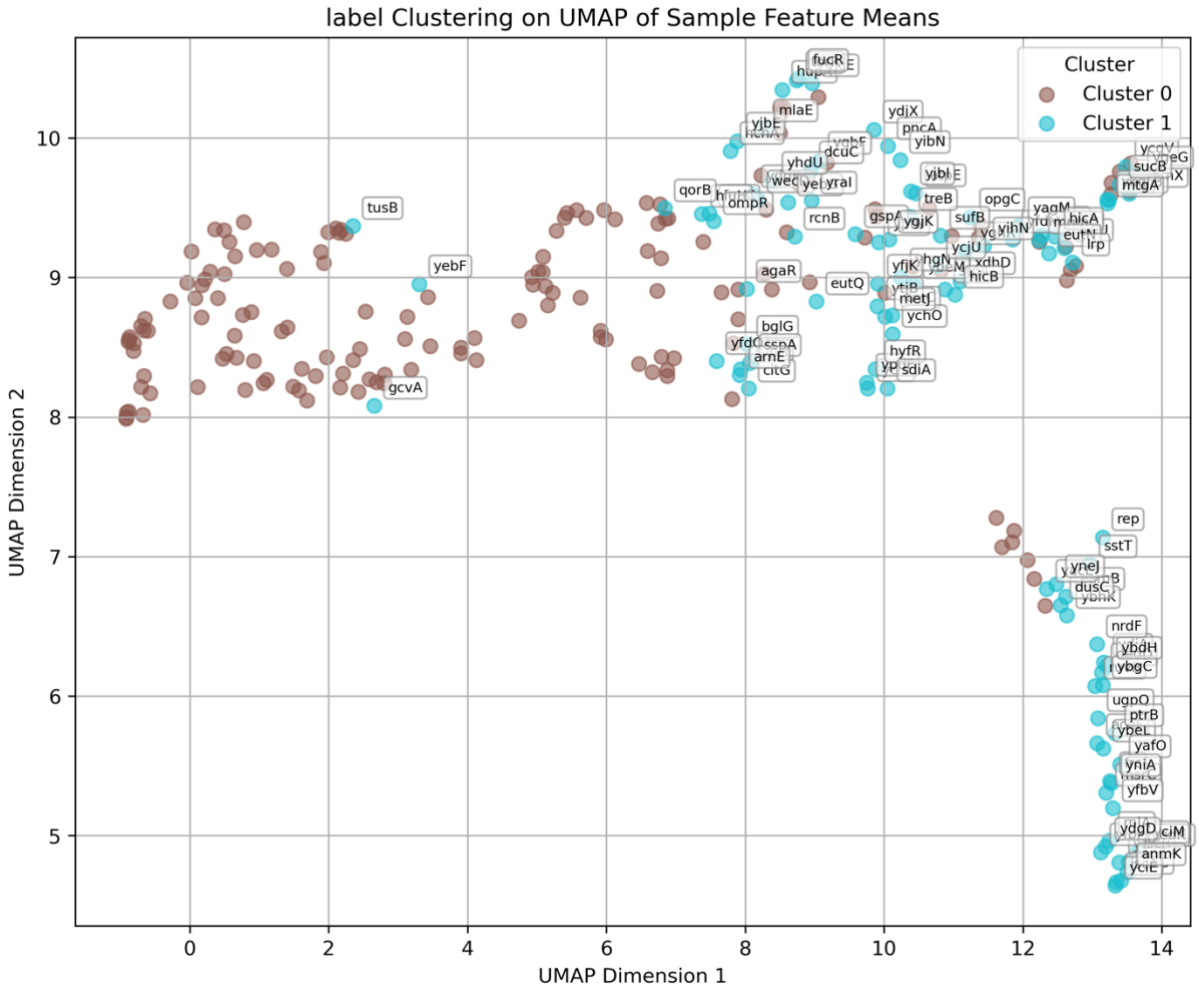
A.2.6: t-SNE plot showing HDBSCAN clustering of samples. Mutant samples are annotated.



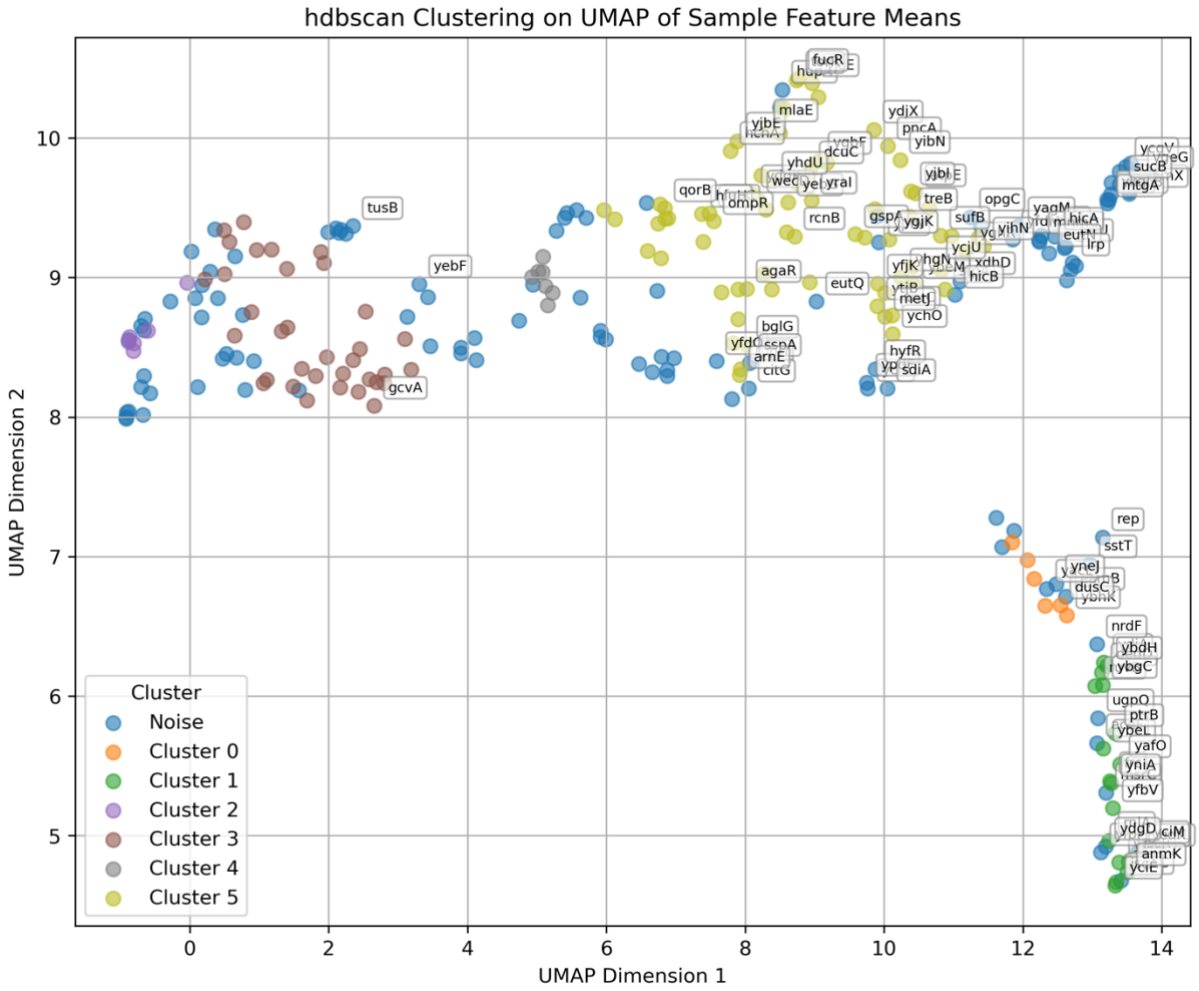
A.2.7: t-SNE plot showing Gaussian Mixture Model clustering of samples. Mutant samples are annotated.



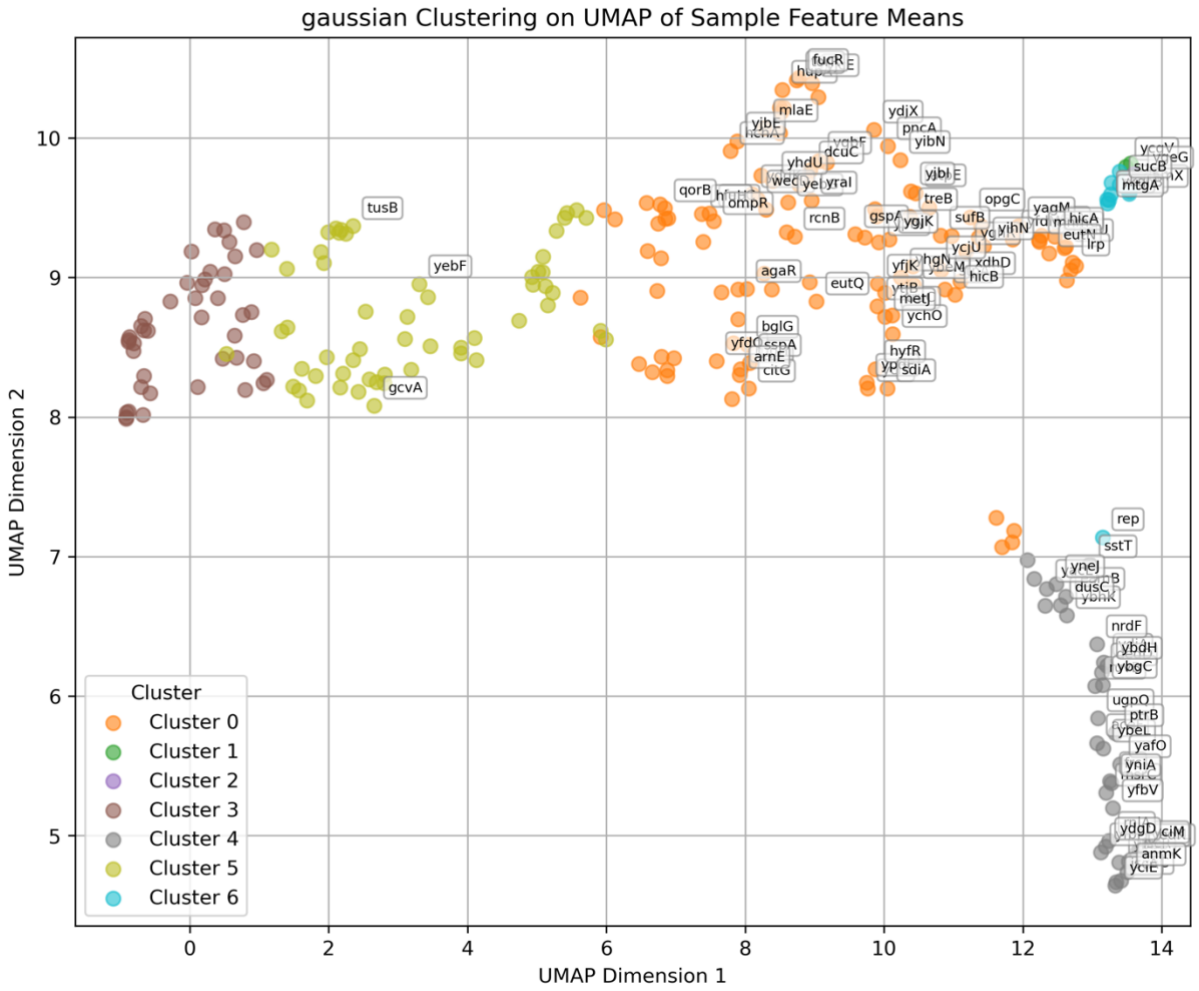
A.2.8: t-SNE plot showing k-Means clustering of samples. Mutant samples are annotated.



A.2.9: UMAP plot showing wild-type and mutant deletion samples. Mutant samples are blue and annotated. Whereas wild type samples are brown.

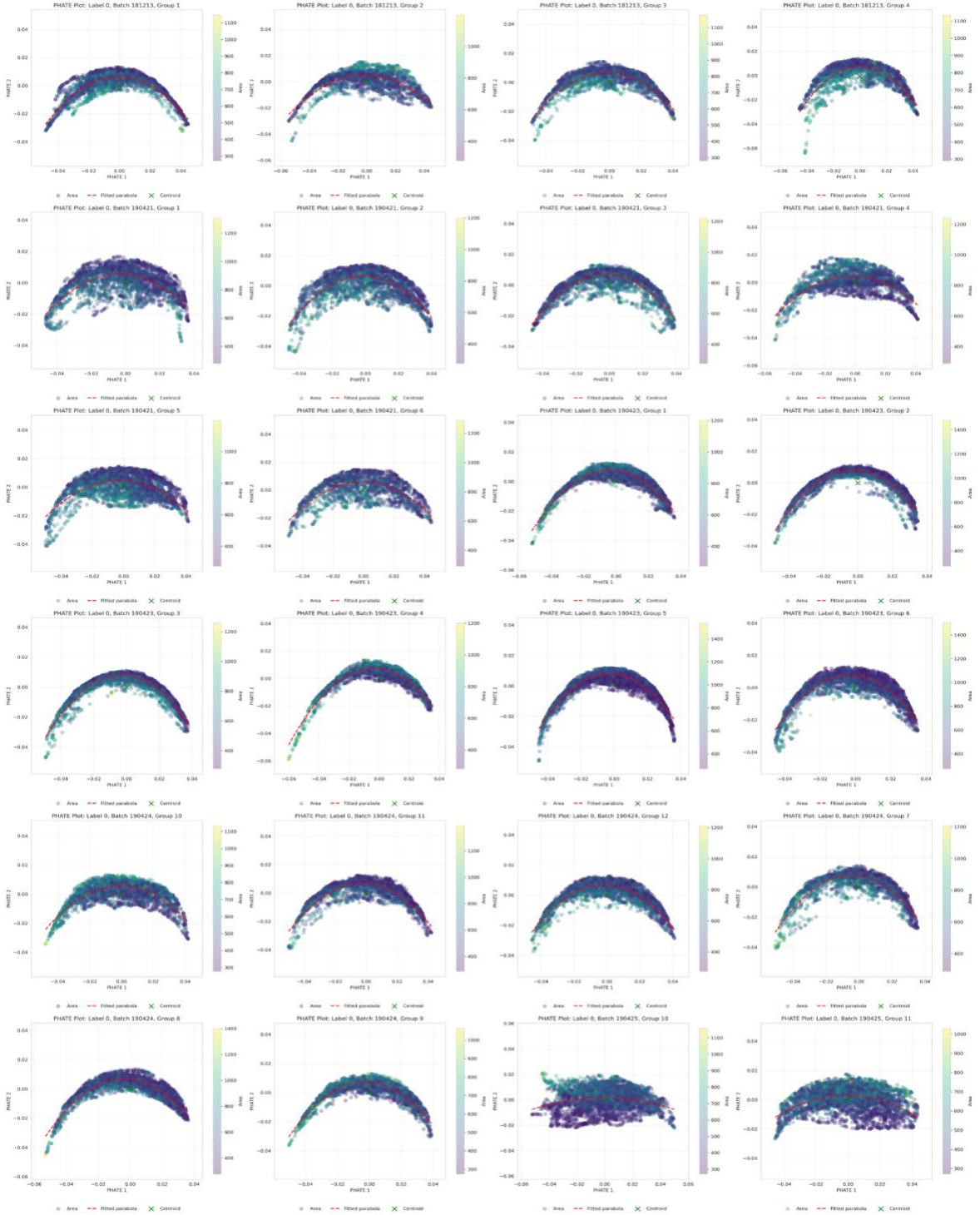


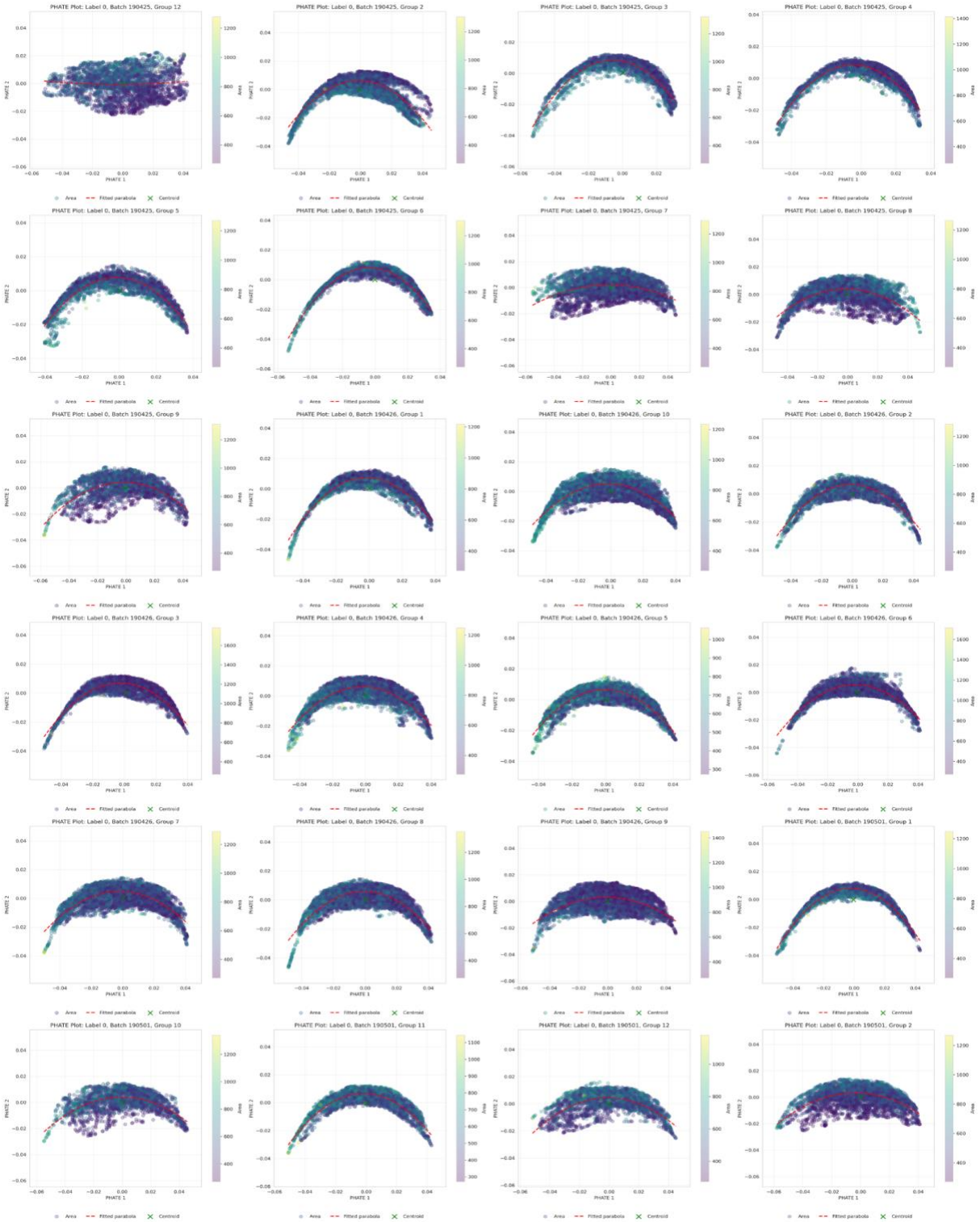
A.2.10: UMAP plot showing HDBSCAN clustering of samples. Mutant samples are annotated.

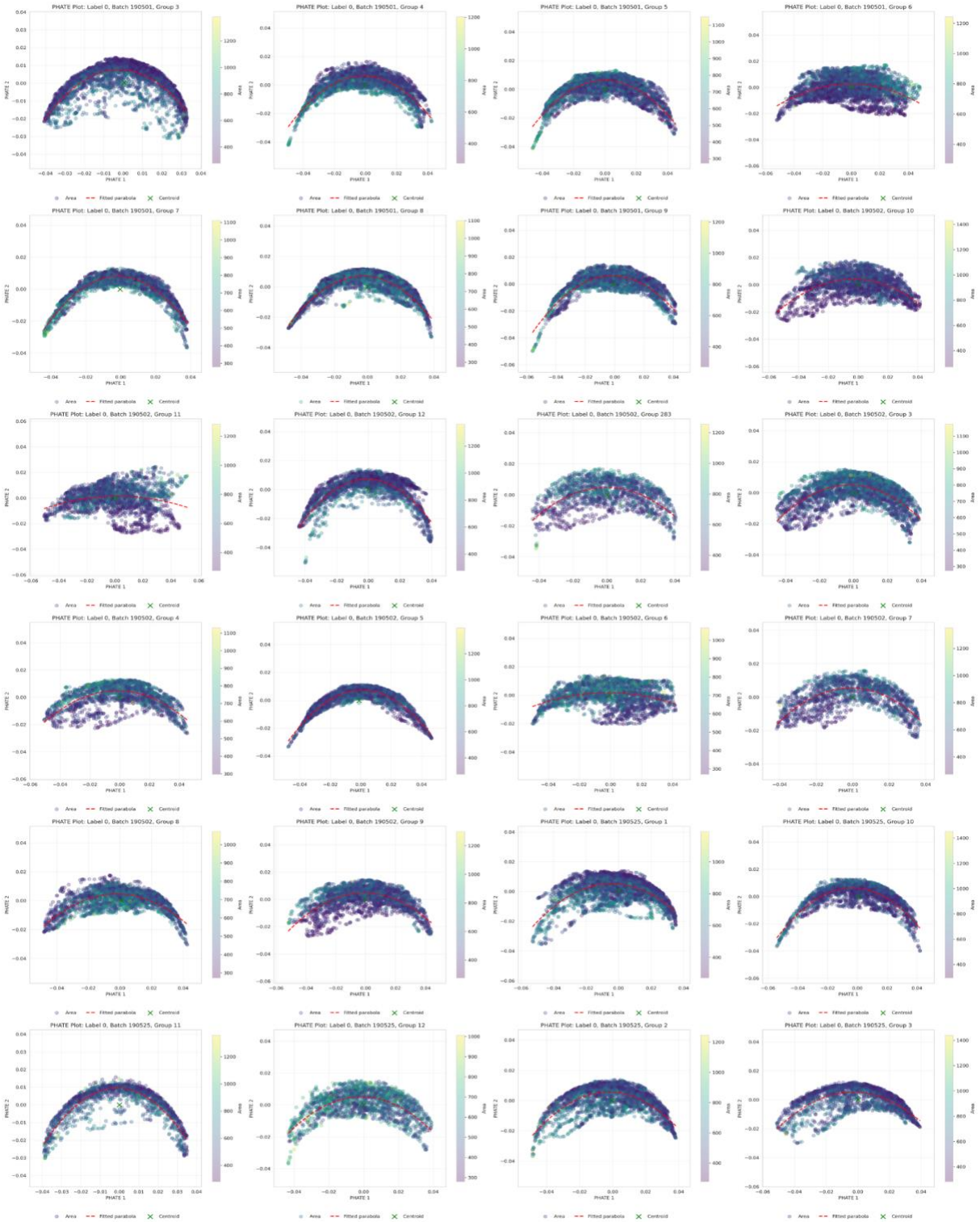


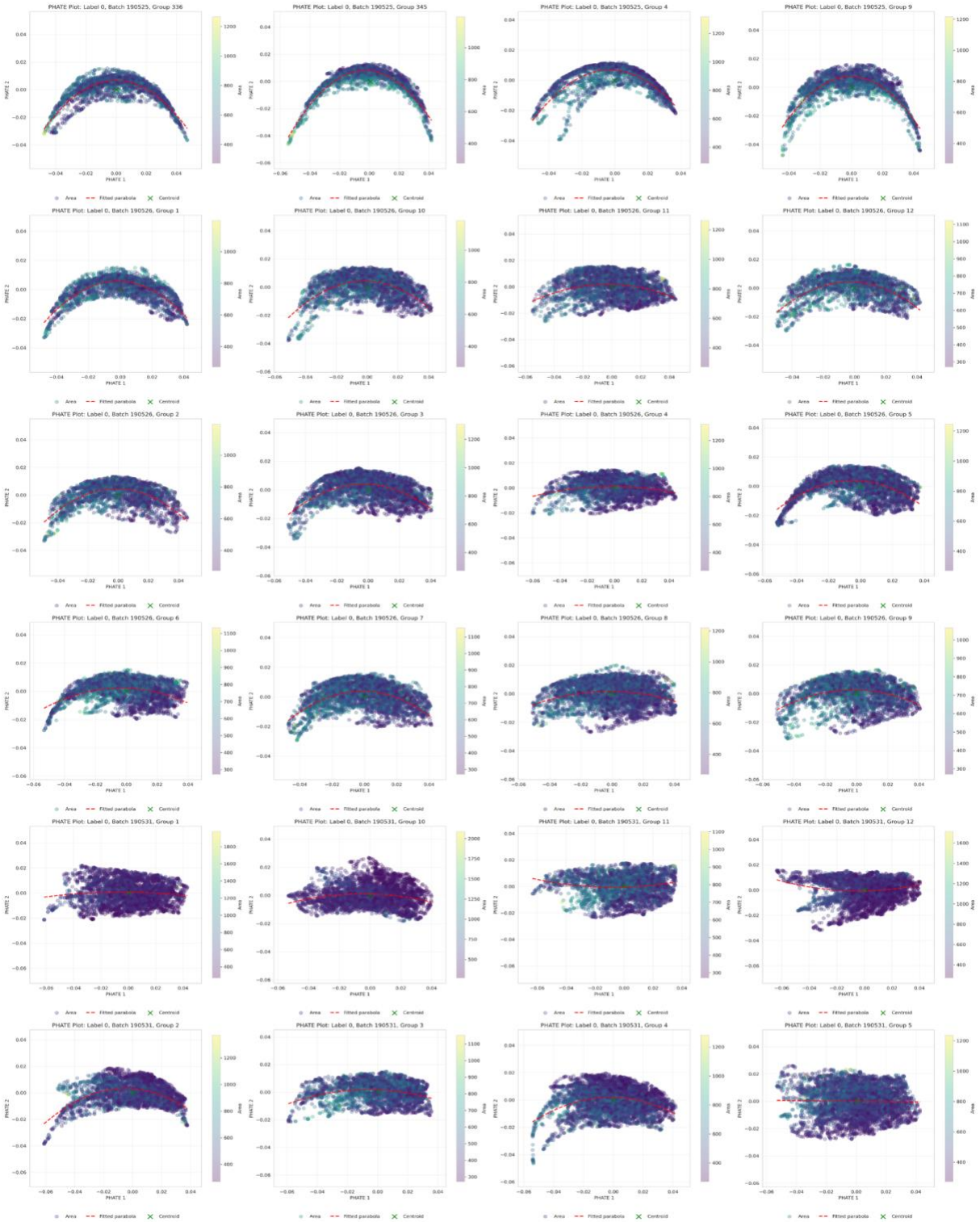
A.2.11: UMAP plot showing Gaussian Mixture Model clustering of samples. Mutant samples are annotated.

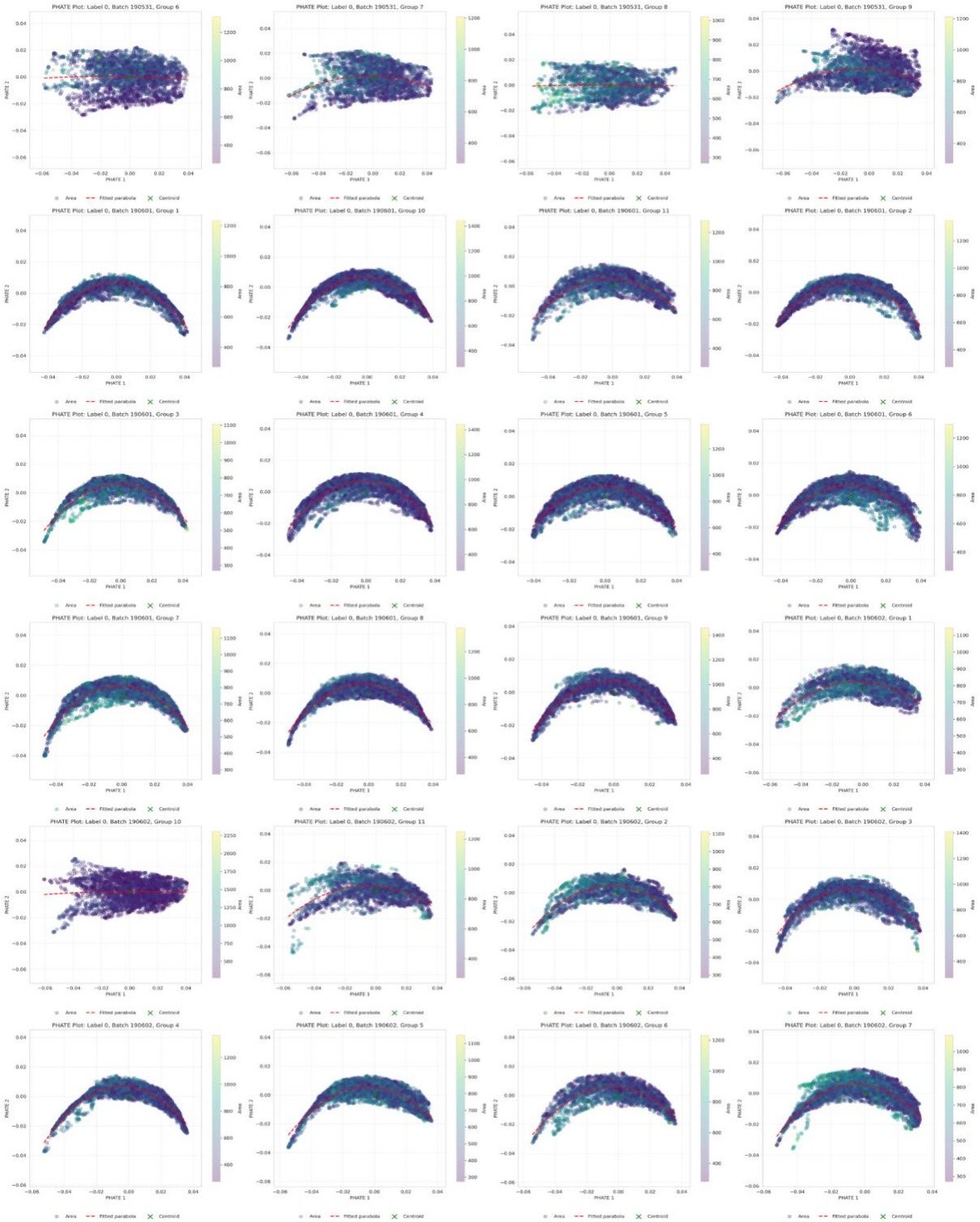
A.3 Wild Type PHATE plots

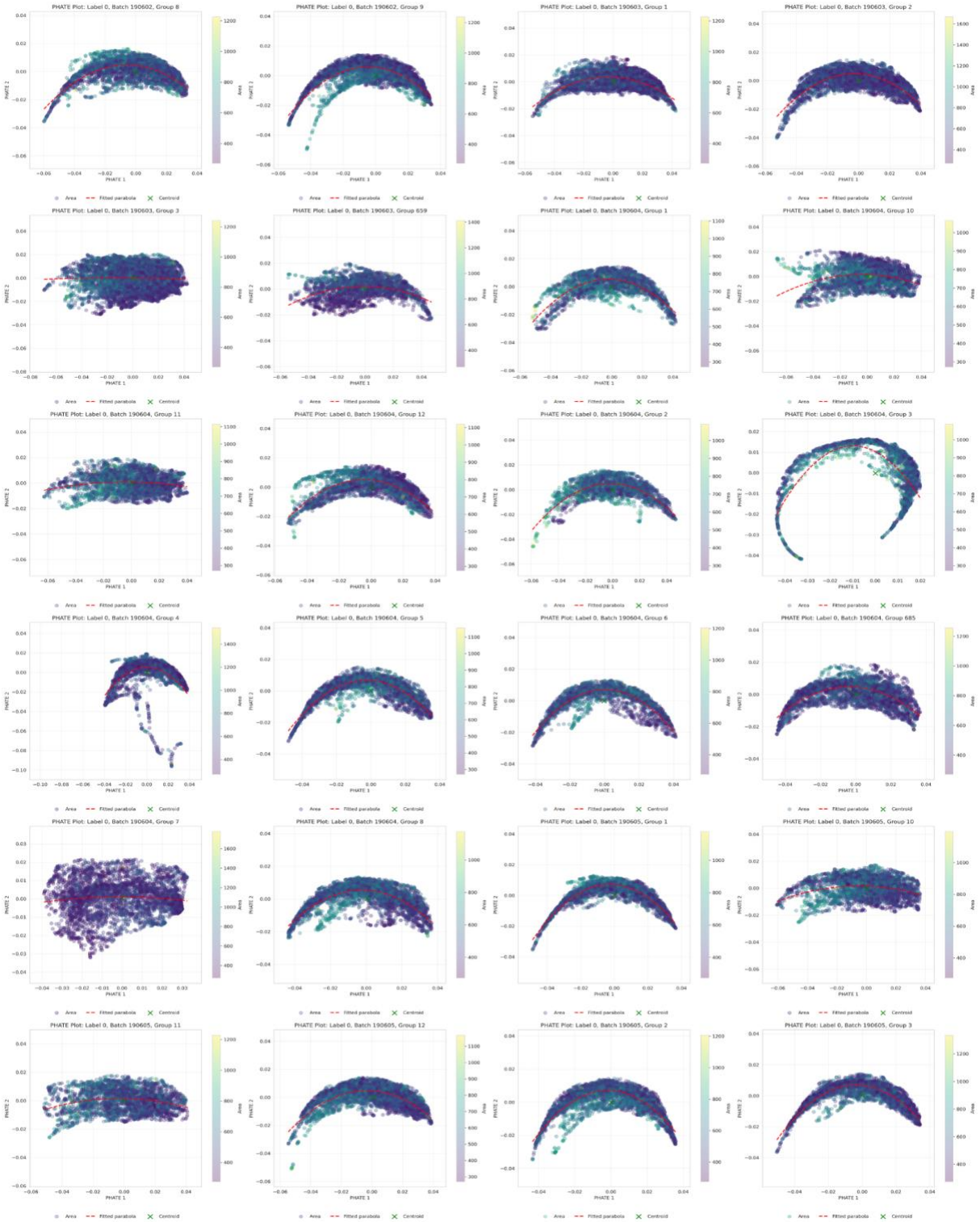


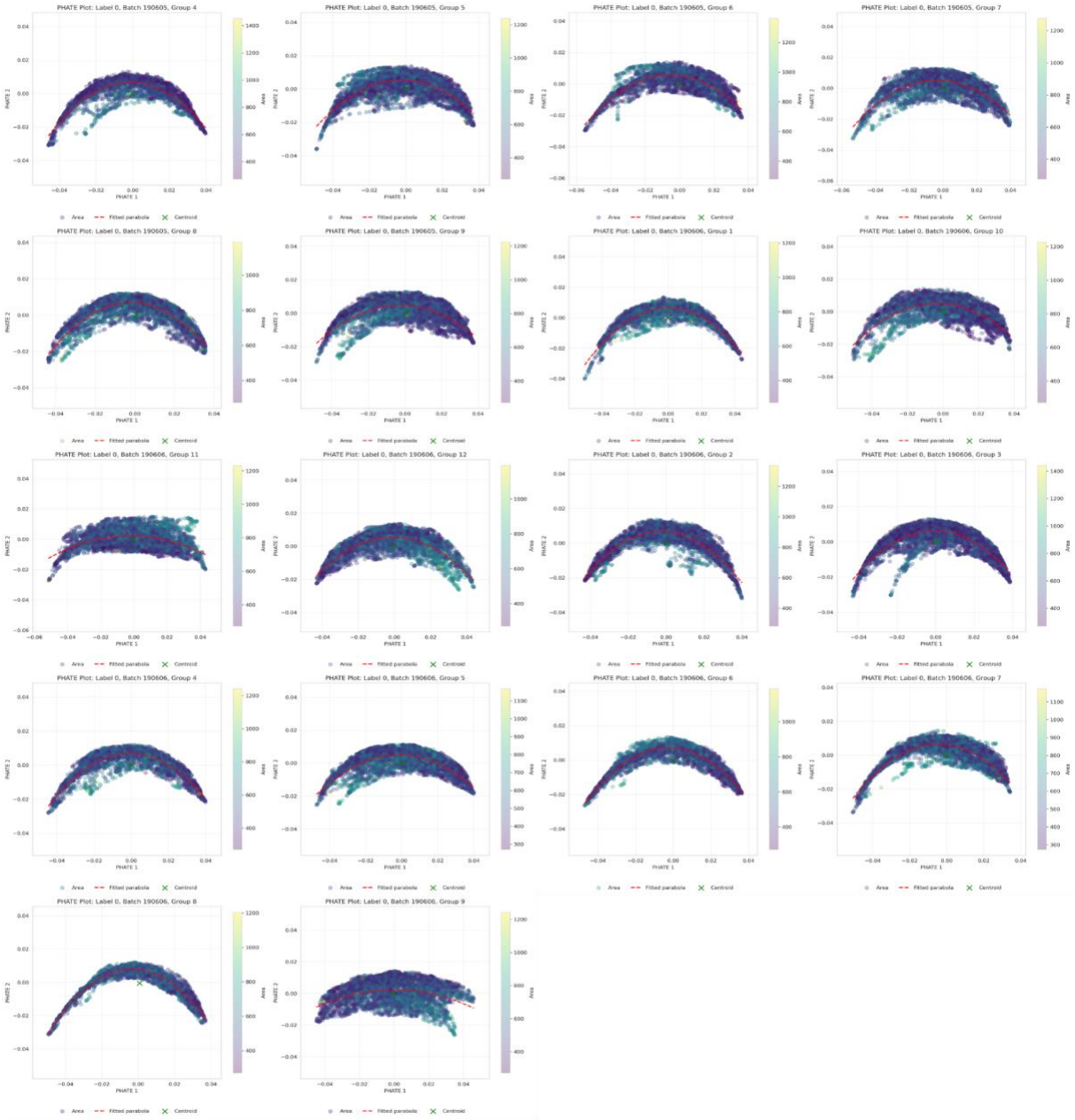

PHATE Plot: Label 0, Batch 190425, Group 12
PHATE Plot: Label 0, Batch 190425, Group 2
PHATE Plot: Label 0, Batch 190425, Group 3
PHATE Plot: Label 0, Batch 190425, Group 4
PHATE Plot: Label 0, Batch 190425, Group 5
PHATE Plot: Label 0, Batch 190425, Group 6
PHATE Plot: Label 0, Batch 190425, Group 7
PHATE Plot: Label 0, Batch 190425, Group 8
PHATE Plot: Label 0, Batch 190425, Group 9
PHATE Plot: Label 0, Batch 190426, Group 1
PHATE Plot: Label 0, Batch 190426, Group 10
PHATE Plot: Label 0, Batch 190426, Group 2
PHATE Plot: Label 0, Batch 190426, Group 3
PHATE Plot: Label 0, Batch 190426, Group 4
PHATE Plot: Label 0, Batch 190426, Group 5
PHATE Plot: Label 0, Batch 190426, Group 6
PHATE Plot: Label 0, Batch 190426, Group 7
PHATE Plot: Label 0, Batch 190426, Group 8
PHATE Plot: Label 0, Batch 190426, Group 9
PHATE Plot: Label 0, Batch 190501, Group 1
PHATE Plot: Label 0, Batch 190501, Group 10
PHATE Plot: Label 0, Batch 190501, Group 11
PHATE Plot: Label 0, Batch 190501, Group 12
PHATE Plot: Label 0, Batch 190501, Group 2
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid


PHATE Plot: Label 0, Batch 190501, Group 3
PHATE Plot: Label 0, Batch 190501, Group 4
PHATE Plot: Label 0, Batch 190501, Group 5
PHATE Plot: Label 0, Batch 190501, Group 6
PHATE Plot: Label 0, Batch 190501, Group 7
PHATE Plot: Label 0, Batch 190501, Group 8
PHATE Plot: Label 0, Batch 190501, Group 9
PHATE Plot: Label 0, Batch 190502, Group 10
PHATE Plot: Label 0, Batch 190502, Group 11
PHATE Plot: Label 0, Batch 190502, Group 12
PHATE Plot: Label 0, Batch 190502, Group 283
PHATE Plot: Label 0, Batch 190502, Group 3
PHATE Plot: Label 0, Batch 190502, Group 4
PHATE Plot: Label 0, Batch 190502, Group 5
PHATE Plot: Label 0, Batch 190502, Group 6
PHATE Plot: Label 0, Batch 190502, Group 7
PHATE Plot: Label 0, Batch 190502, Group 8
PHATE Plot: Label 0, Batch 190502, Group 9
PHATE Plot: Label 0, Batch 190525, Group 1
PHATE Plot: Label 0, Batch 190525, Group 10
PHATE Plot: Label 0, Batch 190525, Group 11
PHATE Plot: Label 0, Batch 190525, Group 12
PHATE Plot: Label 0, Batch 190525, Group 2
PHATE Plot: Label 0, Batch 190525, Group 3
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid

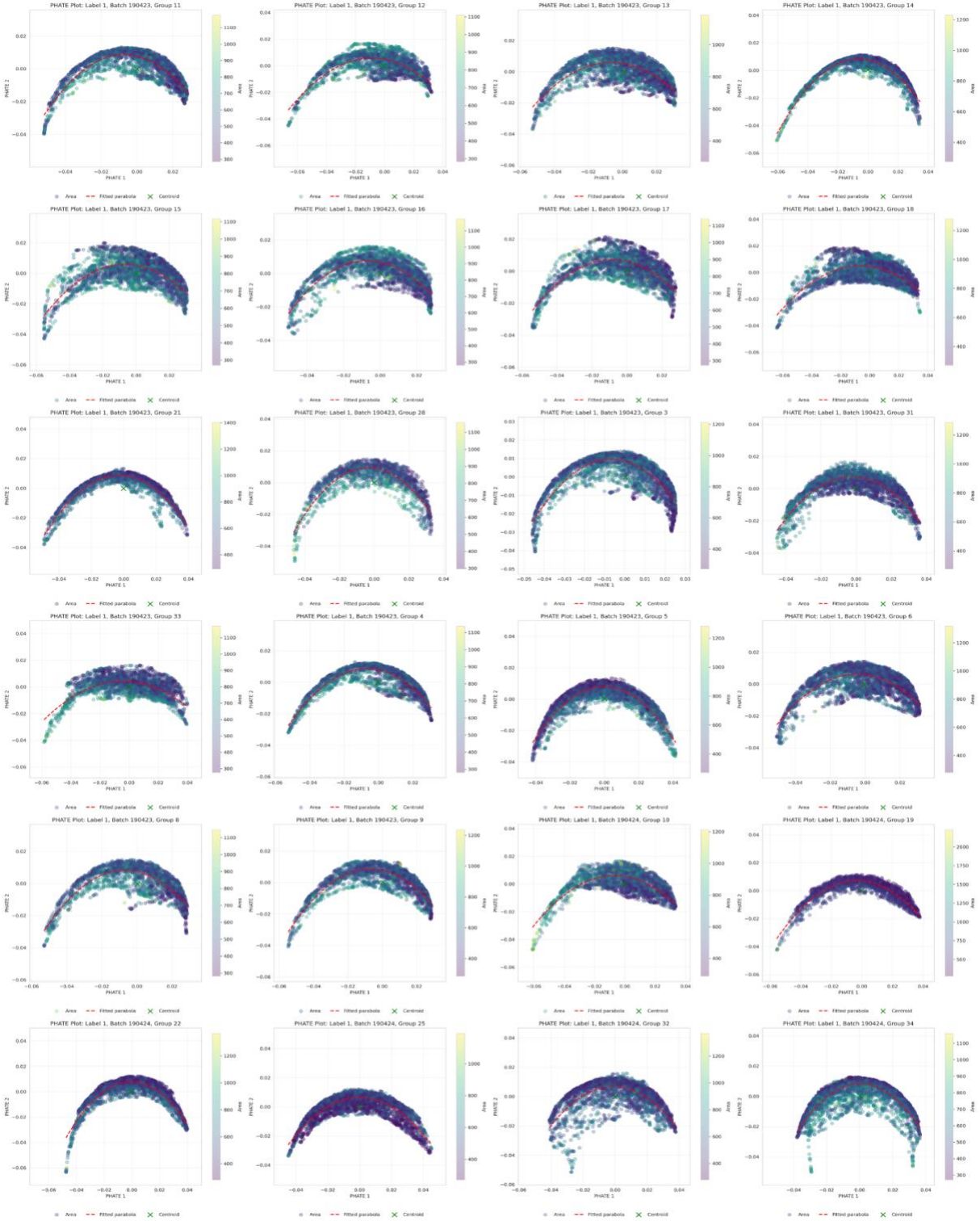

PHATE Plot: Label 0, Batch 190525, Group 336
PHATE Plot: Label 0, Batch 190525, Group 345
PHATE Plot: Label 0, Batch 190525, Group 4
PHATE Plot: Label 0, Batch 190525, Group 9
PHATE Plot: Label 0, Batch 190526, Group 1
PHATE Plot: Label 0, Batch 190526, Group 10
PHATE Plot: Label 0, Batch 190526, Group 11
PHATE Plot: Label 0, Batch 190526, Group 12
PHATE Plot: Label 0, Batch 190526, Group 2
PHATE Plot: Label 0, Batch 190526, Group 3
PHATE Plot: Label 0, Batch 190526, Group 4
PHATE Plot: Label 0, Batch 190526, Group 5
PHATE Plot: Label 0, Batch 190526, Group 6
PHATE Plot: Label 0, Batch 190526, Group 7
PHATE Plot: Label 0, Batch 190526, Group 8
PHATE Plot: Label 0, Batch 190526, Group 9
PHATE Plot: Label 0, Batch 190531, Group 1
PHATE Plot: Label 0, Batch 190531, Group 10
PHATE Plot: Label 0, Batch 190531, Group 11
PHATE Plot: Label 0, Batch 190531, Group 12
PHATE Plot: Label 0, Batch 190531, Group 2
PHATE Plot: Label 0, Batch 190531, Group 3
PHATE Plot: Label 0, Batch 190531, Group 4
PHATE Plot: Label 0, Batch 190531, Group 5
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid

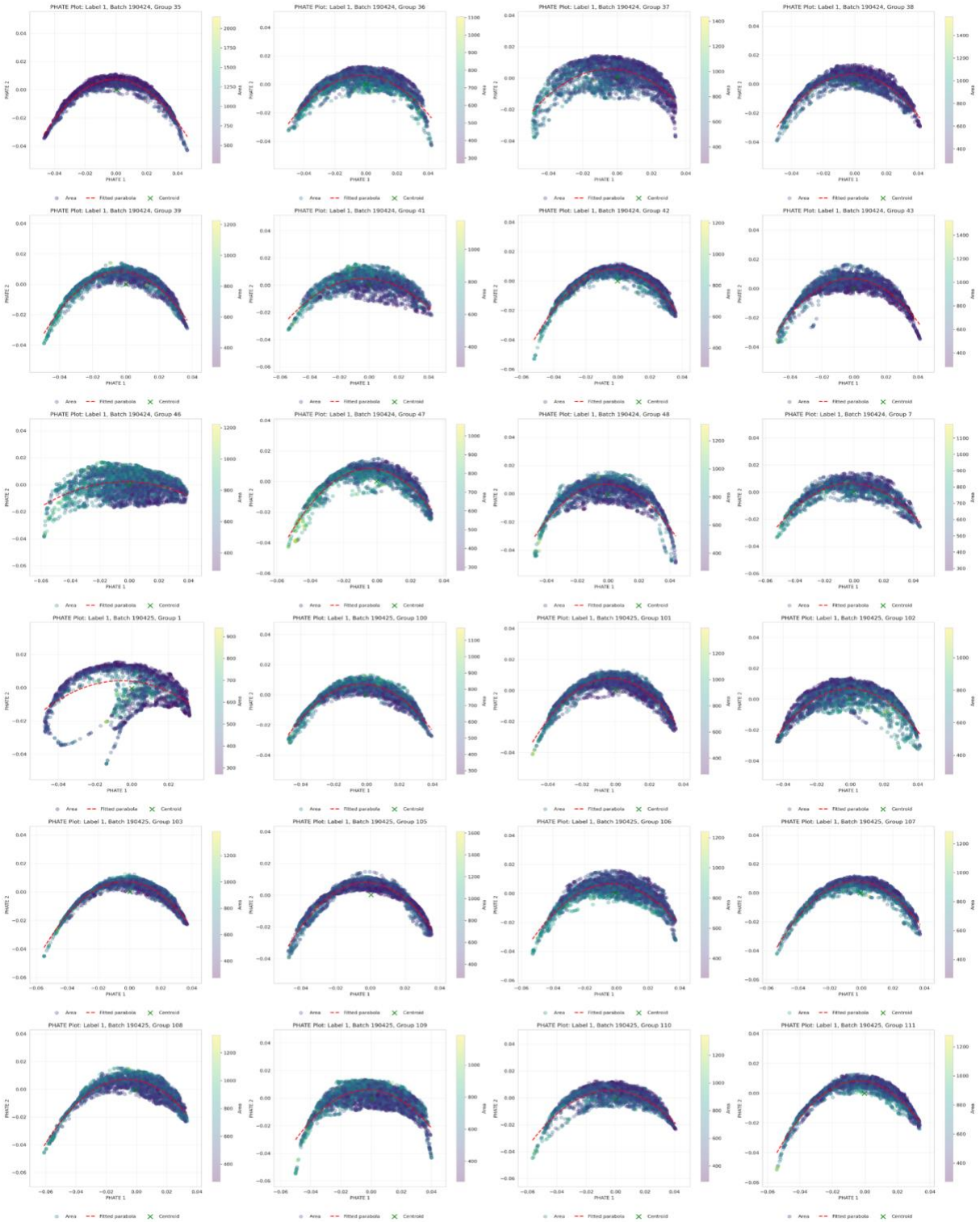

PHATE Plot: Label 0, Batch 190531, Group 6
PHATE Plot: Label 0, Batch 190531, Group 7
PHATE Plot: Label 0, Batch 190531, Group 8
PHATE Plot: Label 0, Batch 190531, Group 9
PHATE Plot: Label 0, Batch 190601, Group 1
PHATE Plot: Label 0, Batch 190601, Group 10
PHATE Plot: Label 0, Batch 190601, Group 11
PHATE Plot: Label 0, Batch 190601, Group 2
PHATE Plot: Label 0, Batch 190601, Group 3
PHATE Plot: Label 0, Batch 190601, Group 4
PHATE Plot: Label 0, Batch 190601, Group 5
PHATE Plot: Label 0, Batch 190601, Group 6
PHATE Plot: Label 0, Batch 190601, Group 7
PHATE Plot: Label 0, Batch 190601, Group 8
PHATE Plot: Label 0, Batch 190601, Group 9
PHATE Plot: Label 0, Batch 190602, Group 1
PHATE Plot: Label 0, Batch 190602, Group 10
PHATE Plot: Label 0, Batch 190602, Group 11
PHATE Plot: Label 0, Batch 190602, Group 2
PHATE Plot: Label 0, Batch 190602, Group 3
PHATE Plot: Label 0, Batch 190602, Group 4
PHATE Plot: Label 0, Batch 190602, Group 5
PHATE Plot: Label 0, Batch 190602, Group 6
PHATE Plot: Label 0, Batch 190602, Group 7
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid

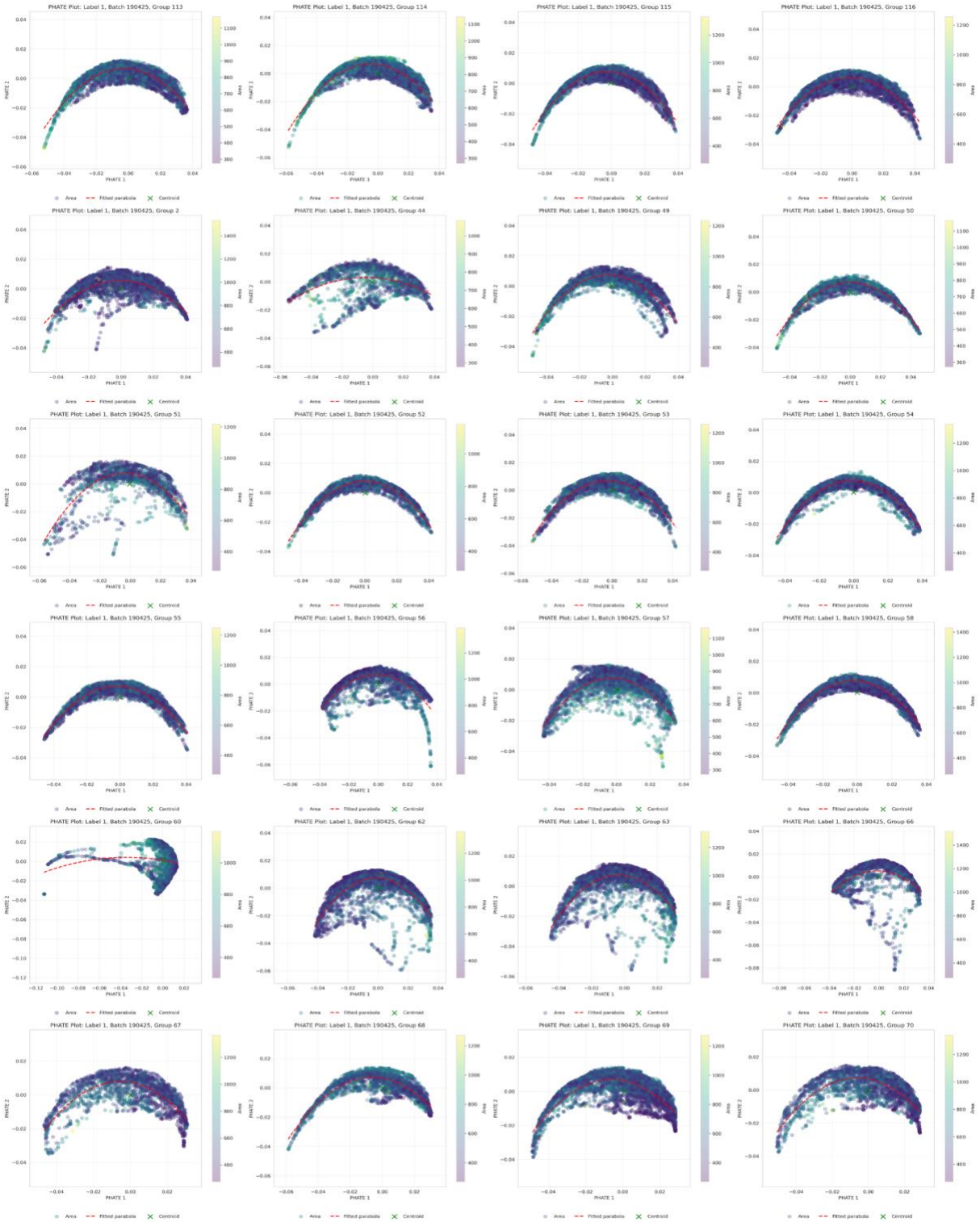

PHATE Plot: Label 0, Batch 190602, Group 8
PHATE Plot: Label 0, Batch 190602, Group 9
PHATE Plot: Label 0, Batch 190603, Group 1
PHATE Plot: Label 0, Batch 190603, Group 2
PHATE Plot: Label 0, Batch 190603, Group 3
PHATE Plot: Label 0, Batch 190603, Group 659
PHATE Plot: Label 0, Batch 190604, Group 1
PHATE Plot: Label 0, Batch 190604, Group 10
PHATE Plot: Label 0, Batch 190604, Group 11
PHATE Plot: Label 0, Batch 190604, Group 12
PHATE Plot: Label 0, Batch 190604, Group 2
PHATE Plot: Label 0, Batch 190604, Group 3
PHATE Plot: Label 0, Batch 190604, Group 4
PHATE Plot: Label 0, Batch 190604, Group 5
PHATE Plot: Label 0, Batch 190604, Group 6
PHATE Plot: Label 0, Batch 190604, Group 685
PHATE Plot: Label 0, Batch 190604, Group 7
PHATE Plot: Label 0, Batch 190604, Group 8
PHATE Plot: Label 0, Batch 190605, Group 1
PHATE Plot: Label 0, Batch 190605, Group 10
PHATE Plot: Label 0, Batch 190605, Group 11
PHATE Plot: Label 0, Batch 190605, Group 12
PHATE Plot: Label 0, Batch 190605, Group 2
PHATE Plot: Label 0, Batch 190605, Group 3
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid

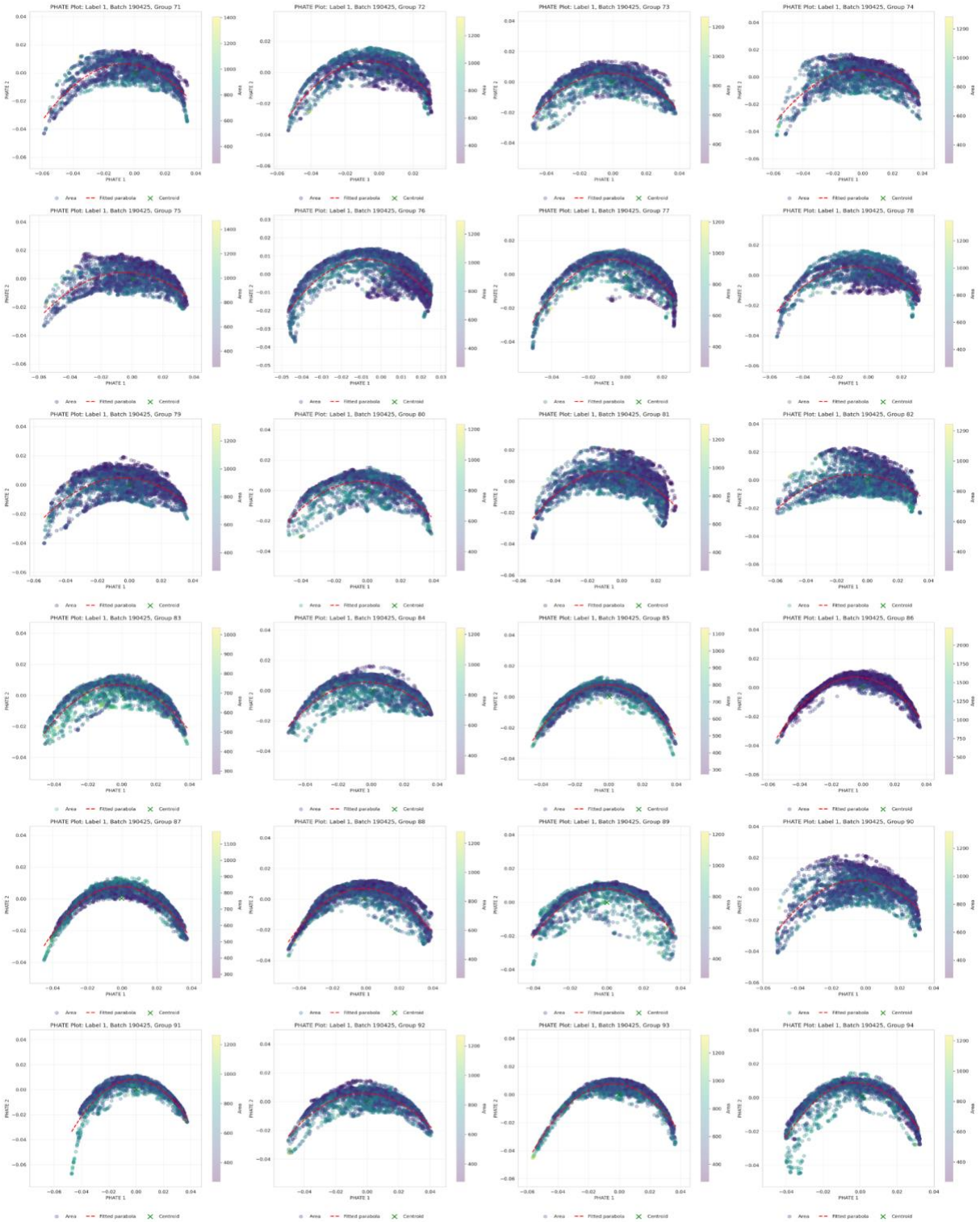

PHATE Plot: Label 0, Batch 190605, Group 4
PHATE Plot: Label 0, Batch 190605, Group 5
PHATE Plot: Label 0, Batch 190605, Group 6
PHATE Plot: Label 0, Batch 190605, Group 7
PHATE Plot: Label 0, Batch 190605, Group 8
PHATE Plot: Label 0, Batch 190605, Group 9
PHATE Plot: Label 0, Batch 190606, Group 1
PHATE Plot: Label 0, Batch 190606, Group 10
PHATE Plot: Label 0, Batch 190606, Group 11
PHATE Plot: Label 0, Batch 190606, Group 12
PHATE Plot: Label 0, Batch 190606, Group 2
PHATE Plot: Label 0, Batch 190606, Group 3
PHATE Plot: Label 0, Batch 190606, Group 4
PHATE Plot: Label 0, Batch 190606, Group 5
PHATE Plot: Label 0, Batch 190606, Group 6
PHATE Plot: Label 0, Batch 190606, Group 7
PHATE Plot: Label 0, Batch 190606, Group 8
PHATE Plot: Label 0, Batch 190606, Group 9
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid

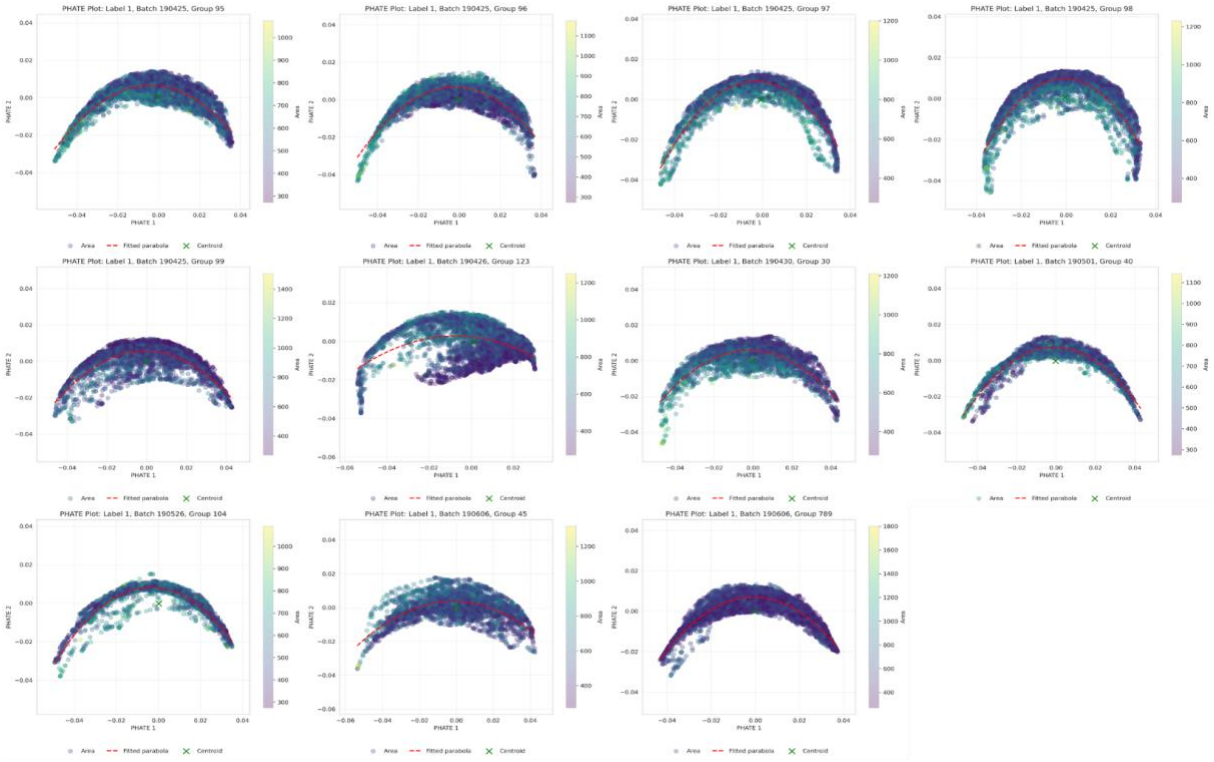
A.4 Mutants PHATE plots




PHATE Plot: Label 1, Batch 190424, Group 35
PHATE Plot: Label 1, Batch 190424, Group 36
PHATE Plot: Label 1, Batch 190424, Group 37
PHATE Plot: Label 1, Batch 190424, Group 38
PHATE Plot: Label 1, Batch 190424, Group 39
PHATE Plot: Label 1, Batch 190424, Group 41
PHATE Plot: Label 1, Batch 190424, Group 42
PHATE Plot: Label 1, Batch 190424, Group 43
PHATE Plot: Label 1, Batch 190424, Group 46
PHATE Plot: Label 1, Batch 190424, Group 47
PHATE Plot: Label 1, Batch 190424, Group 48
PHATE Plot: Label 1, Batch 190424, Group 7
PHATE Plot: Label 1, Batch 190425, Group 1
PHATE Plot: Label 1, Batch 190425, Group 100
PHATE Plot: Label 1, Batch 190425, Group 101
PHATE Plot: Label 1, Batch 190425, Group 102
PHATE Plot: Label 1, Batch 190425, Group 103
PHATE Plot: Label 1, Batch 190425, Group 105
PHATE Plot: Label 1, Batch 190425, Group 106
PHATE Plot: Label 1, Batch 190425, Group 107
PHATE Plot: Label 1, Batch 190425, Group 108
PHATE Plot: Label 1, Batch 190425, Group 109
PHATE Plot: Label 1, Batch 190425, Group 110
PHATE Plot: Label 1, Batch 190425, Group 111
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid


PHATE Plot: Label 1, Batch 190425, Group 113
PHATE Plot: Label 1, Batch 190425, Group 114
PHATE Plot: Label 1, Batch 190425, Group 115
PHATE Plot: Label 1, Batch 190425, Group 116
PHATE Plot: Label 1, Batch 190425, Group 2
PHATE Plot: Label 1, Batch 190425, Group 44
PHATE Plot: Label 1, Batch 190425, Group 49
PHATE Plot: Label 1, Batch 190425, Group 50
PHATE Plot: Label 1, Batch 190425, Group 51
PHATE Plot: Label 1, Batch 190425, Group 52
PHATE Plot: Label 1, Batch 190425, Group 53
PHATE Plot: Label 1, Batch 190425, Group 54
PHATE Plot: Label 1, Batch 190425, Group 55
PHATE Plot: Label 1, Batch 190425, Group 56
PHATE Plot: Label 1, Batch 190425, Group 57
PHATE Plot: Label 1, Batch 190425, Group 58
PHATE Plot: Label 1, Batch 190425, Group 60
PHATE Plot: Label 1, Batch 190425, Group 62
PHATE Plot: Label 1, Batch 190425, Group 63
PHATE Plot: Label 1, Batch 190425, Group 66
PHATE Plot: Label 1, Batch 190425, Group 67
PHATE Plot: Label 1, Batch 190425, Group 68
PHATE Plot: Label 1, Batch 190425, Group 69
PHATE Plot: Label 1, Batch 190425, Group 70
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid


PHATE Plot: Label 1, Batch 190425, Group 71
PHATE Plot: Label 1, Batch 190425, Group 72
PHATE Plot: Label 1, Batch 190425, Group 73
PHATE Plot: Label 1, Batch 190425, Group 74
PHATE Plot: Label 1, Batch 190425, Group 75
PHATE Plot: Label 1, Batch 190425, Group 76
PHATE Plot: Label 1, Batch 190425, Group 77
PHATE Plot: Label 1, Batch 190425, Group 78
PHATE Plot: Label 1, Batch 190425, Group 79
PHATE Plot: Label 1, Batch 190425, Group 80
PHATE Plot: Label 1, Batch 190425, Group 81
PHATE Plot: Label 1, Batch 190425, Group 82
PHATE Plot: Label 1, Batch 190425, Group 83
PHATE Plot: Label 1, Batch 190425, Group 84
PHATE Plot: Label 1, Batch 190425, Group 85
PHATE Plot: Label 1, Batch 190425, Group 86
PHATE Plot: Label 1, Batch 190425, Group 87
PHATE Plot: Label 1, Batch 190425, Group 88
PHATE Plot: Label 1, Batch 190425, Group 89
PHATE Plot: Label 1, Batch 190425, Group 90
PHATE Plot: Label 1, Batch 190425, Group 91
PHATE Plot: Label 1, Batch 190425, Group 92
PHATE Plot: Label 1, Batch 190425, Group 93
PHATE Plot: Label 1, Batch 190425, Group 94
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid


PHATE Plot: Label 1, Batch 190425, Group 95
PHATE Plot: Label 1, Batch 190425, Group 96
PHATE Plot: Label 1, Batch 190425, Group 97
PHATE Plot: Label 1, Batch 190425, Group 98
PHATE Plot: Label 1, Batch 190425, Group 99
PHATE Plot: Label 1, Batch 190426, Group 123
PHATE Plot: Label 1, Batch 190430, Group 30
PHATE Plot: Label 1, Batch 190501, Group 40
PHATE Plot: Label 1, Batch 190526, Group 104
PHATE Plot: Label 1, Batch 190606, Group 45
PHATE Plot: Label 1, Batch 190606, Group 789
PHATE 1
PHATE 2
Area
Fitted parabola
Centroid

Use of Generative Artificial Intelligence (GenAI)

Student name: Theodoro Gasperin Terra Camargo
Student number: r0974221

This form is related to my master's thesis.

Title master's thesis: Deep Learning Methods for Bacterial Image-Based Profiling
Promoter: Sander Govers

Please indicate with "X":

- o I did not use GenAI tools.

X I did use GenAI tools. In this case specify which ones: *ChatGPT, Grok, Gemini, DeepSeek*

Which way you were using it:

- As a language assistant for reviewing or improving texts you wrote yourself, provided that the model does not add new content. In this case, the use of GenAI is similar to the spelling and grammar check tools we already have today, so you do not need to explicitly mention using GenAI for this).
- As a search engine to get initial information on a topic or to make an initial search for existing research on the topic. (This way of gathering information is similar to using an ordinary search engine when working on an assignment. As a student, you are responsible for checking and verifying the absence and correctness of references. Therefore, after this initial search, look for scientific sources and conduct your own analysis of the source documents. Interpret, analyze and process the information you obtained; don't just copy-paste it. If you then write your own text based on this information, you do not have to mention you used GenAI.)
- To generate some code as part of a larger assignment. (Watch out, this can only be done if the teacher/promotor explicitly allows it.)

Further important guidelines and remarks:

The faculty follows the KU Leuven policy regarding responsible use of GenAI. This form is an aid towards transparency about the use of GenAI by the student which is essential. Irresponsible and non-transparent use of GenAI can be considered an irregularity and can be sanctioned. Students who consider using GenAI should inform themselves through the university website concerning the additional guidelines (How to correctly quote and refer to GenAI? What is (not) allowed? Tips and points of attention for responsible use):
https://www.kuleuven.be/english/education/student/educational-tools/generative-artificial-intelligence